PROCEEDINGS

HP 3000 IUG Amsterdam Conference '85

* Sponsored by Interex, The International Association of Hewlett-Packard Computer Users

Amsterdam, The Netherlands, **March 31 - April 5, 1985**
Internationaal Congrescentrum RAI

PROCEEDINGS

HP 3000 IUG Amsterdam Conference '85

* Sponsored by Interex, The International Association of Hewlett-Packard Computer Users

Amsterdam, The Netherlands, **March 31 - April 5, 1985**
Internationaal Congrescentrum RAI

The technical papers are grouped according to the nine subject categories. Within these categories the alphabetical order of the first author has been followed.

Papers are reproduced exactly as they were submitted. Scientific, grammatical or typographical inaccuracies have not been corrected.

c/o Interex
2570 El Camino Real West
4th floor
Mountain View, Ca. 94040
USA

Printed in The Netherlands
by order of Interex, The International Association of Hewlett-Packard Computer Users.

COMMITTEES

Organizing Committee

H.A.G. van Roosmalen	-Chairman
D. Leeksma	-Secretary/HP Liaison
H. Swart	-Treasurer
J. Grim	-Public Relations/ Entertainment Manager

Program Committee

H. Swart

W. Derksen

Vendor Show Committee

H.A.G. van Roosmalen	
D. Leeksma	
F. Bührer	-System Manager

PCO

Organisatie Bureau Amsterdam bv
Europaplein
1078 GZ Amsterdam
The Netherlands

TOPICS OF THE CONFERENCE

The technical papers are divided in the following subject categories:

AL - ADVANCED LANGUAGES
This category contains contributions pertaining to "4th" and "5th" generation languages, as well as program generators.

BS - BUSINESS SYSTEMS
Applications for the business environment are disposed.

DB - DATA BASE MANAGEMENT SYSTEMS
Papers dealing with the traditional DBMS fall in here.

DC - DATA COMMUNICATIONS AND NETWORKS
The distinction between networking from the DC perspective and OA is sometimes narrow. Where the emphasise is put unto data transport, it falls into this category.

OA - OFFICE AUTOMATION AND NETWORKS
This group contains contributions about OA as well as PC's as stand alone or integrated in the network. The latter where the multi-purpose station character is described.

OP - OPERATIONS MANAGEMENT
Sessions with Data Centre support systems fall into this group.

SD - SYSTEM DEVELOPMENT
Contributions of importance to the application software development staff's.

SM - SYSTEM MANAGEMENT
Lectures, tools and experiences with the present and future operating - and other - support systems, the bread and butter of the system programmers staff's fall into this category.

VS - VARIOUS
All other papers which cannot be put in the previous groups are gathered under this mnenomic.

I N T R O D U C T I O N

It is with much pleasure that the host committee of the HP3000 IUG Amsterdam Conference '85 presents to you the proceedings of this Conference.
The success of the Conference is greatly determined by the quality of the technical speakers. We have received a fine selection of outstanding papers for which we are very obliged.
You will find the contributions within the proceedings arranged by subject categories, which we consider reasonably appropriate.
People and especially computer people are difficult to put between brackets, so we realise that, in your opinion, their might be deficiencies.
Nevertheless, we hope to give you sufficient güidance for your convenience.

We succeeded in having a plenary session by a knowledgeable and renowned speaker at the beginning of each Conference day. It was not possible for all Keynote Speakers to submit their full paper, but at least a short biography is listed.

An absolute novelty as far as we know is the student contest in conjunction with this Conference. A grant of ADAGER made it possible for us to offer a prize to university students. To participate in this competition students had to send in a paper on their work with HP-3000 computers during their study.
We are happy to include in these proceedings the winning paper by Elisabeth Forster. We hope this initiative will find an echo in the forthcoming Conferences.

We planned 80 papers to be presented at the Conference and received 140 abstracts. It was a difficult task to make a selection of abstracts that will arouse the interest of the Conference participants. I am grateful to Wim Derksen, Jelle Grim and Jan Moen for their assistance. Eventually we selected 80 papers. We regret that there is no possibility to schedule more contributions.

We are also grateful to Pam Tower of HP. She was of great help in gathering the contributions from HP speakers from the USA.

We hope this Conference will be a valuable contribution to your present and future work and that it will bring you many useful ideas.

Program Chairman

Haije Swart

CONTENTS

Page

INVITED SPEAKERS

KEYNOTE SPEAKERS:

Dreyfus, Ph.. 3
The need for future 5th generation hard/software?

Herschberg, Prof.dr. I.S.......................... 5
Malice in Bitland

Hudson, Dr. P.T.W................................. 17
Artificial intelligence and the cognitive approach to software: The programmer's apprentice

Jepson, Ch.W...................................... 21
HP Overview

ADAGER AWARD WINNING PAPER:

Forster, Elizabeth.R............................. 23
The ultimate challenge - perfecting the man - machine interface

AL - ADVANCED LANGUAGES

Aalders, Kenneth.................................. 40
How fourth generation languages improve the man-machine interface

Bishop, Sharon.................................... 53
AI in the commercial marketplace

Kellogg, Harry and Bale, Jon...................... 59
Native language support for the HP3000

Kemp, Larry....................................... 74
Getting the most from Transact/3000

Laiho, Martti and Harris, Kim..................... 81
PPL/3000 - a user interface programming language

Leeuw, Hans van der............................... 87
An integrated approach to software engineering; experience within the ASSYST-RAET GROUP

Page

Lewis, Chris.................................... 103
Application building with ARTESSA/3000, description of concepts and facilities

Moreno-Dávila, Julio............................ 109
Taking the myth out of artificial intelligence (Building and using expert systems)

Remillard, Robert............................... 115
Opportunities and dangers of fourth generation languages

Rush, Stephen M................................. 129
How RAPID/3000 can eliminate the "moving target" syndrome

Sharaf Eldin Ahmed.............................. 139
PDS/3000 - a man-machine interface

Vekaria, Hitesh................................. 148
What are expert systems?

BS - Business Systems

Belford, Stephen................................ 160
User-directed application development

Boll, Franz-Josef............................... 167
A thesaurus of information - document storage and retrieval

Drunen, Do van.................................. 177
Hewlett-Packard's internal data processing; a way to office automation

Frijda, Ron..................................... 203
MR - A shop time reporting system on the HP-3000

Harrier, Laymond and Rinesmith, Ralph........... 214
At last - Relief for frustrated repetitive manufacturers

Lawshe, Jim..................................... 232
Integrating order entry with manufacturing How to handle options?

Lawson, Roger................................... 246
Using computers to create brand names

Shroff, Vasant R. and Raghunathan, K............ 254
Funding projects management system in third world countries - an application of HP3000

Page

DB - Data Base Management Systems

Frydenberg, Rolf.................................. 268
Remote data base access - to non-HP computers

Isloor, Dr. Sreekaanth S.......................... 276
4th generation DBMS as the heart of future systems of the automated office

Mund, Ewald Maria................................. 299
Synchronization and recovery of a distributed data-base an implemented example on HP/3000 computer systems

Rego, F. Alfredo.................................. 307
IMAGE reflections

Yang, C.C., Shyh-Ming Chien and Jen-Shyh Lai...... 316
The design and implementation of Chinese data base management system

DC - Data Communications and Networks

Aramoonie, Philip and Scope, Arie................. 336
Designing a network to match your operating environment

Baynton, Ken...................................... 344
Advances in the world of data communications

Buiteweg, Anton J.W............................... 360
From terminal to computer port in 232 easy steps

Driskell, Andree.................................. 377
The evolving micro-to-mini/mainframe interface

Faulkner, Kevin................................... 384
Network services/3000 transport: An inside look

Geesbergen, Rene van.............................. 392
The poor man's DS, fact or fiction

Gresset, Christian................................ 408
Local area network: build up your private X.25 network

Lynn, Brian....................................... 419
Network services for the HP3000

Martina, G.F. and Nicotra, S...................... 428
Interactive local network for HP computers

Page

Williams, David Roy.............................. 435
Travelling light for data communications

OA - Office Automation and Networks

Burch, Marc...................................... 446
Ergonomics of personal computer software in data communication networks

Christman, Wesley S.............................. 453
Business - Management Decision Support Systems - MDSS

Crow, Bill....................................... 479
Integrating series 100 personal computers with the HP3000

Damme, Jacques Van............................... 498
Converting SPL programs from the HP3000 to the HP150

Dummer, David C and Beek, Henny van.............. 516
Growing need for HP3000 -to- micro

Fisher, E.S...................................... 520
The place of the micro in an HP3000 network

Folkins, Dale.................................... 527
Using your HP PC's as HP3000 workstations

Kohon, Michel.................................... 542
Mini/micro merge strategy

Padanyi-Gulyas, David and Benz, Hans Ulrich....... 556
Complex documentation applications with laser-print-output
Implement it just by moulding HP-supplied subsystems for individual needs

Rypma, Ted....................................... 574
New ideas in spreadsheet programs

Setian, Kathy.................................... 581
Personal or powerful: Can we compute both ways?

Vanstappen, Hans................................. 592
An intelligent processor for 'text-creating' users

Wilk, Steve and Boles, Sam....................... 603
Man-machine interface in technical publications simplified with laser printing

Page

Wilson, Paul.................................... 609
Exchanging electronic mail between HPDESK, PROFS and others

OP - Operations Management

Campbell, Ian.................................... 632
Shadow - providing users with 100% application uptime

Day, Barrie W.................................... 641
Automated scheduling of computer operations - A case study

Harris, Kim.................................... 646
Scheduling for better system usage

Kooy, J.................................... 655
Specific services of the Computer Uitwijk Centrum, in particular on contingency service levels

Leight, Betsy.................................... 666
It is time to automate your data center

Parkinson, John.................................... 671
Who needs an operator anyway? - The potential for an automatic HP 3000

SD - System Development

Alblas, Henk.................................... 684
Friendliness standards

DiCecco, Martha.................................... 691
My experience installing application software

Franklin, Bill.................................... 702
Software technology for the 80s (Man <--> Software <--> Computer)

Huysmans, Rudi.................................... 716
Standards for ergonomic interactive programs

Larson, Orland.................................... 731
Application prototyping - A methodology for perfecting the man-machine interface

Marsh, Bill.................................... 744
Guidelines in designing user-oriented software

Page

Mowinski, Julie T. 750
Many men - one machine - the design of international applications packages by adapting the use of the machine to the European market

Ofslager, Nancy 758
VPLUS: Improving the end user interface

Olsen, Roger J. 772
A guide to software evaluation and selection

SM - System Management

Beasly, Dave 786
How dispatching queues really work

Boles, Sam 797
Interfacing the MPE man with the UNIX machine

Dijk, Rob van 813
The poor man's performance measurement

Duncombe, Brian 819
Performance self-analysis

Grim, Jelle 833
The twilight zone.......between MPE capabilities

Grim, Jelle 853
Towards a better utilization of print resources

Heidner, Amy and Heidner, Dennis 873
The world according to GREP

Muntean, Mark and Boles, Sam 887
Structured tuning: Man's interface to the machine's performance components

Primmer, Paul 897
HP LABS systems performance evaluation project

Spitz, Carolyn 905
A file access method for source version management

VS - Various

Engelbreit, Greg 916
Digital optical recording . . . What it is and how it will impact the mass memory hierarchy

Page

Heidner, Dennis and Heidner, Amy.................. 929
Throw away the keyboard

Lehane, John F.................................... 947
Managing information services in the eighties

Simmons, E.R...................................... 957
The nature and needs of human beings

AUTHOR'S INDEX

	Page
A	
Aalders, K.	40
Alblas, H.	684
Aramoonie, P.	336
B	
Bale, J.	59
Baynton, K.	344
Beasley, D.	786
Beek, H. van	516
Belford, S.	160
Benz, H.U.	556
Bishop, S.	53
Boles, S.	603
Boles, S.	797
Boles, S.	887
Boll, F.	167
Buiteweg, A.J.W.	360
Burch, M.	446
C	
Campbell, I.	632
Chien, S.-M.	316
Christman, W.S.	453
Crow, B.	479
D	
Damme, J. van	498
Day, B.W.	641
DiCecco, M.	691
Dijk, R. van	813
Dreyfus, Ph.	3
Driskell, A.	377
Drunen, D. van	177
Dummer, D.C.	516
Duncombe, B.	819
E	
Engelbreit, G.	916
F	
Faulkner, K.	384
Fisher, E.S.	520
Folkins, D.	527
Forster, E.R.	23
Franklin, B.	702
Frijda, R.	203
Frydenberg, R.	268

G	Page
Geesbergen, R. van	392
Gresset, C.	408
Grim, J.	833
Grim, J.	853

H	
Harrier, L.	214
Harris, K.	81
Harris, K.	646
Heidner, A.	873
Heidner, A.	929
Heidner, D.	873
Heidner, D.	929
Herschberg, Prof.dr. I.S.	5
Hudson, Dr. P.T.W.	17
Huysmans, R.	716

I	
Isloor, S.S.	276

J	
Jepson, Ch.W.	21

K	
Kellogg, H.	59
Kemp, L.	74
Kohon, M.	542
Kooy, J.	655

L	Page
Lai, J.-S.	316
Laiho, M.	81
Larson, O.	731
Lawshe, J.	232
Lawson, R.	246
Leeuw, H. van der	87
Lehane, J.F.	947
Leight, B.	666
Lewis, C.	103
Lynn, B.	419

M	
Marsh, B.	744
Martina, G.F.	428
Moreno-Dávila, J.	109
Mowinski, J.T.	750
Mund, E.M.	299
Muntean, M.	887

N	
Nicotra, S.	428

O	
Ofslager, N.	758
Olsen, R.J.	772

P	Page
Padanyi-Gulyas, D.	556
Parkinson, J.	671
Primmer, P.	897

R	
Raghunathan, K.	254
Rego, F.A.	307
Remillard, R.	115
Rinesmith, R.	214
Rush, S.M.	129
Rypma, T.	574

S	
Scope, A.	336
Setian, K.	581
Sharaf Eldin Ahmed	139
Shroff, V.R.	254
Simmons, E.R.	957
Spitz, C.	905

V	
Vanstappen, H.	592
Vekaria, H.	148

W	Page
Wilk, S.	603
Williams, D.R.	435
Wilson, P.	609

Y	
Yang, C.C.	316

INVITED SPEAKERS

Ph. Dreyfus

"The need for future 5th generation hard/software?"

BIOGRAPHY

Mr. Philippe Dreyfus is Vice Chairman of the world-wide CAP Gemini Sogeti Group, where he was chairman of CAP Europe since 1968.

After completion of his studies in chemistry and physics in Europe and at Harvard University in 1951, he was involved in the design and marketing of the Gamma 60 computer of Compagnie des Machines Bull.

He introduced the first Control Data computer in Europe and was VP of European Development of that concern. Mr. Dreyfus is consulted frequently by the French government and he organized for president Giscard D'Estaing the week of "Information and Society". He invented the conception and the French word "Informatique".

Philippe Dreyfus is Chairman of "Syntec Informatique" and Fellow of the British Computer Society, ex member of the council of Association for Computing Machinery A.C.M. in USA and of SICOB, France.

ABSTRACT

Ph. Dreyfus

"The need for future 5th generation hard/software?"

From the early 60's through the 80's Information Processing was constructed around main frame computers operating in a closed-shop environment.
During these 25 years huge investments have been made in operational software built around first generation high-level languages and even assembler code.
The staff DP Manager, analysts and programmers have been trained and have aquired great skills working with these systems.
The need for change in these well proven techniques does not originate from this environment but from demands from end-users.
Their needs can only be satisfied if they do not have to depend from that former type of EDP organization.
Hence the outburst of professional micro computers and of fourth generation software.

However, two big questions remain:
Does an end-user accept the disciplines of programming even with new languages?
Can he operate without accessing corporate data, stored within the traditional structure?

The answers to both these questions will be discussed.

Prof.dr. I.S. Herschberg

"Malice in Bitland"

BIOGRAPHY

Prof. Herschberg started his career as a chemist with a strong theoretical bent. In 1955, this led him into a computer treatment of a problem in theoretical organic chemistry. When he found that about a day's work on an electrical calculator could be performed in a minute on the ARMAC, one of the first Dutch home-grown computers, he never looked back on chemistry.

After serving three major Dutch-based internationals, he took his present post with Delft University of Technology, where he teaches and researches in operating systems, emphasizing their security and privacy aspects, or rather, the lack thereof. This has naturally led him into various advisory functions in which he advocates enhancing security, by adversary process if need be.

MALICE IN BITLAND

by

Prof. dr. I.S. Herschberg[1] and R. Paans MSc[2]

Computer-system security is penetrable to the clever. Some causes of this sad fact are presented, the historical absence of security-consciousness being first among them. A few instances, representing an anthology from the authors' portfolio of documented cases, will substantiate our main assertion, viz. that no system is proof against a really astute programmer. It follows that programmers should be excluded from systems processing reliable data.

INTRODUCTION

The clever man has always had a malicious trait in him. The first familiar example in history must be Jacob. He had an agreement with Laban: whatever lambs were speckled and spotted would fall to Jacob, whichever progeny of the flock were white lambs would be for Laban, his innocent partner.

So Jacob, certainly a clever man and, in our view, a programmer *avant la lettre*, saw to it that whenever the best animals mated, they had fresh, half-peeled rods in front of them, in consequence of which they produced speckled and spotted offspring to this prime programmer's advantage. The 30th chapter of Genesis has all the details about this extremely clever programmer manipulating the system.

Many present-day programmers have inherited Jacob's propensities. They can, and will, program against the system and, in doing so, will wreak havoc to security in proportion to their quality as programmers. A crack programmer implicitly is a cracksman.

We shall analyse the causes that enable them to do so and hope to frighten our readers by citing a few cases in relevant detail.

[1] Delft University of Technology, Department of Mathematics and Informatics, Julianalaan 134, 2600 GA Delft, The Netherlands.
[2] When researching for this paper, Mr. Paans was with Delft University of Technology, Department of Electrical Engineering.

THE BURDEN OF HISTORY

To some it may seem incredible that present-day systems are vulnerable and easily penetrated by any Jacob. The penetrations in themselves have been amply documented: though many are collected by EDP auditors, only a few have been described in open literature [5]. The authors' collection of known breaches numbers some fifty cases and is still growing by the month. However, these fifty cases seem to be only the top of the iceberg.

One cause at the root of this vulnerability is the burden of historical development shouldered by our current systems. To quote but one instance: System/360 will reach its twenty-first birthday this year [1]. As a self-imposed constraint, the manufacturer has undertaken to make sure that programs valid for the very first models of System/360 will still be textually correct and have the same effect on his very latest models (43XX and 308X Series). Yet, in 1963, the notion of security, especially against the malicious programmer, had not even, so to speak, been invented. Those old enough to recall early computing days will remember that the system then was programmed and operated by a small dedicated group, who literally slaved day and night to persuade it to do its job. In this tightly-knit group, there was a great deal of social control and neither time nor inclination to pervert the instrument. Also, access to the facilities was only by physical entry to the computer room, again making for tight social control.

Since then, access has spread geographically: in-house terminals came into being, followed by remote stations, capable, in some cases, of free programming in time-sharing mode or its equivalent. This development in itself makes for vulnerability, especially when one considers that social control is not only absent but also impossible in such a large and scattered population of users. Yet many systems, such as IBM's, remained compatible with their predecessors which were not bothered by a security problem. It follows logically that whatever security is now built into our systems has been tagged on, almost as an afterthought. It is symptomatic in this respect that even a leading manufacturer has not integrated security into his OS (which is by no means defenseless), but provides it as an optional, additional package available for an extra fee.

Not to mince words: security has been tagged on. It is comparable to the kitchen article that goes by the name of a strainer when required to be converted into a pressure cooker. Soldering shut the holes one by one seems a poor procedure for obtaining an air-tight vessel. Indeed, the metaphor is too optimistic: the holes in a strainer have a very regular pattern, while the potential breaches in security seem to have no pattern whatsoever.

In tagged-on security another, psychological element may contribute to its weakness. The supplier and his programmers are locked into a mode of thought which tends to persuade them that their security measures render the system secure. This mode of thought tends to made them blind to any flaws still remaining. This blindness is strictly comparable to chess blindness. Weinberg [8] has wise remarks on cognitive dissonance in programmers; they apply equally well to a supplier's security experts.

HACKERS vs. BACKERS: AN ADVERSARY PROCESS

The above cause of penetrability is worthwhile restating: many present-day vulnerable systems have not been constructed with security as one of their primary design goals.

It is characteristic that one of the systems in which security ***was*** a major and even an overriding design criterion has turned out, in our hands, to be proof against major breaches. Not only was security paramount in the design of GE's MARK III, but it was also supported by a procedure in which expert programmers were charged with the task of cracking the system. Breaches discovered by the cracksmen were then given to others to repair. In brief, the system evolved to a fair degree of security by an ***adversary process***, avoiding the cognitive dissonance that must arise when cracksmen and constructors belong to the same team. It is equally characteristic that even in this system a malicious programmer could still do some mischief denying part of a machine to its users, though only transiently.

While still retaining our assertion that any programmer worth his salt can program almost any system to pieces, it is at least encouraging to note that the USA Department of Defense [3] prescribes security testing by adversary process for systems to be evaluated to the Department's criteria. All systems should be considered totally penetrable unless, to coin a phrase, its backers have been shot at by hackers.

SQUATTERS' RIGHTS?

Another cause of penetrability is the glaring co-residence of system data and user data in what might be termed a single compartment. In our view, co-residence amounts to committing a sin. Sinful instances are: catalogues will be found to reside on the very volumes they describe, highly sensitive programs with unlimited access (system utilities) will be found running in conjunction with user programs intended to have strictly limited access, all-important control-block fields indicating 'access should be verified' co-reside with low-privileged user programs.

Co-residence, sinful in itself, is heightened to the level of mortal sin when, as in most installations, users having very limited privileges co-reside with system owners of unlimited privileges. It is commonly assumed that the latter must have all privileges. They normally delegate all of these privileges to their technical subordinates, such as system programmers and, to a lesser degree, to data-base administrators, network operators and the like. The sin is mortal because it is often found that expert programmers, starting with lowly privileges, can gradually acquire all the privileges pertaining to those happy few delegates of the system owner.
More compactly: squatters can assume owner's rights.

To restate this cause: a poor separation of residence is equivalent to a poor separation of functions. Now the principle of separation of functions and duties is well known to be an absolute requirement for the auditability of an enterprise. Reasoning by analogy, a separation of residence between system data and user data would equally seem to be a prime requirement for the auditability of a system's security.

We remark in passing that the smaller systems (typically minis and micros) have poor separation of residence. This exactly mirrors the poor separation of duties often found in the smaller type of enterprise.

FRIENDLY IS TOO FRIENDLY

There is yet another cause for the easy penetrability of computer systems. The cause is fairly subtle and arises from the need of any supplier effectively to market his system. It should be noted that this marketing need not be a commercial operation: an in-house supplier of services to a company or to a scientific institute also must market his wares. Should he fail to do so, he is threatened with loss of prestige, power and, ultimately, with the loss of his job. Now this need of marketing implies ease of access to a user; the marketing is often in the shape of providing 'user-friendly' systems. Security requirements, in the present state of the art, greatly detract from ease of access and from other characteristics considered to be user-friendly.

Hence, we have a dissonance: if the marketeer is to enforce security, he is hampered in his marketing efforts. Experience then shows that the balance is always tilted against security and in favour of perceived friendliness.

Again, this cause may be restated in slightly different terms: when a balance must be struck between the requirements of security and those of operational speed, experience shows that speed of operation is almost invariably opted for, greatly to the detriment of security. Computer-centre management are on record to have refused the installation of security packages because, reportedly, these would increase overhead by a few percent [6].

THE 5000-YEAR JIGSAW-PUZZLE

Current systems are hypercomplex. Their very size is terrifying: ten million statements are seen to be too complex to be manageable when it is known that a good programmer creates only 2,000 statements a year. Five thousand man years of intellectual labour is too large to be surveyed by any human being.

As a consequence, the system contains flaws both in concept and in execution or, to be more precise, errors in logic as well as errors in coding. A constant flux of corrections to operating systems is a fact of life and users are lucky when they see their complaints responded to: cases are known where documentation reflecting the design has been retrofitted to meet incorrigible errors...

It follows that those responsible for the system's long-term welfare have a fairly continuous job of installing such corrections as are published, modifying the system at regular intervals. To this end, operator intervention is often needed. The impact on security is that the operator, when confronted with such a request, is forgiven for thinking that 'those system programmers are it again' and therefore consents readily to requests for changes which may well stem from the malicious.

To make matters worse, there is a current tendency to disintegrate whatever poor structure may have been originally present in operating systems. It has recently shown that, in the interest of speed, a major operating system has resorted to programming tricks such as undisciplined branching, modifying links and the like which, while they may speed up operation, are a bane for maintainability and further complicate a complexity already known to be unmanageable.

WHO SHOULD HAVE DONE IT?

On top of everything, there is a great deal of confusion in the allocation of responsibilities for the system's correctness, maintenance and security. Operators are often being given responsibilities which should not be theirs, system programmers, knowing all, are conventionally permitted all. As a result, in many installations nobody is sure whether it is his duty to install and update or whether it is his fellow's.

The result is *confusion*, usually with everybody being allowed sensitive accesses, but nobody being in charge of them. This greatly contributes to the many opportunities all too often given to the malicious. Needless to say, this also greatly detracts from such security as can be achieved.

MODERN INSTANCES

So far, we have presented an abstract case for penetrability without exhibiting penetrations. We think the reader is entitled to some actual instances of systems having been breached. Three of them will follow, but we stress again that these three are no more than an anthology from our well-stocked collection [5]. The cases to be presented have been selected because they reflect the causes. They possess a number of common traits:

- the knowledge needed for penetration was either in the public domain (published in manuals or in the scarce and scattered open literature) or was deduced from facts in the public domain,

- the breaches were made by persons having at least some limited privilege to program freely,

- the breaches were effected by pure programming, which is to say that they did not rely on human carelessness (e.g. broadcasting passwords),

- the programming involved admittedly was often less than straightforward in nature and in all cases relied on a knowledge of machine code and other suitable means of access at the bit level: here, too, a crack programmer is at an advantage as a cracksman.

Our experience has shown that almost every system, regardless of its supplier, is extremely vulnerable to this type of breach. It should also be noted that none of the systems quoted were defenseless: all embraced some degree of access control.

CASE 1 : EAVESDROPPING

Our first instance deals with a time-sharing system. It may be identified, viz. as the XEROX SIGMA-5, a 32-bit processor now obsolete. It was run under that supplier's standard operating system and did not enjoy the benefit of a front-end processor. Because of this, any I/O data to or from terminals must necessarily be present in main memory. An attempt to breach this system's security therefore could well start by searching all of memory for printable strings, the assumption being that (almost) all matter destined for or originating from a terminal would consist of printable character strings.

In those bygone days, there normally were some 64 printables, each stored in an 8-bit byte allowing 256 patterns. Strings can efficiently be searched for because the chance of a fullword (4 bytes) appearing to a printable string without being it is only (64/256)**4 or 1 in 256. Hence, a hunting program was written to search for presumably printable strings of length≥4 characters, printing these, when found, with their memory locations. It was hoped that the terminal-I/O buffers, presumed to be memory-resident, would thus be found. To our surprise, this yielded all system-error messages (e.g. 'SQUARE ROOT ARGUMENT ERROR'), but no terminal-I/O data.

After some experimentation, it became clear why: the program hunted for EBCDIC strings while terminal I/O used ASCII. This was easily remedied, after which these buffers were found to occupy a contiguous area of 32 by 80 bytes, amounting to one punch-card-sized buffer for each of the 32 terminal-I/O-ports. Watching the activity of these buffers soon revealed that each buffer had an associated variable with three possible values: 0 corresponding to an inactive port, and 1 to one actively engaged in I/O. Strangely, the value -1 also occurred. A little reflection made it likely that such a port would be engaged in something out of the ordinary. The hypothesis was formulated that I/O during log-in, when the system must be aware of the I/O activity but cannot, as yet, know which user to charge for it, corresponded to the value -1. This hypothesis was soon corroborated.

Once this knowledge was acquired, watching the I/O buffers for significant information was much simplified. Our program scanned the associated variables; if none was found to be -1, the program went to sleep for 10 seconds. The contents of each buffer having its variable at -1 was printed, repeatedly if necessary.

As follows logically, this yielded all userids and passwords of users logging into the system during the program's activity. It must be admitted that the program failed to obtain *all* userid/password combinations. As against this, it is seen that the program is more likely to catch a user in proportion to that user's activity. In a few days' cheap operation (our program was asleep almost always), the vast majority of users were captured. It might be noted that, once a single password was detected, the program might have been run at somebody else's expense, making the victim pay for having the privilege of having his files accessed, inspected or even modified to the point of deletion.

The breach was reported to the supplier who, in this case, reacted fairly promptly. The countermeasure installed was a twofold scrambling of the terminal-I/O buffers. First, each symbol was encrypted while, second, the 80 positions of each port were no longer contiguous, but randomized over the entire 32*80-bytes buffer area. Moreover, the scrambling algorithm, we were told, contained two variable instructions which were changed weekly.

The countermeasure proved completely ineffective. The FORTRAN provided allowed us to write a fairly efficient disassembler, reducing the resident machine code of the operating system to an easily readable form. In this form, it was not difficult to identify the terminal-I/O driver in the operating system. It had the form of a routine requiring two integer parameters, *i* and *j*, and, when called, yielded the decrypted value of the *i*-th symbol in the *j*-th buffer.

Consequently, our next program copied this subroutine into its own program area (to take care of the variable instructions and to avoid interference possibly due to concurrent use) and simply called this driver instead of reading out the buffers directly. It should be clear that, with this modification, our program again fully achieved its purpose, viz. capturing userids and passwords. As a curious footnote we remark that we never understood the encryption/scrambling algorithm, nor did we have to...

CASE 2 : HUSH, I TELL YOU

Our next instance refers to a system very much alive, currently in daily use in several tens of thousands of major installations. In hardware, the system has a fairly large number of I/O processors, operating at the main processor's request. Once such a request ('I/O program') has been given to the I/O processor, the latter operates independently. In these systems, any user program may offer an I/O program to the operating system (OS). The latter then scans it for apparent validity, may transform its main memory's addresses etc.; in due course, the OS then transfers it to the I/O processor for execution.

A peculiar kind of breach may now be forced by any programmer who constructs an I/O program containing an infinite loop (though not a conspicuous one, which the OS will reject). When this is offered to the I/O processor, that processor and any I/O devices, such as disks, attached to it are effectively disabled. Moreover, there is nothing to stop the malicious programmer from offering such an I/O program to all I/O processors in turn, bringing the system to a complete standstill.

It has been found experimentally that the action of the malicious program is very hard to detect for what it is. The OS has been found, moreover, to have no means for undoing the damage short of reinitializing the entire system, which is, to make matters worse, not an orderly reinitialization. Hence, much of the work in progress (e.g. edit sessions) will be lost in some unknown state of processing.

Thus, in summary, any malicious programmer is in a position at least temporarily to deny the entire system to all other users, including the system's owner.

When first describing this case, we had one supplier in mind. It has since been brought to our attention that at least one other supplier felt he had been referred to. On investigation, it turned out he had.

CASE 3 : THE OVERBURDENED OPERATOR

Our last instance covers a composite of sins to be found in almost all major computing centres. They all rest on an assumption which, while pernicious, is very much part of computer-centre practice though it runs counter to recommended procedure [4]. Whatever the guise it appears in, it boils down to giving the operator ultimate authority. Often, it is the operator who decides whether or not to allow:

- a modification of the system's kernel,
- overriding an expiration date 'protecting' information,
- jobs to access information protected by a password to be supplied by the operator.

Let us briefly analyse how sins have been heaped on one another in this construct:

- the operator should be charged with the minute-to-minute welfare of the system: matters of security are ***not*** within the scope of his brief, nor should he be charged with them,
- passwords are also without ***his*** scope. Unfortunately, for historical reasons, "...the operator must respond with the password before a data set can be opened...", as Brown [2] recorded in the early days of the System/360,
- nor is this practice imposed by the supplier: whenever the operator should respond, the appropriate command (Write To Operator with Reply) can be given a route-code. When this code equals 9, the message should reach the security console with a presumed security officer in charge, who can then express his approval. However, as matters are usually arranged, this scheme fails for either or both of two reasons:
  - there is no security officer; as a substitute, though a poor one in our view, the operator is given that officer's authority,
  - it is thought too expensive to assign a distinct console to security, so the security console is made to coincide with the operator's.

For whatever reason, the nett effect is that the operator is inclined to allow whatever is requested. Moreover, the security-sensitive messages, some tens a day, are swamped by the tens of thousands other messages passing the screen.
It follows that the consenting operator unwittingly allows at least the following breaches to be perpetrated:

- modifications to the system's kernel; the simplest of these is that a malicious programmer installs his own entry into system mode which, when installed, will allow him all privileges [5],
- the apparently harmless overriding of an expiration date allows the malicious, amongst others, to change or add modules to authorized system libraries, granting themselves any and all powers for good or evil,

- accesses to any sensitive data set, such as that containing all user attributes or passwords, are habitually 'protected' by passwords to be supplied by operators (in batch; time-sharing users are beyond the operator's control) who promptly supply them.

Quoting Brown [2] again, we find that it was known, even in 1970, that "...password protection is adequate for non-critical data sets but may not be appropriate for highly sensitive data sets such as payroll". In spite of this early warning, password 'protection' is the rule rather than the exception for data sets recording passwords (in clear!) which are, in our view, infinitely more sensitive than mere payrolls.

THE ETHICS OF THE ADVERSARY PROCESS

Inevitably, publications of breaches lead to the question of whether penetration in itself is ethical and whether one should publish any found. We have stated our case before: an adversary process is, in the present state of the art, probably the best and possibly the only means of improving security [3]. When the supplier fails to employ such a process, the authors believe it is ***their*** duty to act as adversaries, accepting the obligation to report their findings to the suppliers. Unfortunately, - our first instance provides a hint - suppliers act inadequately and, let it be added, often sluggishly.

Our position is that publishing is ethical, it being the only means of persuading suppliers to enhance security, though publications should be couched in discreet language so as not to provide a hacker's handbook. Granting this, suppressing attempts at penetration, neglecting to report them or failing to publish them unless remedied in our view comes close to being unethical.

Our view, though, is not shared by one and all. During the process of persuading the supplier to take corrective action, the penetrator is subject to a specific form of harassment and slander: "We [the suppliers and users] are honest and decent people, who should not be worried by reports about disreputable matters such as breaches in security. Mr. X, who found the breach, must have a peculiar cast of mind to be so fascinated by the subject. In effect, there is something unhealthy about Mr. X's interest. Isn't Mr. X a bit of a suspect character himself? After all, gentlemen do not read letters not addressed to them..."
We have found it necessary to steel the many students we have taught and coached against this attitude, which is all too likely to confront them [5].

CONSEQUENCES

By way of causes and instances, we have, we believe, pointed out the astute programmer as a major threat to any system entitled to regard its output as relatively reliable. It follows that programmers should be excluded from production systems, as should their potentially threatening partners, job schedulers and operators.

A companion paper [7] points out the consequences of our findings: in data processing, day-to-day production and software-development systems must be completely and inpenetrably separated, both logically and physically; an unbreachable barrier between them should operate as a guarantor of this separation.

ACKNOWLEDGEMENTS

The authors wish to record their indebtedness to the many persons with whom they have exchanged opinions on computer-security issues. They count themselves fortunate in that their discussion partners have come from all ranks of life, ranging from security managers of major installations and their systems programmers to perpetrators of breaches at the student level. Most of our partners have expressed the wish to remain anonymous, which wish we shall somewhat reluctantly respect.

REFERENCES

1 G.A. Blaauw and F.P. Brooks, Jr., "The structure of System/360, Part I - Outline of the logical structure", IBM Systems Journal, Vol. 3, No. 2 (1964), pp. 119-135.

2 G.D. Brown, "System/360 Job Control Language", John Wiley & Sons, SBN 471-10870-7 (1970) p. 237.

3 DoD, "Trusted computer system evaluation criteria", U.S. Department of Defense, document CSC-STD-001-83, Library No. S225,711 (Aug 15, 1983).

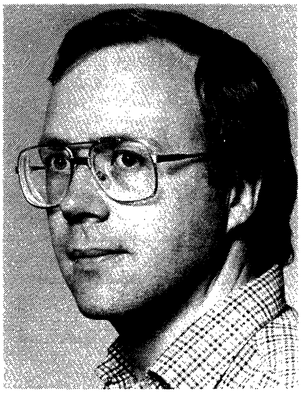
4 IBM, "42 Suggestions for improving security in data processing operations", IBM order number G520-2797.

5 R. Paans and A.H.J. Bonnes, "Surreptitious security violation in the MVS operating system", Computers & Security, Vol. 2, No. 2 (June 1983), pp. 144-152.

6 R. Paans, "Performance aspects of supervisor call services in MVS systems", ECOMA-11 Conference Proceedings, Copenhagen (Oct 1983), pp. 93-105.

7 R. Paans and I.S. Herschberg, "A topology for secure MVS systems", in "Computer security: a global challenge" Proceedings of IFIP/Sec'84 Conference, J H Finch and E G Dougall (eds) (Sept 10-12, 1984) Toronto, ISBN 0-444-87618-9 North-Holland Publ, pp. 137-147.

8 G.M. Weinberg, "The psychology of computer programming", Computer Science Series, Van Nostrand Reinhold Co., ISBN 0-442-29264-3 (1981), pp. 54 sqq.



Dr. P.T.W. Hudson

"Artificial Intelligence and the cognitive approach to software: The programmer's apprentice"

BIOGRAPHY

Dr. Patrick Hudson is a British scientist currently working at the Institute for Perception-TNO. He first met computers in 1966 working for a company building flight simulators.

As a psychology student in Edinburgh he came into contact with Artificial Intelligence in 1968. His 1976 thesis at the University of St. Andrews was about computational models of brain function.

In 1975 he came to the University of Amsterdam to lecture in the department of Linguistics and then became Visiting Professor there in "Artificial Intelligence Approaches to Natural Language" in 1976.

At present Dr. Hudson is working on problems of Computer-Human Interaction and developing a number of Expert Systems. One is for an expert in Printed Circuit Board design for CAD/CAM, the other an expert to help airborne Search and Rescue missions. He is a member of the editorial board of the journal "Future Generations Computer Systems" and secretary of the NATO research group on "Computer-Human Interaction in Command and Control".

ABSTRACT

Artificial Intelligence and the Cognitive Approach to Software: The Programmer's Apprentice.

Patrick Hudson.

Institute for Perception TNO,
Soesterberg,
The Netherlands.

Software costs already form the major part of any computer system. High development and maintenance costs are a function of the increasing complexity of modern systems. Many approaches have been developed to try and cope with these problems, such as rapid prototyping and program generators. These are all essentially approaches to software which allow minor variations on a theme. What, however, if we want to develop entirely new software? What if we don't know exactly what we want? The chances are that whoever writes the software will not maintain it and vice-versa; whoever defines the needs will not be a sufficiently good programmer to produce it; whoever writes the program won't know what is really important.

Central to many of these problems is the idea that programming a computer is pretty difficult, they are dumb beasts and need to be driven all the way to a solution. But this easily leads to a situation where no one actually knows what is going on as each specialist in the chain may develop his or her own interpretation of the real goals of a system. One solution, which can certainly help keep the astronomical costs such proliferation can incur, is to use off-the-shelf components. This forces the defined or perceived problem into a standard mould, followed by the late tailoring of the result. This is equivalent to casting something in metal and then filing it by hand until it fits. While we may be satisfied because there is something which goes in the hole and works a bit, this is far from ideal. It is all too reminiscent of the drunk looking for the house-keys under the lamp post, they were lost somewhere else but it was easier to look for them under the light.

What we need is something which enables those who define the needs to define their computer programs as well, even though they are not wizard hackers. In order to do this we need a number of features in the environment we provide.

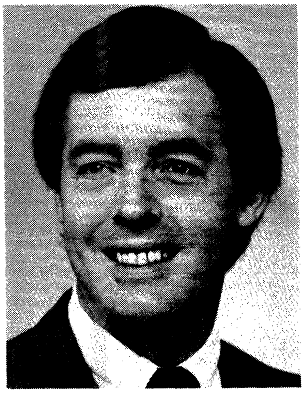
1) The system should be capable of creating software in an environment made to fit the individual who is controlling the computer, and not the reverse;

2) The user should be provided with an environment in which it is possible to try out alternatives, see their consequences, and be made aware of possibilities other than those currently considered; and

3) The system should not only write its own documentation, what it should do is create its own, useable, maintenance environment.

This defines, essentially, an intelligent programming environment which attempts to understand the programmer and is therefore capable of interfacing at the functional level. The general notion which this covers is sometimes seen as the provision of a reasonably expert programmer at the system designer's elbow, a sort of Programmer's apprentice, who can do the tedious bits. Two elements are crucial to this understanding. The first is that we must be able to provide an environment which operates at the human level, in terms of the concepts of the user, hence the term This is a natural extension of the line from machine code, to symbolic assemblers to so-called higher level languages. But, contrary to the idea that a good computer language is one which is strictly standardised, we have the idea developed in Artificial Intelligence, of a computer language which alters to fit the problem. The second element is that such an environment be capable of entertaining alternatives, different points of view, so that the system becomes more and not less flexible.

This talk will discuss some of the ideas here in more detail, attempting to lay out what is necessary to create a programmer's apprentice and discussing how that might work in practice. Central to this are the notion of different models, of the user, the task, the dialogues and of the possibilities which the computer offers. A good system will be one which supports those models which the user also has in their head. All in all the programmer's apprentice will have to act as an interactive aid to the writing of software. It will 'understand' why the software is written the way it is and will attempt to use its knowledge of other systems to suggest code (or steal it) and also to suggest refinements to the task description. If such a system does nothing else it may make it less likely that perfect code is produced to one specification, but the specification has changed in the meantime. The creation of such systems will represent a further step on the way from systems which are "What You See is What You Get" to "What You Want is What You Get". They will also mean that what you get is what you deserve, because they are intended to support the most important process of defining exactly what it was that was wanted and, only then, writing code for that.



Ch.W. Jepson

"HP Overview"

BIOGRAPHY

Charles W. (Chuck) Jepson is Marketing Manager of the Informations Systems Group where he is responsible for the marketing of Hewlett-Packard's office automation products including HP3000 business computers, HP250 small business computers, office automation software, administrative productivity applications and programmer productivity tools.

Chuck formerly was Marketing Manager of the Information Products Group in Boise, Idaho, where he was responsible for HP's Advancenet networking products. Prior to this, he was Marketing Manager for HP's Boise Division which makes computer printers and magnetic tape storage units, and prior to that he was Division Controller responsible for facility information systems.

Chuck joined HP in 1971 after receiving an MBA from the University of California at Berkeley. He has an undergraduate degree in economics from San Jose State University in San Jose, California. Prior to joining HP, Chuck held positions with Singer Co., and IBM.

Chuck, his wife Judy and their two daughters reside in Saratoga, California after moving from Boise, Idaho in June of 1984. He enjoys snow skiing, golf and football.

ABSTRACT

Ch.W. Jepson

"HP Overview"

Chuck Jepson will provide the conference attendee with an update on the general direction in which HP is moving. He will review HP's current financial picture and comment on the outlook for the remainder of 1985. In addition, he will recap HP's reorganization and the effect it will have on the customer.

Chuck will present an overview of HP's computer strategy, placing special emphasis on the Office Strategy. He will discuss HP's current hardware and software product offerings and provide insight on future product direction.

ELIZABETH RICHARDS FORSTER

Education

BRYN MAWR COLLEGE, Bryn Mawr, PA. B.A. Economics. May 1985. Computer Science Minor.

UNIVERSITY OF PENNSYLVANIA, Philadelphia, PA. 1980-1981.

THE BALDWIN SCHOOL, Bryn Mawr, PA. Graduated 1980.

Qualifications

* PROGRAMMING & COMPUTER SKILLS: Proficient in utilizing PASCAL and FORTRAN, experience writing BASIC and C.

* COMMUNICATIONS EXPERIENCE: Instructing students in computer programming and systems operations.

* INVESTIGATIVE AND RESEARCH ABILITY: Collecting and interpreting data in scientific laboratories.

Experience

DEPARTMENTAL ASSISTANT: Bryn Mawr College, Bryn Mawr, PA. Teach and assist students in "Introduction to Computing: Pascal" and "Introduction to Computing: Fortran"; grade homework and examination problems. School Year 1984 - 1985.

PROGRAMMER, FUNDED BY IBM SCHOLARSHIP: Bryn Mawr College, Bryn Mawr, PA. Created a software project introducing a new method for recording and processing questionnaire results. Summer 1984.

OPERATOR/CONSULTANT: Bryn Mawr College Office of Computing Services, Bryn Mawr, PA. Supervise the operation of Hewlett Packard HP3000 computer system and answer questions on programming and operations. Fall 1983 - present.

LAB TECHNICIAN: Columbia School of Physicians and Surgeons, New York, NY. Examined effect of cigarette smoking on premature infants. Summer 1983.

SALESPERSON AND BOOKKEEPER: Wicker Imports, King of Prussia, PA. Balanced books. Organized, evaluated, and updated inventory. Summer 1982.

LAB TECHNICIAN: Children's Hospital of Philadelphia, Philadelphia, PA. Conducted experiments to find cures for cystic fibrosis. Summer 1981.

Activites

Yearbook Committee, Drama Club, President of Senior Class and Captain of Varsity Volleyball Team at the Baldwin School.

"The Ultimate Challenge - Perfecting the Man - Machine Interface"

It is raining, it's Monday, and you feel terrible. You enter your office and on the top of your desk is a stack of papers over six inches high. On the top of the stack is a note from your boss saying, 'Record all the responses to these questionaires and hand me a list of all the data by six p.m. tonight.' You feel like someone just hit you with a ton of bricks. How could all of this data be reported and organized efficiently in eight hours? What are you going to do? Read on...

Questionnaires, surveys, and other paper instruments all contain data that must be entered into a computer . Data entry is the term that describes the task of typing numbers and names into a computer. Although data entry may seem to be a simple but tedious task, previous approaches have produced many problems.

One method by which an individual can enter data is by simply using an editor. One can directly type in the applicable numbers and names and create a data file. This technique is error-prone, because the user must meet the needs of the computer. In addition to exhibiting perfect typing ability, the typist must correctly place each piece of information in the appropriate columns or fields of each record of data.

Another method of data entry is the utilization of menus. A menu is a template for data which appears on the screen. The user enters the data by typing the numbers and names in the designated places, that is by filling in the blanks. Getting to these particular places is an ordeal because the tab key is used as a spacing device and backspacing is awkward. What results is excessive movement within and among menus. Menus are organized in some specified order by the programmer and have to be accessed sequentially. One is unable to look at one menu and then another way down the line without observing all of them. This process is both time-consuming and irritating. Lastly, the menu shows no physical similarity to a questionnaire or survey form so that the user is confused in transcribing his data from paper to screen.

The graphics tablet composed of a stylus which is a pen-like instrument, and the tablet which is a flat, reactangular surface, sensitive to the touches of the stylus, exhibits advantages over the older methods of data entry. One is able to place the paper instrument on top of the graphics tablet and point with the stylus to specified locations written on the questionnaire or survey. This allows one to utilize the same form of a paper device as was used in the original recording process. This means that the individual can design a template that acts as the original guide to his paper instrument and then record all the data from the results using this device. Second, the stylus and the tablet are nearly the same instruments as the pencil and paper. The physical similarities between these two utensils create a comfortable atmosphere for the user. Taking this idea one step farther, the stylus and tablet represent an "electronic clipboard". The computer is made to look like the user's familiar tool, rather than the user being forced to make his work look like the computer.

Perfecting man-machine interface requires that the machine be more like a person. Typing on a keyboard is not a normal function of the human body but using a pencil and paper is. This idea of accomodating man's operations is the backbone of the interaction between man and machine.

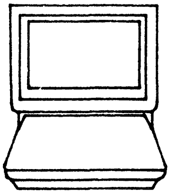
What begins to happen is that machines are made to look and act more like human instruments. An example of this approach is that the graphics tablet can be utilized for data entry using the stylus and the tablet. As discussed earlier, the individual can place the paper instrument on the tablet and poke the desired locations. Program control can be maintained using function keys located on the keyboard or softkeys, keys which are found on the tablet itself. When touched by the stylus, softkeys are activated and functions previously defined go into operation. Utilization of function keys requires that the user holds the stylus in one hand while the other hand uses the keyboard. The choice of hands depends on the user's own dexterity preferences. This enables the user to feel the comfort of a pencil and paper atmosphere but yet must frequently interact with the keyboard. Softkeys, on the tablet, give the user more program control. They allow the user to do almost all of his work on the tablet with little interaction with the keyboard.

It is also an important factor that the user feel that the machine is a guide trying to help, not dictate. In order to accomplish this, the computer speaks to him through dialogue on the screen. This dialogue tells the user what is going on, what choices the user is allowed to make, and how he is supposed to react at any moment. The dialogue also can include general explanations of the procedures and helpful hints for the user. If, in addition, the tablet softkeys are being utilized, overlays are placed over them that display the designated function for each key. This enables the user to see all the possible choices directly on the tablet without need to look at the screen for help. The operations of function keys can be displayed on the bottom of the screen. Tablets with softkeys illustrate a more perfected man-machine interface . Dialogue and overlays are both efforts on the programmer's part to be user friendly.

User friendliness can, however, be emphasized to a degree such that the importance of programming is lessened. When this happens, the program and it's algortithms begin to be awkward and hard to maintain for other programmers. This choice between user friendliness and programming is one that confronts many programmers.

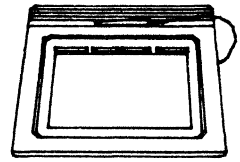
Utilizing the tablet for the purpose of data entry has disadvantages. The one discussed earlier is the addition of another instrument, the tablet. The user must be able to switch back and forth between tablet and keyboard. With softkeys, the switching will be considerably less. Allowing the user to totally interact with the tablet is not possible at present because hardware is not so far advanced. The use of any system still forces some constraints on the user. One restriction with the use of a tablet is the "shape" of the paper instrument. A unit of it can not be bigger than the size of the tablet. In addition, it must be placed in the exact same position each time (lengthwise or widthwise) or else the stylus touches will not be consistent. Another restriction caused by using the tablet is that some responses must be recorded through the keyboard. To solve this problem it is possible to collect all keyboard responses together at begining or end of questionnaire.

...The answer is an easy one. You pull out your graphics tablet from under your desk where you have been hiding it from your fellow employees and begin using the software package,QUESTIONNAIRE 3000, created at Bryn Mawr College for data entry. It will only take a couple hours and you can even treat yourself to an ice cream sundae at lunch!!



1. Editor -
   error-prone

2. Menus-
   time-consuming
   physically dissimilar to
   paper device



1. Physically similar to
   paper device

2. Acts as template -
   "Electronic Clipboard"

3. Program Control

4. Dialogue & Overlays =
   user friendly

Restrictions -

1. Shape of paper

2. Requires exact same
   position each time

3. Keyboard Responses

HP GRAPHICS TABLETS

17623A	*9111A*
2623, 2627	2647, 2648
HPDRAW HPMAP	IDSCHAR

QUESTIONNAIRE

QUESTIONNAIRE/3000

Paper instrument

questionnaire
survey
lab records

Develop "setup"

question types
number of responses
position of responses

variable names, labels
value labels

Enter data

question-by-question
page-by-page

"flat" ASCII file

SPSS control "cards"

QUESTIONNAIRE/3000

Paper Instrument

acceptable to either tablet
prefer "landscape" rather than "portrait"
64 questions – 32 responses
collect keyboard entries together

Questions

1. One choice
2. Choose all that apply
3. Ranked list
4. Continuous scale
5. Keyboard entry

Setup

Review any question
Graphic image of question
Function key or menu square

Entry

Review any question
Review any questionnaire
Resolution – typed symbol – 0.1"
Function key or menu square

1. Vote for

- ○ Reagan
- ○ Carter
- ○ Anderson

2. Valuable experience

- ○ Congress
- ○ Governorship
- ○ Private business
- ○ Armed services

3. Issues

- ○ lower inflation
- ○ lower unemployment
- ○ lower taxes
- ○ stronger defense

4. Doing a good job

disagree ________________________________ agree

5. Comment

1. Vote for	(x) Reagan ( ) Carter ( ) Anderson
2. Valuable experience	(x) Congress ( ) Governorship (x) Private business ( ) Armed services
3. Issues	2 lower inflation 1 lower unemployment 3 lower taxes 4 stronger defense
4. Doing a good job	disagree ——— x ——— agree
5. Comment	No comment

Use of QUESTIONNAIRE/3000

SETUP phase

1. Vote for	○ Reagan ○ Carter ○ Anderson
2. Valuable experience	○ Congress ○ Governorship ○ Private business ○ Armed services
3. Issues	○ lower inflation ○ lower unemployment ○ lower taxes ○ stronger defense
4. Doing a good job	disagree ____________________ agree
5. Comment	

ENTER phase

1. Vote for	⊗ Reagan ○ Carter ○ Anderson
2. Valuable experience	⊗ Congress ○ Governorship ⊗ Private business ○ Armed services
3. Issues	2 lower inflation 1 lower unemployment 3 lower taxes 4 stronger defense
4. Doing a good job	disagree ____________X_______ agree
5. Comment	No comment

QUESTIONNAIRE/3000

Prototype HP 85/9111A	*1982*	*N. Lerman*
Prototype HP 3000/262xA/17623A	*1984*	*L. Nickerson* *E. Forster* *M. Garrahan*
Alpha test	*11/84*	
Extension to HP 3000/264x/9111 Extension to HP 150/9111?	*12/84*	
Beta test	*1/85*	
Release	*6/85*	

AL - ADVANCED LANGUAGES

HOW FOURTH GENERATION LANGUAGES IMPROVE THE MAN-MACHINE INTERFACE

Kenneth Aalders
COGNOS Inc., Ottawa, Canada

SUMMARY

The progress of systems development from its beginning to the present has meant changes to the user interface as well as to the development methodolgy. As data processing moves to the Fourth Generation, users find that their computer interface provides new features and uses. For the users, Fourth Generation Languages have opened up a tool kit that allows them to become the 'new programmers'. These tools may be used to meet user requirements that until now were invisible to the data processing department because the user did not want to add to the backlog. For the data processing staff, Fourth Generation Language is a productivity interface that requires unlearning the procedural methods of Third Generation Languages and developing a nonprocedural approach.

The traditional system development life cycle must be reassessed as developers find re-development in Fourth Generation Languages more effective than maintaining Third Generation code and prototyping becomes a replacement for a formal systems specification. Also, users now find the computer interface a friendly tool with which they can start to become craftsmen after a few days training.

This paper discusses the improvements in the man-machine interface realized by companies today as they move to Fourth Generation Languages and personal computing. Based on current trends, it suggests how this interface may develop in future.

INTRODUCTION

'Based on current trends and forecasts, there are not enough people to meet the needs even if every man, woman and child became programmers.' This statement was made by EDP industry analysts approximately 5 years ago. For those of us who were then a part of that industry, the concept of training everyone so that they could get results from the computer was unrealistic. Yet we knew that something had to happen to overcome the growing application backlog. Today, companies are developing more applications in one year than they were planning to accomplish in five. They have completed building their backlogged applications. This was not done by increasing the number of programmers. What has happened is Fourth Generation Languages - a new way of building applications and meeting user needs.

With Fourth Generation Languages an environment is created where the computer resource is accessible to the people who need it's data. The man-machine interface of the Fourth Generation Language has improved to allow the non-programmer to obtain results from the computer.

The development of the man-machine interface will progress with new technical innovations. An examination of the changes in the man-machine interface, as the computer has developed, shows why improvements have been made and provides an understanding of what changes may be made in the future.

THE SYSTEM DEVELOPMENT GENERATIONS

When experienced data processing professionals have the opportunity to get together they often like to compare old war stories and lightheartedly discuss their data processing roots. What was the first computer you ever used? How much disk storage and memory did it have? Do you remember the card reader, card punch and those paper tape machines? In today's data processing environment of Supermicros, Superminis, Mega-mainframes, local area networks, fibre optics communication, color graphics workstations, etc., the improvements in hardware technology is humorously obvious. However, the same is not true in application development technology. Software improvements have been realized, but not to the same degree as hardware. The Fourth Generation and Fifth Generation are the hot topics in software development today. A review of the earlier generations helps uncover the fundamental changes in the way that man interacts with the machine.

FIRST GENERATION

The First Generation of data processing was from 1940 to the mid 50's. The computer language was binary and in order to use the machine, hard-wired logic boards were used. This interface was usable only by those who were trained in the specific computer's machine language.

SECOND GENERATION

The Second Generation introduced symbolic assembler languages and the use of magnetic media for storing program source code. Writing program code in textual format was an improvement but still required programmers specially trained to develop programs on behalf of the people who wanted to use the computer. Program flow charts were often used to document the user's requirements.

THIRD GENERATION

The Third Generation of application development came into use in the 1960's. Programming languages such as ALGOL and FORTRAN were developed for scientific applications, other languages such as COBOL were developed for commercial applications. Although COBOL was never as machine-efficient as assembler, it was generally accepted as a better way of developing systems. COBOL became the most commonly used programming language. Database management systems and parameter-driven utility programs came on the scene, assisting the user with the difficult tasks involved with consistently and efficiently storing and retrieving data. This step to the Third Generation was a milestone. The name 'high level languages' was used for these languages because they addressed the human side of processing - the man-machine interface. The COBOL compiler took on an English-like form, a more natural interface. One COBOL instruction translated to many machine instructions, which provided programmer productivity. Data processing organizations were set up to handle the computer resource and corporate information systems. Structured systems specifications were used to formally document user requirements and many data processing organizations formalized a System Development Life Cycle (SDLC) to ensure that a controlled methodology would be used in the development of all applications. The objective of the standardized life cycle was to turn the process of developing systems into more of an engineering discipline. The basis for the SDLC was to document clearly what the users perceived their requirements to be, freeze the document and have the users sign it off so that the programmer(s) could develop the system as documented.

Often the formal document was verbose and led to misinterpretations by the user, analyst or programmer. However, having been signed off, it was the system that the user got and unfortunately was locked into. The user community began to see the potential of how automation could make them more productive. The data processing department, having difficulty coping with the users' enthusiastic requirements, felt user demands were excessive. Under these conditions, organizations requested application systems in a much faster timeframe than they could be developed. Users could not get the information they needed and decisions often had to be made on out-of-date or inaccurate manual systems. The organization's needs were not being satisfied. Applying the engineering principles of the structured techniques provided some improvement, but not enough for the technology of today. Today, there is a technology that mimics the activities of building systems. This is the technology of Fourth Generation Languages.

FOURTH GENERATION

Fourth Generation Languages provide a whole new structural way of building systems. The emphasis is on making the person's time on the computer more profitable. They are high-productivity languages. Productivity is gained by two principle factors: Fourth Generation Languages greatly speed up the development process and they allow non-programmers to obtain results from the computer.

The speed-up of development addresses the man-machine interface of the data processing staff. Fourth Generation Languages are specification-driven. They have a powerful list of features that allow the user to create data entry and update screens, or an intelligently formatted report, with just a few statements. Users specify what they want, not how to get what they want. It's like getting a taxi. You don't tell the driver to go 100 yards, turn left and turn right after 2 blocks; you tell him to take you to the airport. If the Fourth Generation Language has an active dictionary, it will be able to generate reports, screens and files automatically, plus enforce integrated standardization of the data attributes and relationships in the user's applications.

By allowing the non-programmer to obtain results from the computer, Fourth Generation Languages directly address the man-machine interface between the computer and the end user. It is essential that a non-programmer be able to learn a useful subset of the Fourth Generation Language in a few days. Fourth Generation Languages are interactive and provide active user assistance by way of interactive help messages, informative error messages and intelligent defaults that cover the obvious and tedious tasks in building applications. Since Fourth Generation Languages were developed for online operation, the interactive refinement iteration process of prototyping succeeds as a development methodology that delivers an order-of-magnitude improvement in productivity.

Fourth Generation Languages have changed the application demand-and-supply balance by introducing a new class of programmer - one that has never programmed before. Users are stepping up to their opportunity to get results from the computer that helps them with decision making on an 'as required' basis. The benefit is there.

The technical innovations in hardware have out-paced software. The tremendous improvement in computer power and price is attracting those looking for sub-second response, and consistent online availability. The relationship between the data processing department and user is now being impacted by a new phenomenom - micro mania. The man-machine interface of the Fourth Generation Languages is typically a result of the organizational model conceived and implemented by the data processing department. The microcomputer now adds personal computing to the data processing environment.

THE MANY INTERFACES OF THE DATA PROCESSING ENVIRONMENT

In today's business organization, the prime responsibility of the data processing department is to be the keeper and protector of the corporate information resource. As data processing departments move to Fourth Generation Languages, the change in the man-machine interface is reflected on the way they interact with other parts of the organization.

The demands for information from production systems and the requirements for new systems came from three basic user types. Business executives require summarized tabular report information, usually on a scheduled basis, in order to view the health of their company in the marketplace. Their ad hoc requests are usually handled expeditiously. Operational managers demand information systems to support their functional areas. Their requirements are a mixture of scheduled and ad hoc reports based on the data in the corporate database and related directly to their areas of responsibility. The information reported is commonly summarized lists with enough details to pin-point any specific problems or exceptions. End users are the operational staff who need application systems that they may use as tools to make themselves more productive in their business activity. They enter and retrieve data on a transaction or record basis, and run summarized listing reports to monitor the on-going activity of operational information.

THE TRADITIONAL DATA PROCESSING ENVIRONMENT

Traditionally, the data processing department's role of supporting the corporate information resource includes the responsibility of acquiring, maintaining and planning adequate computer capacity, and the responsibility of developing and maintaining all of the application systems required by the users.

In many Third Generation development environments, application development and support is needed in a shorter timeframe than it can be delivered. An application backlog results, as the development of new important procedures and systems is inhibited by the difficult, time-consuming problems of maintaining existing code. If users are not receiving the information they require in a timely fashion, then many business decisions must be made without optimal source information. The organization's needs are not satisfied.

FOURTH GENERATION LANGUAGES IN THE DATA PROCESSING DEPARTMENT

Virtually every Fortune 500 company in the United States is using some kind of Fourth Generation Language. The primary reason for this is the pressure from the organization's user community to do more in less time. The company that does not automate in every area could actually fail. The organization's dependency on technology demands an application development tool that will remove the application backlog. Fourth Generation Languages promise this.

Typically, when an organization moves to a Fourth Generation Language, a trial is conducted by the data processing programming staff. A trial is advisable as there are a number of products marketed that do not deliver the 10-to-1 productivity gain and user-friendliness that they advertise. When the data processing staff incorporate the use of a Fourth Generation Language, their man-machine interface receives a powerful shift that benefits the developer. Screens, reports and files can be generated with only a few commands, using the nonprocedural, specification-driven language. Within a few days, the developer finds that functioning prototypes can be built in less time that it takes to write a program specification. However, some experienced programmers may find it difficult to unlearn the procedural techniques of the Third Generation Language that they prided themselves on.

The process of prototyping presents a revised approach to requirement definition and the Systems Development Life Cycle. Prototyping is an adaptation to the uncertainty principle in Physics - 'the act of observing subatomic events changes those events'. In application development, the act of providing a user with what he perceives as his requirements changes his perception of those requirements. A prototype is a representation of an application that becomes the full functioning application after a process of improvement iterations. In Third Generation methodology, many of these improvements would not be realized until the application was implemented and would subsequently add to the maintenance. With the Fourth Generation Language productivity improvement and the prototyping capability, most data processing departments abandon the formal approach and open up to the user.

The interaction between the user and the computer is filtered by the application developer. However, prototyping with the power of a Fourth Generation Language is fast and the developer and user can work together in a creative partnership that keeps the user actively involved and maintains his interest. A Development Center can be set up to provide a functionally integrated environment for the developer and user to create an application without interfering with the other corporate-level activities of the data processing department.

Although applications are built more quickly and accurately, this does not necessarily diminish the backlog. Often the users bring forward other information system requirements - invisible backlog. They orginally decided against putting these requirements in the backlog because they knew that the data processing department did not have time to develop them.

The maintenance workload on data processing staff changes because it is often easier and faster to rewrite a program with a Fourth Generation Language than to debug Third Generation Language code.

Another area where the man-machine interface changes is in the area of machine loading. The 10-to-1 increase in productivity results in many more applications being implemented on the hardware resources. This should be reflected in data processing's capacity planning or the man-machine interface will be too limited in use.

USER DEVELOPMENT WITH FOURTH GENERATION LANGUAGES

Fourth Generation Languages were created for two principle reasons: first, to greatly speed up the programming process; and second, to allow non-programmers to obtain results from the computer. Some Fourth Generation Languages are very user-friendly, and users become competent at obtaining useful results after a few days training. The data processing department has the opportunity to push productivity out further into the organization by making more users productive, bringing the non-data processing users into the automation arena.

A positive and safe first step for the new programmers, is to use the reporting capability without an updating function. With some training on the Fourth Generation Language and their databases, users will be able to generate and modify reports of data they need to help themselves be more productive. As users take on this activity they are also taking on some of the maintenance workload and application backlog. They not only enjoy this capability, but because of the increased information understanding they can achieve, they demand it. The user will now be able to take some of the responsibility in the reporting areas of the application system .

In order to understand the impact of Fourth Generation Languages on the man-machine interface, it must be understood that the capability of building application systems is no longer the sole property of the data processing department. User development with Third Generation Languages failed because the tool required too high a level of expertise from the user. Fourth Generation Languages provide features and uses not previously available - a tool kit that is friendly enough to allow the new programmer to become a craftsman.

Realizing that we are now at a point where more computers are being built than there are people graduating from formal data processing training, user computing becomes somewhat of a necessity for the organization. Whether it is a micro or a terminal to the central computer, users now have their own man-machine interface. Once users have learned a Fourth Generation Language they have the tools necessary to change and build applications. There is a philosophical shift, so that users can now have entire control of their applications. As this control rapidly moves from the data processing department to the end user, some data processing managers view the evolution to end user computing as a defacto drain on their traditional power base. They perceive a personal computer and Fourth Generation Language intoxication. New programmers may believe that, because they've used these automated tools, results are always accurate. But, because data can be extracted in different ways, it may be susceptible to incorrect conclusions and this results in a "garbage in gospel out" credo that can produce a 'Tower Of Babel' effect. As the traditional keeper and protector of the corporate information resource, the data processing department has very justifiable concerns with the controls needed for data security and integrity. As this shift in control of the application takes place, user programming hours exceed those of the data processing department. How the users and data processing department identify and react to their new roles, will have a major consequence on the success on the man-machine interface of the new programmer.

USER COMPUTING

The data processing environment of Fourth Generation Languages must realize the potential of the man-machine interface of user computing. It exists conceptually as soon as any system development takes place outside the data processing department and, with the advent of the personal computer, it is a reality.

Once users have the ability to use Fourth Generation Languages, they will be able to put applications together, extract information, and put it into a graphics format so they can better understand their complex data. Or, they may load that data into a spreadsheet, put it on a floppy and take it to study in the privacy of their own home. Organizations find that by using the information center, users are able to upgrade their understanding of the corporate information resource.

In a development center, data processing staff develop the applications that are of benefit to the corporation. There will still be users who don't want to be programmers and they need the development center to build their application. The data processing department may try to persuade the user to create the reporting side of the system. However, an analyst will be there to ensure the performance of that system, and the user will be able to concentrate on the subject matter for that application.

The opportunity of user computing can be taken advantage of by creating an Information Center to assist users apply their own knowledge and creativity to solving business computing problems, adding to the organization's data processing resource. The data processing manager must motivate his staff to taking a consultative and advisory role. This is a positive step, as the data processing staff become the instructors bringing users and their applications together.

With the shift to user computing, the data processing department's role as keeper and protector of the corporate information resource becomes increasingly important as demand for that information increases. The data processing department supports user copmuting in the information center by teaching the use Fourth Generation Languages, and microcomputer systems.

Once the new programmer has learned a Fourth Generation Language he should require little mental effort in continuing to use it. He can therefore concentrate on the subject matter of the application, which results in more effective applications. Using these new programmers as a system development resource, the data processing department can free itself from much of the maintenance workload. Also, users themselves will be able to develop systems that will take away from the application backlog. By no far stretch of the imagination does this mean that an organization's data processing needs can be totally satisfied by user computing.

There a number of problems that can be identified with this shift of application control to the user. To the experienced data processing staff these problems are self evident. Being new to applications development, the new programmer will lack fundamental system design concepts and fall prey to many of the traps the experienced system designer knows about. The resulting system design may not be concise or complete, and may even include redundancy. The new programmer, being unfamiliar with system development tools, have a problem due to improper use of the Fourth Generation Language. Problems may also occur due to the lack of data management concepts. Although much of this should be handled by the data dictionary component, there may be problems with such things as backup or data security. These types of problems have already been solved a number of times in the data processing department. In order to overcome the problems associated with the shift of application control to the user, the data processing staff should remove these problems without trying to take control of the user's application. In other words, bring end user programming into the mainstream of data processing.

COMPUTER POWER TO THE PEOPLE

With Fourth Generation Languages, organizations are providing the opportunity for all employees to get results from the computer. These users can become experts in applying software solutions to their business problems. Using Fourth Generation Languages, users can avoid the bottleneck of the data processing department and interact directly with the computer. For the new programmer, the success of an improved, user-friendly man-machine interface is productivity, both in development and usage. They now have the potential to make better business decisions.

Originally, employees feared obsolesence due to the new computer technology. Machines do the work of people. Now that computer power can be given to the people, employees fear obsolesence if they do not get the opportunity to use it. With Fourth Generation Languages, non-programmers have become computer users and application experts, building their own applications. The primary feature of a Fourth Generation Language that provides this capability is user-friendliness - data processing's most obvious and important trend.

User-friendliness is derived from ease of use. Applications are more easily built, using a development tool that mimics the development process. Application systems will be easier to use if it mimics the real-world application. Another parameter in this user-friendliness is the consistency of the man-machine interface. Users will find their man-machine interface friendly if it provides satisfactory online response and availability. By solving the problem of backlog and maintenance with user computing, the data processing department has created another. It is the data processing department's responsibility to ensure the security and integrity of the corporate information resource and provide the computer power required to adequately support the ten-fold productivity gain of user computing.

When IBM formally introduced the Information Center in 1979, it's purpose was to supply computing power to users to relieve backlog, leaving the data processing department to secure the corporate computing environment.

HOST ENVIRONMENT

As user computing and corporate information grows, the host or central computer resource must grow to meet its demands. Host environments have a high overhead of system operational software. Pushed to the limit, computer resources are dedicated to housekeeping and no useful computing throughput is performed. The data processing department must re-address their computer environment and investigate deploying it's computer resource with today's technology.

The concept of a host environment is central to the future of information processing - there will always be centralized control of the corporate information resource. Physical form is less important than the role it plays to control the growing data and information needs. The host becomes a utility, to support the large-scale transaction processing and protect the corporate information resource to which all want access. The host will have the vast job of data collection, storage, corporate reporting and analysis.

MICRO PROLIFERATION

There are many options for deploying the computer resources required to support the Development Center or Information Center. The most popular implementation in today's technology is the microcomputer - mainframe on a desk.

The proliferation of micros has happened. In general, the acquisition of micros is not managed or planned into any overall strategy. A significant number of companies are spending more on microcomputers than data processing computer resources. For many companies micros have fallen short of their potential. Because the data processing department was not involved in the planning, these micros were acquired with incompatible hardware and application software. They do not fit into the data processing environment, and in some cases are used only as terminals. Micros are often brought into a company because they are seen as tremendous solutions. Unfortunately, there is not a clear understanding of what problem they should solve.

Companies with a cogent information policy have implemented the microcomputer resource to the benefit of their users and data processing department. Connecting the micro and the host with a controlled link, brings the user the best of all worlds. The user-friendly, menu-based micro products provide the new programmer with tools that can be used to explore data that has been down-loaded from the host.

The man-machine interface of the microcomputer sets new standards for the software products industry. Packaging includes well written manuals, tutorials, quick reference cards and documentation upgrades. In use, the more popular micro products emphasize user-friendliness, making it easier for the user to get results from the computer.

In combination, the microcomputer, the user-friendly software tools, the information link to the host and authorized use of the corporate information resource provide a decision support workstation where the user is provided with a tremendous change of mind set.

We are now at a point in our industry where technology offers the potential for everyone to get useful results out of the computer. For those of us who have experienced this improvement in the man-machine interface, there is a realization that this technology may be available for our use. The next generation will demand it!

THE MAN-MACHINE INTERFACE OF THE NEW GENERATION

It is not uncommon to see a father being advised by his children when he is about to purchase a personal computer. The belief that only professionally trained people are able to get meaningful results from computers is a preconceived notion tied to our past. Computer literacy is in such high demand, that it is being taught at the junior high school level and the majority of today's children will be using computers in their future careers.

Since the beginning of the electronic data processing industry, technological progress has continued to remove the constraints of building applications. The physical user interface has progressed from wired logic boards and 80-column punched cards, to personal computing with ergonomic keyboards, displays, touch screens and the mouse. The improved hardware environment has provided the processing power needed to support the new methods of developing systems where less skill is required of the user, because more intelligence is inherent in the computer. The logical constraints of the user interface have progressed with less magnitude, but the application development software of the Fourth Generation has provided great improvement. Powerful nonprocedural tools provide a consistent, menu-oriented development environment where more applications are built by less-skilled users. The computer is adapting to the person.

Fourth Generation Languages have provided an order-of-magnitude gain in application development by providing tools that mimic the development process. People, who are not trained or experienced in the development process, can have difficulty learning this process when getting started. As a result they lose interest in this promised productivity. The next order-of-magnitude gain in application development will be made by providing a tool that automates more of the application building activity and adds more intelligence to the nonprocedural language interface, thus allowing the user to become productive more quickly.

THE NEXT GENERATION

The artificial intelligence of future Fifth Generation Languages promises a natural-language, voice interface to expert systems. The man-machine interface will be at the specification level. The organization of specifications into the facts needed to build the application software will be carried out by the software development system. The race to Fifth Generation is on, but it will be a number of years before the hardware is built, that is going to support it. Many steps will be taken on the way and many improvements to the current technology will be realized before we reach the level of Fifth Generation. Much of today's research directly addresses the man-machine interface and may soon lead to further improvements in Fourth Generation Languages.

With the new technology, things can be done in new ways. A front-end environment has been defined to focus directly on presentation of computer applications to the user and actively adapt them further into natural interpersonal communication. The man-machine interface can be explored and manipulated to improve the mechanisms represented by the application system. Research has shown that it is possible to achieve an order-of-magnitude gain in system use by prototyping this front-end environment of the man-machine interface.

One such project directly addressed adapting the machine to the user by setting up a link to the application system parser, and then intercepting and evaluating syntax entered by the novice user. The user, knowledgeable in the application, was given a small, concisely written user's manual and a set of tasks to perform. As syntax was entered, it was evaluated and passed to the parser if correct. Entries that were commonly specified and and were evaluated as natural application terminology were added to the parser syntax tables. Several iterations with novice users were conducted. The results showed that, with the original syntax, 7% of the novice users actually completed the assigned tasks, whereas, with the final syntax, 76% of the novice users finished. An order-of-magnitude improvement was achieved.

Another project involved restricting application features based on the user-designated skill level. The lower the skill level specified, the fewer options available on the user's menu. Online training was incorporated into the system design and, by progressively blocking the user from less frequently used options, the user's ability to learn the complex system interface was significently improved.

This research shows some of the benefits of user interface prototyping. The key to automating applications lies in the recognition and support of user participation. Soon, we may see application software that will incorporate user skill levels and allow syntax customization. Users may be provided with the capability to enhance the user-friendliness of the application.

PHYSICAL CONSTRAINTS

There has been proportionately little enhancement to the operational environment of Fourth Generation Languages. Software engineering activities in system development life cycles, system requirements definition, error detection and correction, and application documentation, have not provided improvements with the physical constraints of hardware. The following are some of the problems that Fourth Generation Languages have not been able to resolve:

- internal conflict of machine resources
- queueing
- limitations of memory space
- limitations of the type and number of peripheral interfaces
- limitations of the number of simultaneous activities by one CPU
- limitations of throughput for heavily loaded systems

Improvements in the physical environment of application systems may be realized by taking advantage of a powerful phenomenon that has recently impacted the data processing industry - the personal computer. There is no economy of scale in pure processing - quite the contrary. With the overhead of system software, heavily loaded systems do more housekeeping than application processing. The opportunity for improvement involves the use of microcomputers in a distributed architecture. A decentralized data processing environment can be created by deploying computer resources throughout the organization on an 'as required' basis. Local area networks and read-only, micro-to-mainframe links can provide an integrated interface for sharing resources and providing users with the corporate information that they need.

WORKSTATIONS

Secretarial workstations with document processing; financial analyst workstations with corporate information downloaded into spreedsheets; and management workstations with decision support, graphics, ad hoc report writer, corporate report file browser, project management and executive stock analysis tools - all these supply needed computer power to users while relieving the central host computer. The data processing department is responsible for a secure corporate information environment. The implementation of user workstations requires cooperation with the data processing department. They can provide valuable assistance with application software, system software and hardware selection. Information Centers are used by the data processing department to centralize the microcomputer resource. User workstations are implemented to centralize the user's work environment. The Information Center is extremely practical for teaching novice users how to get started. However, the workstation is the tool that will make users more productive. To adapt the machine to the person, the workstation must be located in the person's local environment.

User computing places the burden of satisfying user requirements, including maintenance, on the user themselves. This provides the data processing department with the opportunity to re-assess their role and priorities. They must remain responsible for all corporate computer resources and therefore be involved in user computing, helping the users become experts in applying software to business problems. They must assure the security and accuracy of the corporate information resource from unauthorized access and data contamination. They should upgrade the usability of the information allowing users to access the data that users need to build business systems and make business decisions. The data processing department must concentrate on the corporate systems requirements and set up a data communication environment where data can be shared, files can be transferred and processors can be interconnected. There should be a user computing policy set in place to guide users and establish standards. This will be especially valuable if the data processing department has to at some time pick up the application on behalf of their user.

THE DATA PROCESSING DEPARTMENT'S NEW ROLE

User computing will provide the data processing department with the opportunity to take on new activities. A certification program may be put in place to allow the sharing of applications, if they meet the standard guidelines. They may solicit software evaluations from users and make them available for other users. They may set up a help line to the Information Center to provide quick support to small problems. They may have computer literacy seminars and product demos, raising the consciousness of the user community. They may assist with user groups, developing user knowledge in areas specifically important to user computing. User computing can become a vast resource for the data processing department.

I MUST GO NOW, FOR THERE GO MY PEOPLE AND I AM THEIR L
- attributed to Mahatma Gandhi

When computers became popular as business tools, organizations set up data processing departments to address the automation of corporate data systems. The data processing department was made responsible for the maintenance and accessibility of the corporate information resource, upon which management based their business decisions. The man-machine interface has continued to change and progress with each new generation of system development technology. One of the numerous benefits of this progress is a higher level of 'user-friendliness' in that interface. Fourth Generation Languages were built to achieve maximum results with minimum effort. This has resulted in applications being built in a faster timeframes and with fewer errors, by people with fewer data processing skills.

Now the insatiable demand for computer applications and information reveals itself in a proliferation of microcomputers. The data processing department must upgrade the effectiveness of the corporate computer resource by providing users with the ability they demand - getting results from the computer, so they can make better business decisions. Fourth Generation Languages in the Development Center and Information Center will provide a man-machine interface that will allow the non-programmer to become the 'new programmer'. Microcomputer based workstations can provide a man-machine interface where people can use computer resources directly to fullfil their needs. Controlled access to the central corporate information resource, in combination with Fourth Generation Languages and workstations, can provide users with the man-machine interface that fully meets their demands. They can now make better business decisions.

BIOGRAPHY

Kenneth M. Aalders is PowerHouse Product Development Manager at COGNOS Inc. He is responsible for the research, development and delivery of software in the PowerHouse product line. He received his Bachelor of Science from Acadia University and has been with COGNOS Inc. for 6 years. During his 12 years in the data processing industry, he has gained technical experience as a programmer/analyst, technical consultant, data management consultant and project manager. Working in both scientific and commercial application areas, he has developed information systems using structured systems development methodology, prototyping, and various database management systems.

AI IN THE COMMERCIAL MARKETPLACE

Sharon Bishop

Hewlett-Packard, Cupertino, USA

SUMMARY

Artifical intelligence is popularly defined as the science of enabling computers to reason, make judgments and even learn: and it has received a lot of attention lately. Optomistic analysts forecast a market growth from $20 million in 1984 to somewhere between $2.5 to $5 billion in 1990. Stories about 5th generation computers, and the Japanese effort to be the first to build a new generation of "thinking" computers by 1992 has added urgency to efforts in the U. S.. Machines that eliminate the need for highly paid human experts, computers that speak and undertand speech -- all these and more have been the subjects of scores of articles in the press.

For AI technology to grow, it must be applied, not to narrow specialities, but to situations and problems encountered in a wide range of industries. Hewlett-Packard has for some time been developing this technology at our Corporate Research and Development laboratories and we now have AI projects staffed in the areas of manufacturing, medical, analytical, customer services, engineering design and the applications part of our computer group. We are monitoring the market closely and expect to be in a strong competitive position in the 1985/1986 timeframe.

This paper sets out to give a brief overview of AI -- how AI differs from traditional computing methods, what application areas are currently receiving attention, trends in the AI market and some future predictions for the marketplace. It does not go into depth on any of Hewlett-Packards specific products, but does point out areas where activity using AI techniques is underway. It is intended for commercial users that are looking for a simple introduction to this burgeoning industry.

INTRODUCTION

As a product manager in Hewlett-Packard's Computer Language Lab, I have developed an awareness of artifical intelligence, or AI, mainly through the demand from our sales force for AI languages on our hardware. Computer Language Lab provides the programming languages for a range of Hewlett-Packard's product lines. In particular, we have seen a growing number of requests from our commercial installed base to hear more about the AI subject, thus, our keen interest in tracking the commercial and technical movement toward this technology.

WHY IS HP INTERESTED?

You may ask youself "why would Hewlett-Packard get into the artifical intelligence market?" While computers have been growing dramatically in power and shrinking in price, the software needed to make them useful has become more complicated and costly to develop. Hewlett-Packard is focusing on shortening the life-cycle of software development by concentrating on standardization and productivity. We see artifical intelligence as the beginning of a process that could revolutionize both the computer hardware and software industries.

If we look historically at the evolution of the applications development cycle, we can see the progression from complexity of use and the requirement for technical expertise toward high level productivity tools and ease-of-use for the end-user. We see the real market potential today in the area of 4th generation tools and artifical intelligence.

ANATOMY OF AN EXPERT SYSTEM

For 30 years, AI scientists have been trying to develop computers that can think and learn the way human beings do. Early examples of building computers that could play chess were not taken very seriously by the practical businessman. Now, however, some solid examples of successes in the area of expert systems has captured the attention and imagination of the commercial market.

"Expert systems" is a term that is commonly equated with artifical intelligence. An expert system is a program that can reason and make decisions, based on extensive knowledge of a given field such as geology or medicine. It generally consists of two parts: a knowledge base containing facts and rules which is a sort of database, and an inference engine which "knows" how to manipulate those facts to get results.

Expert systems or knowledge-based systems as they are sometimes called, is an area that Hewlett-Packard is aggressively persuing for the marketplace. We have several internal projects underway and I will refer to some examples as I progress through my overview of AI application areas.

AI VERSES TRADITIONAL COMPUTING

Although the expert systems now available are still far from the ultimate goal of building machines that can reason as people do, learn from experience and communicate in human or "natural" language, they can perform tasks that are well beyond the reach of conventional computers. Expert systems can transform computers which have always been "dumb" calculators into machines that can draw conclusions from huge stores of data -- much like the human brain can do. Unlike a conventionally programmed computer, which can only follow step by step, a rigidly defined set of commands, expert systems are programmed to mimic human reasoning.

Let's compare AI to traditional computing. What best differentiates the operation of an expert system from a conventional computer is it's flexibility. For example: a program that directs a standard computer in conducting huge numerical calculations is written so that the computer performs one task as quickly as possible. The program specifies precisely how all the operations must be performed using a very rigid structure.

In AI, the program tells the computer what to do, without specifying how to do it (indeterminate or inexact limits), compared to the traditional method of telling it what to do and how to do it (determinate, having exact limits). This is achieved by using special computer languages such as LISP or Prolog, that allow a computer to manipulate symbols or concepts rather than numbers. In addition to special programming langugages, the decision-making rules and the facts, or knowledge used to reach a conclusion are separate. New data can be added at any time without altering the program. This flexibility of having the rules and the data separate distinguishes expert systems from other programs.

In an expert system, a typical rule might be: "If the car won't start, then the battery may be dead". The computer will check to see what other symptoms must accompany the car not starting in order for it to be a dead battery: "car in the proper gear, is the key in the ignition, is the gas tank not on empty, etc.". If the program was provided the other information about the situation, it would make a diagnosis of the problem and show the rules it used to reach the conclusion, such as "Chances are 75% that the car has a dead battery because the key is in the ignition, the gas tank is not on empty and the car is in the proper gear".

There are generally two ways to use an expert system. One is as a consultant to human experts, as in a program developed at Digital Equipment Corporation, call XSEL, that DEC's salespeople use to configure a computer system for their customers.

The other way is to use the expert system as a substitute for human experts when such experts are difficult to get. An example of this is a program developed by SRI International call PROSPECTOR which imitates a human geologist in locating ore deposits.

Many market-watchers of AI are convinced that expert systems will quickly, not only broaden the use of computers throughout their current areas of application, but move them into new markets as well.

EXAMPLES OF EXPERT SYSTEMS APPLICATIONS

Expert systems application areas leading the way include:

FINANCIAL SERVICES: This may be a very large market. Millions of dollars flow through financial establishments on a daily basis. Expert systems are ideally suited to areas where the main line of business is "professional judgment". Many of the decisions made today are made on an intuitive basis or "gut-level" feel based on years of accumulated knowledge about the banking profession. This knowledge could be in a knowledge base with facts and rules, allowing the financal expert to quickly see in dollars and cents what his decision may mean. Eight start-up companies are exploring niches in the area of finance. A recent tool from Inference Corp. which has "time-based" capabilities may pave the way for more development of expert systems in this area, since financial statements contain time-based information.

MEDICAL: For some time, expert systems to diagnose diseases has been a natural application for AI because much of a doctor's skill involves applying rules to evidence presented by a patient. An expert system called HELP, marketed by Control Data Corporation, took 15 years to build and cost $30 million. It integrates information from doctors, laboratories, nurses stations, pharmacists and other sources and suggest possible diagnosis and treatment.

INDUSTRIAL; MAINTENANCE AND SERVICES: A T & T has announced an expert system to be available this year to quickly locate faults in telephone cables. It does in an hour a job that took technicans a week before.

General Electric has an expert system to help mechanics repair diesel engines on it's locomotives by asking progressively detailed questions until it can home in on the problem. This has market potential for railroads and aircraft engine repair.

Hewlett-Packard has several activites in the area of customer service that use AI techniques. These applications are based in the AI language LISP and will provide trouble-shooting and problem resolution using knowledge bases containing large amounts of diagnostic data. Pro-active tools will allow us to work smarter and faster in servicing our customers. Since customer service is our number one objective and gives us a competitive advantage, we are agressively persuing expert systems in this area.

We are also developing a system to provide network consulting for configuring HP3000's to our personal computers. This system would assist our field engineers in accurately configuring the cabling requirements and other components needed in configuring a large distributed system.

MANUFACTURING: This is a fertile territory for expert systems to be the brain for robots and to assist managers and engineers in decision-making with massive amounts of data. HP and DEC both have activity in this area.

Hewlett-Packard has active projects internally in the expert systems area for integrated circuit design. We are also using AI technology for an expert system to cost out the process for manufacturing printed circuit boards. We are integrating these techniques into tools for design engineers to make intelligent decisions on whether to use the standard process or ask for modifications to the process. For example: an engineer may want to use a resistor in his design that does not appear on the standard parts library. He can use the expert system to see whether it is cheaper to use the resistor in his design and have hand-loading, or human intervention, or whether he should modify the process, which may have consequences, or rippling effects on others areas, or decide to use another resistor already available and change his design. This decision-making process used to be largely intuitive and sometimes costly. Now the engineer can use the expert system to know in terms of exact dollars and cents what his decision will mean.

DEC has activity in this area for floor scheduling. Hazeltine Corporation offers a knowledge system called OPGEN to generate in 90 seconds detailed operation sheets that took industrial engineeers 20 hours to do previously. Westinghouse uses a system call ISIS to generate factory shedules.

MILITARY: The U. S. military has stepped up funding for expert systems projects for such applications as analyzing battle data, pilots associates, robots for use in hazardous conditions, roving vehicles and the like.

SCIENCE AND ENGINEERING: An expert system developed by Teknowledge Corporation for the French oil company Elf Aquitaine diagnosis drilling problems at oil wells. DRILLING ADVISOR considers more than 500 variables that could cause trouble at the drilling site. The company estimates it saves millions of dollars in reduced down-time.

Hewlett-Packard has activity in this area to market a 32-bit LISP based engineering workstation in the 1985 timeframe. It will provide a set of integrated tools for the software developer. A rule based language will be added in the future to provide a complete software development environment for expert systems.

ADDITIONAL APPLICATION RESEARCH AREAS

Applications that have been explored in the academic research area include COMPUTER VISION AND NATURAL LANGUAGES. Spinoffs of these research efforts have resulted in some commercial products, many augmenting the expert systems area.

NATURAL LANGUAGE interfaces allow users to communicate in spoken natural language with expert systems or conventional programs. This market is expected to grow to $1. 8 billion by 1993.

The near term objective for this technology is found mainly in data base front ends. Several commercial products are available such as Intellect from A. I. Corporation. Right now, natural language systems operate with a highly restricted context. Small vocabulary's limit interaction and since they are based primarily on noun's verbs and adjectives, most systems are focused on sentences as opposed to natural discourse.

Within the next two years, it is forecasted that much larger vocabularies will be supported, up to 20,000 individual words may be recognized either by saying or spelling. Within a decade, these systems will be able to understand well-formed sentences much like a newscaster would give. Greater than a decade away we may see unrestricted speech recognition.

COMPUTER VISION is a difficult area. It includes such applications as robots and roving vehicles. It has great potential in the area of machine vision in factories and in the area of the military for vehicle guidance and robotic operations in hazardous environments.

Commercial systems are available and can operate successfully in very specialized environments - machines can look at a silhouette and translate it in a single pass today. The AI Lab at MIT now has a computer that can look at a bewildering array of objects through a TV camera lens and pick out a cup. So can a child do this, but is is a highly complex task for a machine!

Within this decade, additional products are expected in such areas as scanning radar vision for warehousing, aids to the handicapped and medical image analysis. Home robots are thought to be more than a decade away.

TRENDS IN AI

It has been said that AI is the next frontier to conquer. Computers that can reason as people do, learn from experience and communicate with humans in a natural language will one day be realized!

Trends fueling the growth in the U. S. include:

* the entry of established computer companies such as IBM, HP and DEC are endorsing the arrival of this market.

For example, as I mentioned earlier, HP has several internal expert systems projects and expects to announce products for the marketplace in the 1985 timeframe.

Also, at the recent AI conference in Austin, Texas, IBM announced several major AI-related products.

* venture capitalists are staking millions to back promising AI projects: GM recently staked $3 million into Teknowledge Corporation, Merrill Lynch and Venrock Associates are partners in a $3 million pool behind Inference Corporation, Capital Management invested $3.5 million into Syntelligence, a compnay founded by three scientists from Fairchild's AI lab.

* the introduction of many sophisticated software tools and the emergence of small, powerful, 32-bit microprocessor-based AI systems is making the technology available at a much cheaper entry price.

Hewlett-Packard expects to be very price/performance competitive in this area with the introduction of products in the near future.

Also, at the recent AI conference, Texas Instruments demo'd software packages for it's personal computer and demo's a single user LISP computer to be introduced later this year. They also announced a Navy contract to develop a compact LISP computer for use in future "smart" weapons systems.

* the establishment of a U.S. government-backed standard language (Common LISP) will make applications portable among vendors

* the joining of forces between hardware and software vendors will provide total solutions for end-users.

For example: DEC recently announced at the AI conference cooperative marketing agreements with several AI companies such as Inference Corporation, Gold Hill Computers, USC Information Sciences Insitute and Prologica.

Hewlett-Packard is in the early stages of activity in this area.

AI IN THE 1990'S

The future for AI seems very bright. Predictions by experts in the field forecast by the 1990 timeframe we will have:

* good systems and building tools - parallel processors to do multiple tasks simultaneously, super computers that can process large amounts of data at blinding speeds

* intelligent, friendly and robust interfaces - more natural languages, better ways to store data, machines that can deal with analogy and metaphor rather than just rules and facts

* a quantum leap in productivity that will move society from a "production society" to an "information society" where expert systems open the availability of expertise to everyone.

CLOSING

HP will continue to offer our customers the best possible solutions in hardware and software. We have made substantial investments in using artifical intelligence technology in many internal systems that are in daily use today, and we are actively developing products that we will bring to market.

HP cannot promise that this is the beginning of a process that could revolutionize both the computer hardware and software industries, but we intend to stay abreast of market demands and to provide solutions across our product lines, both for internal use and for our customers.

BIOGRAPHY

Sharon Bishop is a product manger in Hewlett-Packard's Computer Language Lab in Cupertino, California, with responsibility for AI languages, COBOL, and Programming Environments. During her seven years employment with HP, she has held senior programming positions, managed a group of programmer/analysts and, most recently, moved into product marketing for computer languages.

NATIVE LANGUAGE SUPPORT FOR THE HP 3000

Harry Kellogg and Jon Bale
Hewlett-Packard Company, Cupertino, California, USA

Summary

This will be a report on new facilities called "Native Language Support" which have been built into the MPE operating system and the major HP3000 subsystems. These facilities make it easy for application programs to be designed and written which give their end users a natural local language (Dutch, French, etc.) interface and locally correct internal processing. The same program can serve various end users in different natural languages simultaneously without reprogramming or recompiling. Native Language Support and its use by application programmers (a one-time effort) will be distinguished from "localization," which is done once for each language or country in which the application is to run. Localization in most cases now requires only translation of text (e.g., error messages).

An Example Application

To illustrate some of the points in this paper, we would like to introduce a small imaginary application. This is an application used to schedule conference rooms for meetings. In keeping with recent product naming tendencies, we call it "HPRoom", but we would like to emphasize that there really is no such application available from Hewlett-Packard.

HPRoom is a simple application. Its most important screen is shown below. This is the screen which is used to reserve a conference room for a particular day and time. In Figure 1, Thomas Jefferson is shown reserving the Philadelphia room for an hour in the afternoon of December 26. Suppose that you were aware of such an application and wanted to make it available to people in your own country. What would you have to do?

First, of course, if your native language was other than English, you would have to translate the text. In the term "text", we include menus, error messages, prompts, soft key labels, and so on. But that's probably not enough.

Notice the fields in which the from and to times are to be entered. This program, having been written in the USA, will naturally expect American time formats: i.e., on a 12 hour clock using the "AM" and "PM" appendages to distinguish between the morning and the evening hours. Similarly, look at the date displayed in the box at the top of the screen. It is of course displayed in a standard American date format: month, day, year.

But more subtly, if in the lower left box, you were to type the name of another conference room: e.g., "Nürnberg", this program upshifts the name before searching for it in its database. But what will typical American software do with an "ü" when it wants to upshift it? Probably nothing at all, since its author didn't know there was such a letter in the alphabet.

```
                              HPROOM
                   CONFERENCE ROOM SCHEDULING PROGRAM
------------------------------------------------------------------------

                   WED, DEC 26, 1984
                   _______AM__________   ________PM_________
                   6  7  8  9 10 11 12  1  2  3  4  5  6
  PHILADELPHIA             XXXXXX XXXXXX    XXXX      XXXXXX

NAME               PHONE NUMBER    FROM:      TO:       NOTES/COMMENTS

[Thomas Jefferson] [1776        ] [2:30 PM] [3:30 PM] [independence    ]

  To get to another room and/or another date, tab down to the following line:

    NEXT ROOM: [             ]    NEXT DATE: [            ]

........ ........ ........ ........       ........ ........ ........ ........
                  PRINT                                      MAIN/    EXIT
                  SCREEN                                     RETURN   PROGRAM
........ ........ ........ ........       ........ ........ ........ ........
```

Figure 1. The HPRoom reservation screen.

Consider what these kinds of functions typically look like, as coded in the application program source. The editing of the time probably looks for a one or two digit number, checks that it is less than 13, looks for a colon and then a two digit number, which is checked to be less than 60, and finally for the characters "AM" or "PM".

The display of the date probably calls the FMTCALENDAR intrinsic of MPE, passing it an HP internal format date and receiving back a formatted date string.

The upshifting of the name is either done by searching for characters with a value between "a" and "z" and then subtracting 32 from them, or else using the VPLUS "UPSHIFT" processing specification.

Or, let us suppose that we have a further screen on which an overview of reservations for all conference rooms for a given day is displayed, and these conference rooms are displayed in alphabetical order by name. This is typically achieved on HP 3000 software in one of these ways:

1. They are already sorted as sort fields on IMAGE chains, or as KSAM keys;

2. The SORT intrinsics could be called to sort a list of them; or

3. The program does a simple sort in memory using some code that looks like:

```
IF NAME1 < NAME2
...
```

But what do the COBOL or Pascal compilers make of such code? They turn it into "compare bytes" machine instructions, which do their comparison based on the internal values of the characters. How will this code ever succeed in placing "Nürnberg" between "Norge" and "Nyborg"?

In order to make this program work for your own country, you need to modify the source. This may not only turn out to be a lot of work, but may introduce new bugs, and it raises questions about who supports the local version, since it really is a new program.

But the recently released HP3000 Native Language Support (NLS) offers an alternative. If a small extra effort is made in writing the original program, then making the program work in another country is often limited to translation of text. What is necessary is to replace hard-coding of those aspects of the program which are language- or country- dependent with the use of NLS features. The most commonly used feature will be calling new MPE intrinsics.

What is Native Language Support?

Generically, "native language support" is a set of facilities within a computer system which aids applications in the proper handling of native language data, including adherence to local customs. Native language support provides the tools which allow an application to meet the user's country- and language-dependent expectations, and allow the application to be localized for different countries. Native language support is used by the application programmer.

Native language support must be distinguished from localization. "Localization" is the adaptation of a software application or system for use in different countries or environments. In short, if the application developer builds in native language support (which is done only once), then the task of localization (which is done once per country or language) is much easier.

Native Language Support on the HP 3000 consists of intrinsics and utilities in the MPE operating system as well as changes made to KSAM, SORT-MERGE, VPLUS, FCOPY, IMAGE, QUERY and COBOL II.

Figure 2 shows an abstract application which has used NLS features to implement its language-dependent functions. The application must have some extra code at the beginning to select at run-time which language should be used for such functions. This can be done in a variety of ways:

1. The application may have its own configuration file containing various information, including the language(s) to be used, or even which language for each possible user.

2. A program may call the "NLGETLANG" intrinsic to obtain the system default language.

3. It may be appropriate to prompt the user directly for the language choice(s).

4. The program may call the "NLGETLANG" intrinsic to obtain the user interface language and/or data manipulation language of the current job/session.

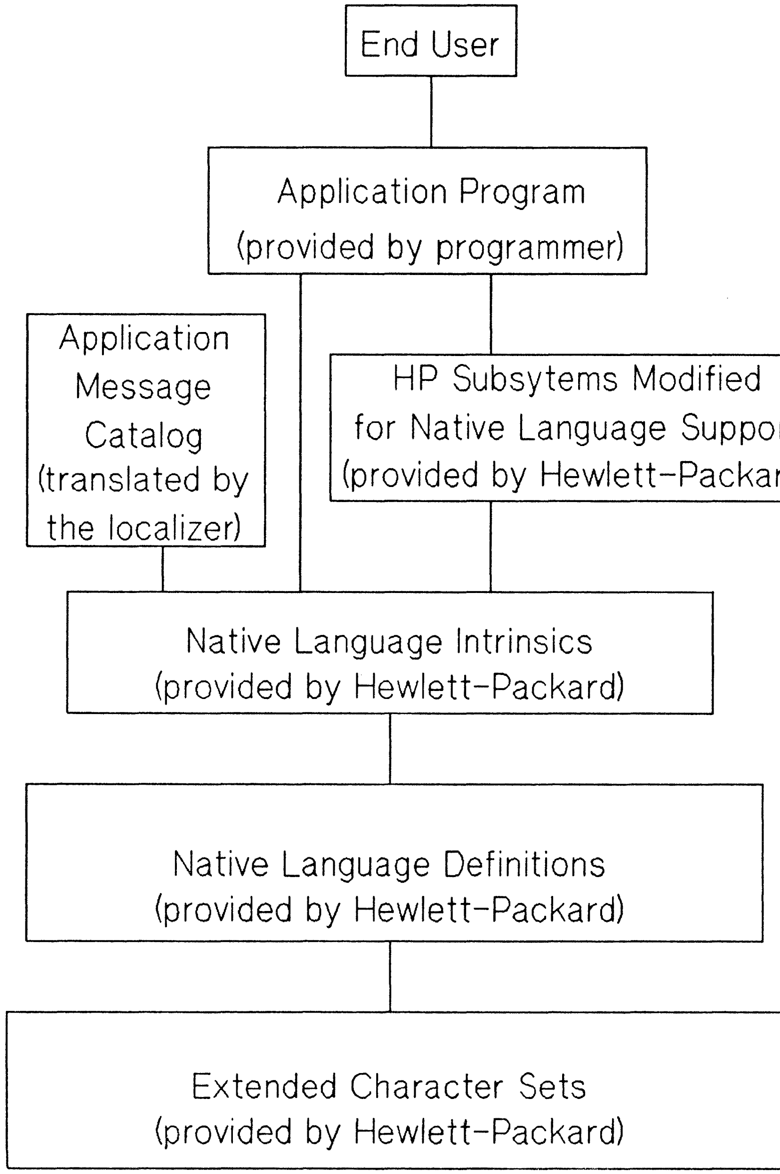


Figure 2. An abstract application and its use of Native Language Support

What Native Language Support is Not

Native Language Support does not provide for the localization of MPE (yet). The MPE user interface speaks the same language that it always has. The intention of the first release of NLS is to make possible localized systems as seen by end users in an application environment. These are the HP 3000 users who are the least technical, and least likely to have knowledge of native languages other than their own.

Localization of applications is not automatic. The application designer has to do some work at the beginning, to use NLS features (e.g., call some new intrinsics) in the application. Then an application localizer for each country or language needs to translate all text into the desired native language.

The application localizer may still have to do some additional work. Not all functions which are country- or language-dependent are handled by NLS features. For example, value added tax calculations for financial applications or hyphenation algorithms for word processing may require that the localizer program a local language or country module for the application to call.

This has been a brief overview of what native language support is (and is not) in general. Now let's look at the details of its implementation on the HP 3000.

Character Sets

Native Language Support is based, first of all, on character sets which are supersets of USASCII. These character sets provide all of the USASCII characters with their normal code values, so that existing software which depends on those characters continues to work. In addition new characters are available to serve the needs of one or more non-U.S. languages. Where possible, character sets providing support for multiple languages are selected, so that data can be exchanged between these countries.

The first release of Native Language Support for the HP 3000 offers two such extended characters sets: ROMAN8, supporting 11 western European languages, and KANA8, supporting phonetic Japanese (Katakana). These character sets are now supported on a wide variety of terminals and printers, as well as in the operating system. See the reference manual ("Native Language Support Reference Manual", 32414-90001) for a complete list of such peripherals.

Languages

The first release supports the following languages:

AMERICAN	ITALIAN
CANADIAN-FRENCH	NORWEGIAN
DANISH	PORTUGUESE
DUTCH	SPANISH
ENGLISH	SWEDISH
FINNISH	KATAKANA
FRENCH	NATIVE-3000
GERMAN	

For each native language which is supported by NLS, a number of characteristics are known. Some of these are really language-related in the traditional sense, such as day and month names and lexical order (collating sequence). Other characteristics associated with a supported native language are, in fact, country or local custom dependent, such as currency symbols and date formats. Another group of data known by NLS for each supported language is data processing oriented; an example of this category is conversion tables between the HP3000's internal codes (which are USASCII based) and EBCDIC.

Within NLS, all of these characteristics -- language related, custom dependent, and data processing oriented -- are considered to be language dependent. All information used by, or available from, NLS is based on the application's choice of language(s). Thus, for example, NLS maintains not only an ENGLISH collating sequence, but also an ENGLISH time-of-day format. In this second case, "ENGLISH" may be considered to mean "used in England" rather than "of the English language".

In addition to the 14 "real" languages supported by the initial release of NLS, a dummy language called "NATIVE-3000" is being introduced. NATIVE-3000 may be thought of as "the way it used to work, before NLS". For example, the NATIVE-3000 collating sequence is simply the order of the characters in the USASCII code; the NATIVE-3000 date format is that returned by the existing MPE intrinsic, FMTCALENDAR. Thus, whenever the language NATIVE-3000 is used in a native language function, the result will be identical to the result of the same function being performed in an unlocalized way, before NLS.

For each language supported, the following characteristics are defined in, and known to, the operating system:

Language name
Language number
Character set which supports the language
Character definitions (actually a function of the character set)
Upshift table
Downshift table
EBCDIC conversion tables (to and from)
Collating sequence table
Decimal delimiter
Thousands separator
Currency sign
Currency abbreviation
Position of currency sign/abbreviation
Month names
Month name abbreviations
Weekday names
Weekday name abbreviations
External date format
External time of day format
External custom (short) date format
Words for "YES" and "NO"
Additional country- or language-specific information (sometimes)

Intrinsics

To make of use of this language information, and to provide the various language-dependent functions for any language configured on a system, new "NL" intrinsics have been written. Each of these intrinsics has a "language" parameter which is passed by the calling program and which specifies that its results should be the correct ones according to the definition of that language. The intrinsics are listed in three categories: information/general, date and time handling, and character handling.

Information/general

ALMANAC	Returns numeric year, month, day of month, and day of week.
NLAPPEND	Constructs a file name from a specified string and a specified language number (used for VPLUS forms files, application catalogs, etc.).
NLGETLANG	Returns "current" languages (system default, user interface, and/or data manipulation).
NLINFO	Returns, for a specified language, all configured information (names, formats, tables, etc.).

Date and Time Handling

NLFMTCLOCK	Formats a time of day into a string, according to the requirements of the specified native language (e.g. "12:27 PM").
NLFMTCALENDAR	Formats a specified date into a string (e.g. "FRI, OCT 15, 1982").
NLFMTDATE	Formats a specified date and time into a string (e.g. "FRI, OCT 15, 1982, 12:27 PM").
NLFMTCUSTDATE	Formats a specified date into the "custom" or "short" date form (e.g. "10/15/82").
NLCONVCLOCK	Converts the external time of day string for a language into the HP3000 internal CLOCK format.
NLCONVCUSTDATE	Converts the external custom (short) date format for a language into the internal HP3000 CALENDAR format.

Character handling

NLREPCHAR	Replaces any nondisplayable characters in a string with a substitute character.
NLSCANMOVE	Moves, scans, upshifts, or downshifts the characters in a string according to the definitions for a language and the nature of the characters (alphabetic, numeric, etc.).
NLCOLLATE	Determines the order of two character strings, according to the collating sequence for the specified language.
NLKEYCOMPARE	Compares a partial key to a whole key to determine chance of match (used with KSAM for generic retrieval).
NLTRANSLATE	Performs conversions between national EBCDIC character sets and HP extended character sets.

Utilities

Two new native language utility programs have been provided:

LANGINST - Before any native languages can be used on a system (except NATIVE-3000), they must be configured by the system manager using this program. The system manager selects which of the supported languages to configure and may modify many of the characteristics of a language being configured.

Modifying characteristics would be useful, for example, for the manager of a system in Austria who wanted to install GERMAN with a different currency symbol than the default for this language.

Changes to a system's language configuration take effect the next time a system is started. At that time the definitions of the configured languages are loaded into system global data segments. There is one data segment plus one for each language configured.

NLUTIL - This program may be used by anyone to obtain a list of languages configured and the details of their definition.

Application Message Facility

A final new utility and three new intrinsics form the Application Message Facility. As stated earlier, a program which is easy to localize has as one of its characteristics that no text is stored in the code itself. This allows the textual material to be translated (part of the localization step) without recompiling. Therefore a good text handling facility is essential to native language support. In NLS for the HP 3000 this is the Application Message Facility.

GENCAT - This utility program is used to convert a source file containing messages into a usable application catalog. The source file is simply a normal EDITOR file containing character strings (messages). These messages may, of course, contain "extended" native characters. Each message is uniquely identified by a set number and a message number within the set. The GENCAT program is used to read the source file and to create a formatted message catalog for run-time use.

Application programs then use the new intrinsics:

CATOPEN - to open a message catalog;

CATREAD - to access and display messages from the catalog, including insertion of variable parameters into the message;

CATCLOSE - to close the message catalog when finished with it.

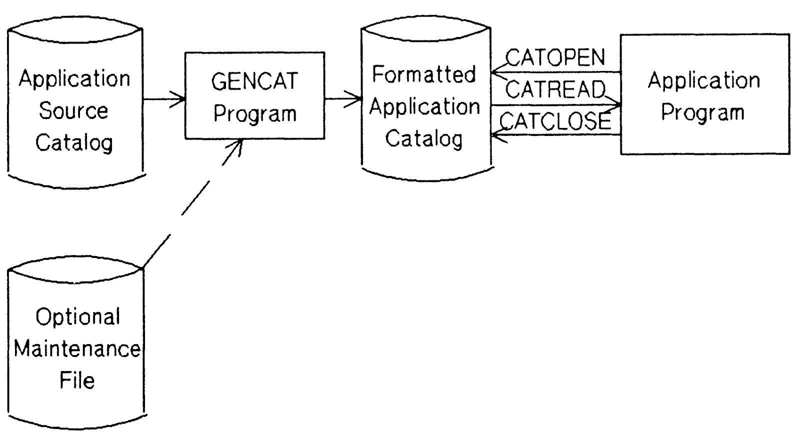


Figure 3. The Application Message Facility

The new message facility features the following:

1. Up to five parameters in a message may be specified either by position or by number. I.e., they don't have to appear in the message in the same order for every language.

2. The GENCAT utility program transforms a standard source file into a formatted version, with a directory (for efficient access) and compression of data (for efficient storage).

3. GENCAT has the facility to merge a master and a maintenance file for a message catalog. This allows the localizer to easily incorporate changes to the original language catalog into the local version.

4. The CAT- intrinsics are easily callable from all of the major programming languages.

Native Language Support in the Subsystems

In addition to the MPE utilities and intrinsics, HP3000 NLS consists of capabilities in the following subsystem products: KSAM, SORT-MERGE, VPLUS, FCOPY, IMAGE, QUERY, and COBOL II. As with the MPE facilities, the features in these products are intended to provide application designers and programmers with the tools needed to design local language applications. Thus, it is the application end user, and not the programmer or subsystem user, who sees the localized interface.

The NLS features in HP3000 subsystems are, for the most part, implemented through the capabilities provided by the MPE NLS intrinsics, or internal variants of these intrinsics. Because of this, aspects of a native language definition within MPE will be consistently reflected across all of the subsystems. Perhaps the best example of this is collating. The collating sequence defined for a particular native language is directly usable by application programs through the NLCOLLATE and NLKEYCOMPARE intrinsics. The same collating sequence is used by SORT-MERGE in ordering records, by KSAM in ordering keys, and by IMAGE in ordering sorted chains, when these products are dealing with character strings that have been associated with the same native language.

The NLS features of each subsystem must be explicitly requested to be invoked. For backward compatibility, the subsystem will operate in an unchanged fashion unless NLS functions are explicitly requested by the application program or interactive subsystem user. This is intended to preserve the validity of existing (pre-NLS) applications, including programs, stream files, etc. With NLS, you don't get it unless you ask for it.

NLS features are "asked for" in several different ways, among them, the following:

1. New intrinsics. Particularly in MPE, new intrinsics have been introduced to provide language dependent services. Thus, to get a local language date format, an application should call the new NLFMTDATE intrinsic instead of the old FMTDATE intrinsic (which is unchanged).

2. New parameter values in old intrinsics. In SORT-MERGE, for example, the SORTINIT intrinsic has been extended to allow the specification of a new key type "character", and of a native language which determines the collating sequence to be used. To get native language sorting, an application must use these new parameter values; existing SORTINIT calls will continue to result in current (NATIVE-3000) sorting.

3. A native language "attribute". Various permanent subsystem structures, including IMAGE data bases, KSAM files, and VPLUS forms files may be assigned a native language attribute at the time they are created. This language attribute then causes these subsystems, at run time, to perform some of their functions in a localized fashion. VPLUS, for example, will perform its upshift function according to the language of the forms file. If the designer or creator of a particular structure has not specified a native language attribute for it, the attribute will default to NATIVE3000, and the subsystem will behave as before.

4. Explicit commands. Certain of the subsystems which provide NLS capabilities include, or consist entirely of, interactive utility programs. Examples of such programs are FCOPY and QUERY. (Remember that the user interfaces of these programs have <u>not</u> been localized, but some of their functions have been made native language dependent.) Commands (or keywords) have been added within these programs which allow the user to specify that their functions should be performed relative to a particular native language. For example, entering "LANGUAGE=FRENCH" within QUERY would cause any sorted character data (IMAGE types X and U) in its output reports to be sorted according to the French collating sequence. If no such command is entered, QUERY or any other subsystem will behave as before.

The specific details of using the NLS features in subsystems and the effects of doing so are provided in the NLS reference manual. Here, we present a brief idea of the major capabilities:

KSAM -- if a language attribute has been assigned to a KSAM file (through FOPEN or using KSAMUTIL), that language's collating sequence is used in ordering any keys of type BYTE. (Generic key retrieval on such keys must be performed with the help of the NLKEYCOMPARE intrinsic.)

SORT-MERGE -- a new key type, CHARACTER, is introduced; keys of this type are sorted or merged according to a native language collating sequence, if one is specified. This capability is available programmatically, and from the stand alone SORT and MERGE utilities.

VPLUS -- several run-time VPLUS functions are native language dependent: date formatting, numeric formatting, character upshifting, and range checking. These functions are performed relative to the rules of a native language, if one has been associated with the forms file in FORMSPEC. It is generally expected that a VPLUS application program which has been localized into several languages will have a separate forms file for each language; each forms file will have a unique language attribute, as well as translated forms text. (This is similar to the scheme of having a separate application message catalog for each language.) Alternatively, VPLUS allows a forms file to be "international"; at run time, the application program must call a new intrinsic called VSETLANG to establish the native language to be used in processing the forms.

FCOPY -- a new "LANG=" parameter influences three functions: upshifting, conversions to and from EBCDIC, and display of characters (the CHAR option). Upshifting and conversions will follow the appropriate native language definition; FCOPY will use the corresponding character set definition to decide which characters are printable, and which must be replaced by ".".

IMAGE -- if a language attribute has been assigned to a data base (in the schema or using DBUTIL), internal comparisons of type U and X data items will use the corresponding language's collating sequence. This specifically affects two areas: the way sorted chains are ordered by DBPUT and the way concurrent entry level lock requests are compared in DBLOCK.

QUERY -- if a language is specified by the user, several functions of QUERY are affected. For type U data items, upshifting of user entered data will follow native language rules; value comparisons for type U and X items in FIND and other retrieval commands use the appropriate collating sequence. In output reports, sorting of type U and X data items and formatting of dates and numeric data follow the appropriate language definition. The language chosen by the QUERY user is independent of the language (if any) of the underlying IMAGE data base(s).

COBOL II -- the syntax and compiler have been enhanced to give the COBOL programmer access to the native language capabilities of SORT-MERGE; this is accomplished in COBOL using extensions to the COLLATING SEQUENCE phrase of the SORT and MERGE verbs. Although in this release only COBOL has specific enhancements for NLS, several other programming languages can also <u>use</u> NLS to produce localizable programs.

A Multilingual Application

Now let us return to our imaginary example application, HPRoom. If this application has been written to take advantage of NLS features and hence to be easy to localize, here's what we would observe in examining the application and its environment, especially its source code.

1. There are no hard-coded messages in the source; the Application Message Facility has been used for localizable text. I.e.,

   a. The GENCAT program is used to create the message catalog.

   b. A naming convention has been used to distinguish the different native language versions of the message catalog.

   c. At the beginning of the program a call is made to the NLGETLANG intrinsic to determine the interface language of the user, and hence which message catalog to open with the CATOPEN intrinsic.

   d. When error messages are to be displayed, the program fetches them into a buffer with the CATREAD intrinsic (including insertion of variable parameters) and then puts them into the VPLUS window.

   e. The CATCLOSE intrinsic is used to close the message catalog at the end.

   f. Future message catalog enhancements are provided as a maintenance file (i.e., a file containing only those messages which are new or changed). Thus the localizer can easily merge them with GENCAT into a previous version which has already been translated.

2. The user interface language is used to determine which of various VPLUS forms file to open so that text portions and soft key labels for each form are in the language of the user. The VPLUS forms files have been created with a "forms file language" appropriate to the language of the text in that file, so that any processing specifications are also carried out according to the rules of that language. (Alternatively, one could create a single "international" VPLUS forms file with all text areas as display only fields to be filled at run time from a message catalog. In this case, the language for processing data is determined by calling the new "VSETLANG" intrinsic).

3. The date which is displayed at the top of the room reservation screen is formatted from an internal form by calling the NLFMTCALENDAR intrinsic.

4. The from and to times of the reservation are tested for validity and converted to an internal form by calling the NLCONVCLOCK intrinsic.

5. The name of another conference room which the user enters in the lower left box is upshifted either by VPLUS itself, using the "UPSHIFT" processing specification on this field, or by calling the NLSCANMOVE intrinsic and passing it the input string.

6. A sorted list of conference rooms is produced by, e.g., doing its own in memory sort using the NLCOLLATE intrinsic to compare each pair of conference room names.

An application which has been designed in this way is referred to as a multilingual application because it can be used in different languages without redesign, recoding, or even recompilation.

An example of what the localizer can do with such an application is shown in Figure 4. The reservation screen of HPRoom is displayed again, as seen by a German user. A localizer has created a German version of the VPLUS forms file by translating the text and soft key labels and associating a language attribute of GERMAN with the file. The forms file has then been compiled and given a name known to the program, based on the fact that it is the German forms file. (A naming convention for such language-dependent files is supported within NLS, taking advantage of the unique numbers which are associated with supported languages. See the Native Language Support reference manual.)

The localizer has also created a German version of the application message catalog by translating the source and processing it with the GENCAT utility. The resulting message catalog has also been given a unique name.

The application is now ready for a German user, Günter Klenk, to use in reserving an hour in the Nürnberg conference room for a French class.

```
                                   HPROOM
                    BELEGUNGSPROGRAMM FÜR KONFERENZRÄUME
 --------------------------------------------------------------------

                       Mi., 26. Dez, 1984

                       6  7  8  9 10 11 12 13 14 15 16 17 18
   NÜRNBERG                    XXXXXX XXXXXX    XXXX      XXXXXX

 NAME             TELEFON        VON:        BIS:        BEMERKUNGEN

 [Günter Klenk  ] [8888     ]  [14:30 ]    [15:30 ]    [français     ]

 Um einen anderen Raum/Tag zu reservieren, TAB drücken und folgendes eingeben:

   NÄCHSTER RAUM: [          ]     NÄCHSTES DATUM: [          ]

 ........ ........ ........ ........    ........ ........ ........ ........
                     AUS                                           PROGRAMM
                   DRUCKEN                                 ZURÜCK  AUSGANG
 ........ ........ ........ ........    ........ ........ ........ ........
```

Figure 4. The German HPRoom reservation screen.

Conclusion

HP3000 Native Language Support is a collection of facilities in MPE and its subsystems which allow the design and implementation of localizable end user application programs. "Localizability" is achieved by separating from the program code those aspects of the application which will vary from one language to another, or from one country to another. Such aspects include not only messages and prompts, but also date, time, and numeric formats, and various internal functions such as upshifting, character comparisons and sorting, scanning, etc.

HP3000 NLS is based on the concept of "supported languages". The attributes of each such "language" are maintained internally by NLS and are accessible via intrinsic calls. These intrinsics may be used directly from application programs, and are used also by several HP subsystems. In addition, NLS contains an efficient catalog facility for the convenient handling of local language application messages.

The representation of native language character data is dependent on HP standard extended character sets. These sets allow for the unambiguous representation of multilingual data, without loss of standard USASCII characters. Most HP terminals and printers supported on the 3000 provide these character sets.

NLS facilities have been introduced into HP3000 software in a way which provides backward compatibility. The application designer/programmer and interactive subsystem user decide which, if any, NLS features to use, as well as what language controls each function.

Biographies

Harry Kellogg
is a Development Engineer in the operating systems lab of the Computer Systems Division of Hewlett-Packard. Harry was a member of the international project team which developed native language support for the HP 3000. He continues to be active in developing new features in this area. Harry has been with HP for 8 years, including 4 in Germany, in a variety of software R&D and internal application areas.

Jon Bale
is Computer Internationalization Manager in the Information Systems Group of Hewlett-Packard. Prior to this assignment, he was the Project Manager for the HP 3000 Native Language Support project at Computer Systems Division. An employee of HP for 15 years, Jon was a member of the original IMAGE Data Base design and implementation team. He has also lived and worked in Europe for 3 years, with assignments in the Netherlands and England.

SUMMARY

GETTING THE MOST FROM TRANSACT/3000

BY LARRY KEMP, HP BELLEVUE, WA

Transact/3000 is the largest selling add-on software product offered by Hewlett-Packard on the HP3000 series of computers. Transact/3000 is a procedural fourth generation language for creation of database oriented programs. Successful users of Transact/3000 have reported significant reductions in application development time over traditional techniques. Unlike report-writer type fourth generation tools, Transact/3000 can be used for the development of entire application systems.

In spite of the successes, Transact/3000 has had mixed reviews. The file access, data element structure, and transaction orientation of Transact/3000 requires a certain programming style to take advantage of it effectively. The extent to which users have had successes is directly related to how well they have been able to recognize and adapt to this style.

This paper attempts to identify what that style is and how that style can be used to effectively shorten program development time. This paper introduces item level constructs, hierarchial file access, step wise development and other programming techniques. This paper assumes the reader is familiar with Transact.

Getting the Most from Transact/3000
By Larry Kemp, HP Bellevue,WA

Transact/3000 represents a substantial step forward in program and systems development. One of the biggest hurdles to get over is unlearning COBOL or FORTRAN language techniques. Transact/3000 is classified as a procedural language, giving you the ability to solve any problem that could be solved with conventional techniques. But Transact/3000 adds some new dimensions to program development:

(I) In Transact/3000 you have the ability to work at the true data element level. This is in contrast to COBOL record structures. In particular, in COBOL you might read data from a screen into a record, then manually move that data to an output record, and then write out the record. In Transact you define the individual data elements and issue a read to the screen and a write to a database. Transact takes care of formatting the records for input and output.

(II) Transact has the ability to process data at the file (or dataset) level, in addition to the individual record level. This eliminates the coding of loops, error checks, and file positioning. For instance, to delete all records in a detail chain you code a DELETE(CHAIN). This operation performs the find of the chain head, sets up an internal loop, and finds and deletes all entries in a chain, and handles any abnormal errors. To display each deleted record, add in a PERFORM option with a DISPLAY. This is a markedly different style of programming than individual record oriented COBOL programming.

(III) With the automatic error-recovery in Transact, and the fast compiler, true prototyping and stepwise development are possible. Rather than write the entire application and then test it, you can develop one part and test it, then go on to the next part, and so forth.

With these ideas in mind, the objective I would like to pursue practices in Transact that emphasize those features while also optimizing performance. I assume that the reader has a knowledge of Transact facilities.

Use high-level constructs.

Probably the best guideline in Transact is that if a high-level construct can be applied, then use it. If a problem is tedious to solve, then a high-level is construct probably exists that will solve the problem.

The reason is that Transact is an interpreter. The interpreter overhead in executing a DELETE(SERIAL) is relatively minimal, while the interpreter overhead in executing a MOVE is relatively high.

Another way of viewing this problem is that if equivalent programs can be written in 100 source lines versus 200 source lines, that the shorter program will probably execute faster; not to mention be easier to debug.

Two facilities to take advantage of are (1) LIST register manipulation and item referencing (to eliminate MOVEs), (2) LIST specification in file references, and (3) elimination of loop code by using multi-access verbs (e.g. FIND versus GET).

Keep the List Register Short.

The LIST register in Transact is probably the most often misunderstood facility to new users. The LIST register is a dynamic storage area for storing data. The LIST register allows you to define only that data that is necessary to the currently executing transaction. This is in contrast to the COBOL construct that would have you specify all of the data for all transactions within one source program. Specification of only the data items necessary at each point in time makes program debugging easier.

There are also performance benefits to keeping the LIST register short. When Transact references a data item, it must search the LIST register to locate the current definition. Keeping a short LIST register makes the search faster. When Transact searches the LIST register, it starts from the top (most recent entry) and searches sequentially until it finds the specified items. A corollary then, is to also keep your most frequently referenced items near the top of the list.

LIST register expansion and contraction are low overhead operations Transact. I suggest use of meaningful marker items to set list contraction points, for example END-OF-LIST or END-OF-GLOBAL. As a last point, be careful of overusing the dynamic LIST constructs, which may complicate debugging.

Avoid using the STATUS modifier in Data Access Verbs.

Use the multiple-record access facility. For example, to read out a file use the FIND operation across the entire file with a PERFORM clause to process the individual records; rather than code a loop with FIND/STATUS verbs.

Most operations can be coded without specifying the STATUS modifier. Use of the STATUS modifier reduces Transact to COBOL style coding: It suppresses the automatic positioning, automatic error handling, and built-in loop constructs.

When coding without specifying the STATUS modifier, the STATUS register returns the number of records that satisfied the selection criteria. To check for existance of a record then, issue a FIND/SINGLE and test for the STATUS equal to one. Any abnormal errors (such as broken chain, etc.) will invoke the automatic error handling facility.

If you wish to override the automatic error handling, use an ERROR= clause. This will allow you to handle the "abnormal" error conditions.

Use either VPLUS or character mode.

Transact works in VPLUS mode and character mode. Choose one option or the other; do not mix the modes. Each mode switch causes a VOPENTERM or VCLOSETERM intrinsic to be executed. These are time and resource consuming.

Declare all databases and formfiles in the main program.

When using subprograms, open all databases and formfiles in the main program. This causes the subprograms to share the opens of the main, rather than invoking their own opens.

Explicitly declare VPLUS forms.

Transact maintains a runtime table of all items that you have declared. To minimize that table size you should minimize the number of elements that you declare. In particular, when declaring VPLS=formfile with the forms not specified, Trancomp will implicitly load form and element definitions for the entire formfile. Explicitly naming the forms in the VPLS= declaration will narrow down which forms Trancomp loads.

A note: if a Transact program requires 255 data elements or more, then it uses double word (rather than single word) addressing; there is a benefit to keeping the number of data elements to less than 255.

Use the appropriate type of segmentation.

Transact offers two varieties of segmentation: the CALL (subprogram) facility, and the !SEGMENT (segmentation) facility.

The CALL facility is intended to allow two or more people to work on independant parts of the same application system. It allows each part to be written and tested independantly, and then later integrated via the CALL verb.

At runtime, the CALL verb executes an FOPEN/FCLOSE each time the CALL verb is interpreted. This can generate excessive overhead if overused. In an appropriately segmented system, the user would enter a subprogram, and stay in that subprogram for a considerable time frame. For example, a tax calculation routine is not an appropriate subprogram, but an accounts payable routine is an appropriate subprogram.

To reduce the size of programs. use the "segmentation" facility. A properly segmented system that uses !SEGMENT constructs will contain at least three segments. The first segment is called the root and will contain all global item definitions and all commonly used routines. The other segments will be brought in as needed from disc. As with any segmentation technique, the optimal segmentation strategy will be to stay in a segment as long as possible. A jump across a segment boundarie is significantly lower overhead than execution of a CALL verb.

Use the "item" construct.

This is in contrast to record constructs. When the item names between the database and forms all correspond, Transact will do all of the reformatting of data records. Well written programs will have very few MOVE verbs. Transact knows how to convert data for display and/or forms processing using the display size.

Not only will this eliminate the coding effort, but this will also increase the execution efficiency of the program.

Use packed decimal datatypes.

For most operations, Transact does packed decimal arithmetic. If all operands are of type integer, then Transact will do integer arithmetic. In case of any type mixing, Transact converts to packed decimal. You can save data type conversion time by generally using packed decimal.

Examples:

```
LET (item)=1;                          CLOSE dset;
SET(MATCH)LIST(item);            LOOP: FIND(SERIAL)dset,STATUS;
OUTPUT(SERIAL)dset;                    IF STATUS=-1 THEN
                                         GO TO DONE;
                                       IF STATUS<>0 THEN
                                         GO TO ERROR;
                                       IF (item)=1 THEN
                                         DISPLAY;
                                       GO TO LOOP;
```

Example #1: Use of high level coding vs. conventional techniques.

```
LIST a:b:c:END-OF-GLOBAL;          LIST a:b:c:
GET(FORM)formname1;                     x1:x2:x3:x4:xn:
      .                                 y1:y2:y3:y4:yn;
      .                            GET(FORM)formname1;
      .                                  .
LIST x1:x2:x3:x4:xn;                     .
GET(FORM)formname2;                GET(FORM)formname2;
      .                                  .
      .                                  .
      .                                  .
      .
SET(STACK)LIST(END-OF-GLOBAL);
LIST y1:y2:y3:y4:yn;
GET(FORM)formname3;                GET(FORM)formname3;
      .                                  .
      .                                  .
      .                                  .
```

Example #2: Keeping the LIST register short vs. declaring all data items.

```
LIST a:b:c;                        LIST a:b:c:
GET(FORM)formname;                      xa:xb:
PUT dset1,LIST=(a:b);                   yb:yc;
PUT dset2,LIST=(b:c);              GET(FORM)formname;
                                   MOVE (xa)=(a);
                                   MOVE (xb)=(b);
                                   PUT dset1,list=(xa:xb);
                                   MOVE (yb)=(b);
                                   MOVE (yc)=(c);
                                   PUT dset2,list=(yb:yc);
```

Example #3: Using the item concept vs. record structures.

BIOGRAPHY

Larry Kemp is a System Specialist at the Hewlett-Packard sales office in Bellevue, WA. He has spent the last five years specializing in databases and productivity tools usage. Larry earned an MS degree in computer science from the University of Oregon 1975.

PPL/3000 - A USER INTERFACE PROGRAMMING LANGUAGE

Kim Harris
Handsworth Programming Services, London, UK

Martti Laiho
Oy Porasto Ab, Helsinki, Finland

Summary

This presentation introduces a "shell-like" user interface programming language "PPL/3000" based on string processing. The language was originally developed for interactive pre-processing of batch jobs as the command language of the JOBLIB/3000 product. Now the same pre-processing technique can also be applied to a series of on-line tasks. The capability of calling intrinsics and user-written subroutines makes the language easily extendable.

The use of MPE-commands, parameterized utilities and application programs, graphics, remote sessions, optionally mixed with database accesses, etc, can be invisible to the end user. The user interface is programmable as menus and dialogues in any national language.

PPL/3000 language

The history of PPL/3000 language dates back to 1977 when the first version of the JOBLIB/3000 system was written at Oy Porasto Ab. The main purpose of the JOBLIB/3000 system has been to provide computer aided batch job preparation by password and parameter substitution and interactive selection techniques as described in earlier reports /1/. The preprocessing of batch jobs is still one of the main functions of the system, which has resently been extended with the interface to HPS SCHEDULER /5/ replacing SLEEPER as a more secure and fully supported job scheduling system.

The preprocessing technique used in JOBLIB/3000 is based on control commands like:

```
..DISP message
..READ parameter
```

which are inserted on separate control lines into the stream template. The parameter value read from terminal will be replaced in the generated stream file on any line where the variable name of the parameter has been written immediately after the substitution operator "&". For example

```
!FILE OUTPUT=&parameter
```

The parameters of the command language are called string variables like the data type "string" in Pascal dialects and in Basic. These variables have only a name and value consisting of a variable length string of any ASCII characters. In the technology of today the interface between man and most commercial computer applications is mainly based on character strings, and there are much that can be done by processing them. If the string values are numeric then the variables can also be used in arithmetic operations.

In the JOBLIB/3000 command language string variables are used for parameter substitutions, calculations, command and procedure modifications, controlling of flow by structures such as IF, WHILE, and CASE, etc. Gradually the command language has been extended to allow almost any processing that can be done using programmming languages and we decided to call it "PPL/3000" - Pre-Processing Language.

As the professional HP3000 users in Finland became more used to the command language the more often we were also requested to make it available for pre-processing of online tasks. To solve the technical problems we started to experiment by executing the generated batch streams immediately after preprocessing using a tailored command interpreter program called JLEXEC. Resent versions of MPE have aided us by providing more powerful intrinsics to enter the external MPE commands internally from SPL, the programming language in which the JOBLIB/3000 system is written.

Using a mixed language of PPL and the external MPE commands with data lines used in stream files, we have now achieved the following capabilities proposed already by Lauesen in /4/ exceeding MPE and UDC commands:

- conditions,
- loops,
- parameters,
- procedures,

together with

- interactive prompts,
- structured control commands IF, WHILE, and CASE.

As the first and most general solution to requests of accessing MPE files and IMAGE databases from the PPL language, we implemented the free format CALL command with some new functions for data type conversions so that we could communicate with any MPE intrinsics and user-written SL modules. To make the intrinsic calls easier to use, we designed and tested a still growing general purpose procedure library PROCLIB written in PPL/3000 and containing self documenting procedures for interfaces with IMAGE, VPLUS, MPE file intrinsics, and utilities accessing JCW variables, etc. The implementation of CALL has had a major impact on the future development of the PPL/3000 language. Today it is a totally open-ended, interpretive programming language, dedicated to user interface programming.

Procedural MENU systems

After the valuable experiences gained with JLEXEC in preprocessing on-line tasks we started to design special procedures just for on-line use. They soon took the form of a "procedural menu". The following procedure is an example of a typical character mode menu structure used in PPL-applications. It can be gategorized as a Cluster Manager system /6/ providing a uniform user interface to integration of standalone software.

```
..proc RWmenu
..option rep=off,<< + other configuring options >>
..string select  << a new variable, set to empty >>
..while select <> "8"
..    disp ***** A Researcher's Workbench MENU *****
..    disp Enter the code of the next task:
..    disp 1 = My Favourite Editor
..    disp 2 = DAISY/3000 Text formatting
..    disp 3 = STATS/3000 Statistical analyses
..    disp 4 = Graphics
..    disp 8 = exit (default)
..    read "Enter task code"; select="8"; (1,2,3,4,8)
..    case select        << branch according to the value >>
..    ="1"
!RUN myeditor
!COMMENT         << stdin and stdlist directed to terminal >>
..    ="2"
..         read "Enter name of textfile: "; textf
!RUN DAISY.Z.ATK
OUT-&user,DIABLO  << 'user' is a system variable >>
IN-&textf         << textfile name will be replaced here >>
EXIT              << exit DAISY >>
..    ="3"
..         invoke STmenu in STATSlib.PUB.STATS
..    ="4"
..         invoke DSGmenu in GRAPHlib.LIB.SYS
..*
..         eoc  << end of case select >>
!COMMENT        << end of while-loop: >>
..    eow
..eop           << end of procedure RWmenu >>
```

The functions of different PPL commands (or statements) used in our example should be obvious to any professional with some experience of high-level programming languages. Therefore we will not go into the details here. The purpose of the example above is just to give a general idea of the use of different "software layers" in same templates:

- PPL-commands
- MPE-commands
- parameterized data and subsystem commands.

The menu procedures can be extended into a Workstation Management systems using character mode escape sequences to control

- devices
- windows, workspaces, softkeys, touch map, etc.

The concept of "mixed software layers" brings new possibilities to integrate and link existing software in a new synergic way. For example, via intrinsic CALLs to DICTIONARY it is possible to make use of definitions for ELEMENTS, file LOCATIONs, etc.

A typical application of linking different software with PPL is to fetch parameters of a QUERY report using VPLUS form and its field processing specifications:

```
..proc QRYREPOR (base, psw, ...)
..string sdate                << search date >>
..set    sdate="&yy&mm.01"    << set default >>
..invoke vopentrm (vcom,vcom2,termf,vstat,vtermtype)
..invoke vprocfrm (vcom,vcom2,"MYFORM",sdate)
..*      This is a combound procedure which fetches the form,
..*      sets default, shows form, reads fields and buffer.
..invoke vclostrm (vcom,vcom2,vstat)
!FILE QSLIST;DEV=&ldev
!RUN QUERY.PUB.SYS; STDLIST=$NULL
DEF
&base          << database >>
&psw           << password >>
1              << mode >>
MYPROC         << QUERY proc file >>
LP             << report to QSLIST instead of STDLIST >>
FIND DATE >= &sdate
REPORT MYREP   << defined in MYPROC >>
EXIT
..eop
```

More flexible systems can be created if we define and parameterize the report in the PPL procedure itself or collect the more complex report with alternative headers or detail lines from different PPL procedures.

Even more dynamic reporting systems can be created with ASK/3000, which is an open-ended superset of QUERY. With ASK it is possible to write concurrent PPL access to VPLUS etc, forward preprocessed commands to ASK and proceed according to ASK's responses.

Systems development work itself, programming, testing, tuning system is a typical application area of tools written in PPL. Some examples are database modifications, mass-changes of database contents, etc. As an independent software layer that can be mixed with any languages, PPL brings the power of interactive macro systems for professional users practically without limitations. It is the ultimate language which can be used to link other software to work in synergy and to write new tools mixing existing tools, intrinsics and string processing. The following ideas of Kernighan and Plauger apply also to the use of PPL:

- "Whenever possible we will build more complicated programs up from the simpler; whenever possible we will avoid building at all, by finding new uses for existing tools, singly or in combinations. Our programs work together;

their cumulative effect is much greater than you could get from a similar collection of programs that you couldn't easily connect." /2b, p 2/

- ".. Learning to think in terms of tools will encourage you to write programs that solve the unique parts of your problem, then interface to existing programs to do the rest." /2b, p 3/

Typical existing tools in MPE environment are line editors, QUERY, FCOPY, SORT, and some "filter" programs of our own.

Comparison of MPE/PPL with UNIX/Shell

The command interpreter Shell of the UNIX system can be regarded both as a programming language and a command language providing a user interface to the UNIX operating system.

So far we have been able to compare PPL and Shell only basing our evalution of Shell on reference /6/, but it seems to be in the common future of all HP3000 users in 3-10 years.

In comparing the languages of the systems we have found many common features though the operating system environments and the "software cultures" are different. The goals of the systems differ in the following ways:

- Shell has been designed to be used both interactively, command by command, by skilled professionals as well as driven automatically by procedure files (scripts). To make the typing of commands easier for the professional user, the language is "short-hand oriented" and tends not to have extensive expression syntax /6/.

- PPL is basically procedure file driven. The commands are written using some editor system only once for every procedure and therefore the expressions and command structures can be longer and more like programming languages. The PPL language is used by professionals who program interfaces for end users or for themselves, and from the programming point of view we continue improving the syntax and the mnemonics of the language according to programming languages.

- Both languages are based on string-valued variables and have control flow stuctures for branching and iteration.

- Logical tests in Shell are based on programs and the operating system, while PPL is more a standalone and independent programming language including arithmetic and logical expressions of its own. The environment can be controlled by JCW variables and condition code variable CC. In addition to use of programs also instrinsics and user written modules can be utilized in PPL.

- In Shell it is very easy to link existing programs into chains communicating via pipes. In PPL this requires more efforts and explicit spanning of message files.

- Due to automatic "rescan" from the beginning of every substituted string value, PPL is a very dynamic interpreted language which can load contents of commands even from databases. The speculation on possibilities allowed by this feature is left to the reader.

References:

/1a/ "JOBLIB - Generalized Job Library and Interactive Job Generating System"
Martti Laiho, HPGSUG Fall Meeting 1980

/1b/ "JOBLIB/3000 - an Interactive Pre-Processing System"
Martti Laiho, HPGSUG Fall Meeting 1981

/1c/ JOBLIB/3000 Release 6.0 Reference Manual, 1984

/2a/ "Software Tools", 1976

/2b/ "Software Tools in Pascal", 1981
Kernighan, Plauger

/3/ "The UNIX System"
Bourne

/4/ "Program Control of Operating Systems"
Soeren Lauesen, BIT 13, 1973, 323-337

/5/ HPS SCHEDULER Reference Manual
HPS, 1984

/6/ "The Outlook for Integrated Software"
Brant Coreson, DATAMATION March 1984, p 225-9..

Trademarks:

UNIX is a registered trademark of Bell Laboratories.
JOBLIB/3000 is a registered trademark of Oy Porasto Ab.

Biography:

Kim Harris
has been in data processing since 1976. After spending 3 years in operation at Empirial College of Science and Technologies on CDC mainframe he joined London Business School in 1978 as operation supervisor, where he and Clive Oldfield developed SCHEDULER. Kim is now a partner in HPS and an Associate with a London firm of Charter Surveyors, where he is the manager of computer systems and software development.

Martti Laiho
is the R&D manager at Oy Porasto Ab. He is the co-author with Lauri Oravisto of the JOBLIB/3000 system. He has been in data processing since 1968 and a HP3000 user since 1977.

An integrated approach to software engineering;
experiences within the ASSYST-RAET GROUP

Hans van der Leeuw

Summary

In this lecture I hope to clarify the following points:

a. the state of the art with respect to the application of integrated tools for software development;
b. starting point of the ASSYST-RAET GROUP for the development of products and services with respect to software development tools;
c. the products available now.

Hence my lecture will be focused on:

- the use of software development methods for the different stages of a development project;
- the use of available tools;
- the necessity of integration of these tools.

Why methods?

I expect that everyone working with computers (from home computer user up to business analyst) has been confronted with the question: Is there a structured approach to the problem to be solved?

In the past the ASSYST-RAET GROUP has also been confronted with this question.

In our search to find the ultimate answer we also had to solve lots of other problems.

The use of a method for only software development was not sufficient to us. A solution had to be found for questions such as improvement of the communication with the end user, project progress and control, method of documentation and making software understandable and maintainable.

In addition, the method should not only cover all the steps of a computerization project but also the implementation and maintenance stage.

From 1968 we have succesfully been using our PARAET method.

In cooperation with a very active users° club (35 members) many improvements have been achieved in the recent years.

Contacts with big international firms of auditors also revealed that the PARAET method provided an insufficient solution for the link between feasibility study and information plan.

During discussions in 1979 it came to the fore that already in this stage of a project there could be misunderstandings between a user and the EDP professional.

Therefore methods are necessary for:

- a systematic approach;
- explaining to the user what the result of the mutual effort will be.

Why tools?

In the foregoing I have shown you the objectives of a method. The following step is to give consideration to the use of tools to support a method, which may involve the additional advantage of productivity improvement.
Properly speaking, the computerization process consists of a number of identical stages, each of them being used in an iterative way.
A number of record lay-outs, screen lay-outs, reports and processes are made. Doing all these things manually would be a very tedious job. The risk of errors with complex systems (e.g. if more than 10 man-years of labour is involved) increases exponentially.

The first development tools were directed towards one of the stages of a computerization project.
First there were the report generators, then the central information dictionaries and at last the enquiry languages.
Each of them brought a significant improvement, but only part of the whole computerization project was solved and the objective for using methods, viz. improvement of the communication between the end user and the DP professional, enforcing the user to follow a systematic approach, was not yet realised.

Considering the demands made on a method, it may be concluded that for the application of a tool there are a number of additional reasons:
1. simplification of the activities of an analyst-programmer to achieve a consistent way of working with less chance of errors;
2. better control of project progress by structuring the activities and making them measurable with respect to quality and quantity;
3. in line with the improvement in communication between DP professional and end user, the user°s wish to be confronted with the final result at an earlier stage of the project.

Requirements for an integrated approach

The choice of the word "integrated" already implies that a solution must cover all the stages of a computerization project.
This must include the choice of one or more aids.
If different aids are used, they should be able to communicate with each other with respect to the way of working, and should be linkable with respect to the structure of data.
A good aid will not only describe the "what-situation" but also how it will be solved.

The choice of one tool for the whole computerization project implies that the DP department as well as the business planning department should accept its use. For practical considerations the choice may fall on more aids.
Another aspect involved is the flexibility of application.
Within an existing organization some standards will already be in use. Also know-how will already be available.
Under the given economic circumstances no organization can afford a capital drain. The tool to be chosen must be so flexible that it can be used in conjunction with existing techniques.

An important role is also played by educational aspects.
Is it possible for the organization to continue on the basis of existing knowledge or is it necessary to adopt a different way of working and thinking?
A tool should also be able to support complex as well as simple organization and computerization projects. Besides a number of major information streams necessary for the operation of an organization, there are different applications which are important in themselves but which are not crucial for the operation as a whole.

Quality improvement of the computerization project has two aspects.

1. the needs of the end user must be translated correctly into automated procedures;
2. the maintainability of systems must be improved.

Owing to the backlog which already exists in the development of systems, productivity improvement must be realised.
This can partially be achieved by a a consistent way of working in different stages, for the output of previous stages can be used as input in subsequent stages.
For the system development stage consideration may be given to the application of fourth generation software development aids.
To set up a business plan a tool must support management decisions.

Last but not least, an important requirement is hardware independence.
This applies to the use of the tool as well as to the application of the operational procedures.
I do not exclude the possibility that the tool will work on suitable hardware which may differ from the target system. Both have there own demands.
In this context I can point to the ongoing penetration of microcomputers within organizations.

The ASSYST-RAET basic model of a computerization project

The policy of the ASSYST-RAET GROUP is to base the formulation and project-oriented implementation of computerization projects on the development model shown here.
The model gives a classification of the whole development stage into:

1. Business information systems directed towards the strategic plans of the business concerned. This will always be derived from the business strategy of our client. It is the stage where the corporate strategy is translated into to an information plan and computerization plan.
   The information plan will give a description of all the information streams necessary for the business goals.
   The computerization plan selects those information streams which are suitable for computerization.
2. Information plan for selected information streams.
   This includes all the measures which will be taken by the board of management to automate information streams, in the field of techchnical structure, quantity and quality of computerization staff and the place of the EDP department within the organization.

3. Realization of the computerization policy on a project basis.
   Since all computerization project must have a clearly defined beginning and end, a project-oriented approach must be possible. Also within the whole development stage clear milestones can be recognized. This applies both to the system analysis and to the system development stage.
4. Implementation and management of operational systems.
   Management of an operational system consists of a correct use of the systems as well as to the introduction of minor modifications based on changing circumstances. By the use of systems I mean the availability of the facilities (for example hardware).

Each of the above mentioned points requires its own approach with regard to the control of the different stages of the development process. The same applies to the methods and tools chosen. In our own organization we have come to the conclusion that a consistent approach of all these aspects will lead to an improvement of the level of information for our clients.

I will now give you an overview of all the aids we use for controlling methods and tools at our company.

1. Business plans
   One of the methods we use is called Prisma, developed by Klynveld Kraayenhof & Co. It is based on the Business System Plan method of IBM and makes use of business models to test the information model of the company we have to investigate.
   This method is supported by a development tool called InDD, to which I will revert later on.
2. Computerization
   Also in this stage InDD is in use.

3.a Functional realization
   For this stage we use the Michael Jackson Development method, called JSD.
   For designing the dialogue we use a dialog simulator with which one can build a complete dialogue with a microcomputer, without having a data dictionary at one°s disposal.
   This simulation can take place at the user°s desk. At this point of time we can also use ARTESSA/3000.
   This subject will also be discussed further on.

3.b Technical realization
   For programming we can use the application generator ARTESSA/3000.
   For network analysis our NOM method is available.
   Available tools for this stage are ARTESSA/3000, JSP Cobol and transaction analysis, which translate the input of NOM into the proposal for the configuration of a network.

4. Production
   The ASSYST-RAET GROUP has written for its clients a manual for the management of a data centre called "Manual Computer Facilities".
   This manual is based on the procedures we use in our own data centre.

In summary I can state that the ASSYST-RAET GROUP strives for integration between these different tools by syntactic and semantic coupling.

For the management of the four stages mentioned we use the project management techniques which are described in our PARAET manual.

As you see, the application of the information model theory is a reality for our company.

The place of the information data dictionary/directory tool

The tool supports the preparation of the business and information plan and directs the information system developments.
The latter is possible because the tool works with information which is gathered during the information analysis stage.
By relating the different information, a model of the business organization is obtained.
Other aspects like reorganization of the various departments and/or places of work, education and instruction, data centre and equipment, offfice automation, and conversion are left out of consideration here.

Basic model

In a model of a business organization the relations between information are reproduced. As building stones of a model there are four types of elements.
The information needs and the information distribution in a business organization are determined by the activities which take place inside the business organization. The activities of a business organization constitute the first element of the model.

For a general purpose model, business activities are of special interest, independent of the specific organizational structure.
The structure of the model is determined by activities which are based on targets and grouped as functional units (departments).
Within these functional units the structure is determined by the nature of the function to be fulfilled by the activity.
The purpose of the activities can be split into three levels, viz. management, control, operations (abbreviation MCO).
The following step is to establish the information needs of an activity and what kind of information it should supply.
In this respect a distinction can be made into:
- information that triggers an activity, the second element;
- information that is contained in directives, the third element;
- information that is or will be stored in a data base, the fourth element.

The four elements mentioned and their mutual relations form the structure of the model.

The basic model gives a general overview of the activities and information streams in a business organization.
It can be used as a reference for the business analyst, when he has to investigate the information streams of a business.

On the basis of this model the actual situation in a company can be critically evaluated with regard to completeness and logic. Such evaluations can be taken into account in planning the improvements of the information streams and the ways and means to be allocated.
The implicit meaning of the development of a basic model is that it can serve as an aid in the formulation of the information policy and information plan, and the computerization policy and computerization plan of a business organization.

Structure of the model

The description and modelling of a business information system places the business analyst in a awkward dilemma between completeness and thoroughness. A model must give a complete picture of the whole business information system.
Completeness involves the risk of being too brief and introducing too high an abstraction level. Thoroughness means details and recognition of day-to-day operations. This again involves the risk that not all essential tasks are recognized or that tasks which are not essential are mentioned as important in the model.
Therefore, the model is based as much as possible on concrete tasks in order to avoid too much abstraction.
This can be achieved by opting both for the functional and for the process approach so as to guarantee completeness of the mutual information relations.

Functional approach

Every organization strives to reach an objective.
This call for the direction and organization of actions, the use of the right tools, registration and evaluation of what is happening and if necessary, adjustment.
So activities take place within the organization which have a relation with each other. These mututal relations form together the business system.
The business system as a whole has a target. It is easy to understand that there are a number of sub-targets which are derived from the main target. So all the activities can be grouped according to the contribution they make to a specific target. This is the functional approach.
In this functional approach the activities can be further distinguished according to management level. Within the functional units this gives a distinction into MCO-activities.

Process approach

The business system can also be examined by using the processes that work within the system. This approach provides an insight into the relation of activities which are needed to achieve the targets.
The functional approach shows what is to be achieved; the process approach shows how, the business system is to achieve its targets and what in- and output is needed or supplied by these activities.

Activating and reference information

A mutual relation between activities can be shown in information streams and related activities. The information consists of data about internal and external developments or is related to directives, rules, authority limitations, etc. So a distinction is made between activating information and reference information. In the model this information is tagged so that the contents of this information can be recognized.
On the basis of the foregoing the building stones can be described as follows:
- first element - the activity;
- second element - the activating information, the stimulus for an activity;
- third element - the reference information based on a directive;
- the fourth element - the reference information based on a data base.

These four elements together with their mutual relations form the overall structure of the model.

Concise description of the InDD tool

Background

Investigations have shown that incorrectly working information systems are for 60% due to incomplete or incorrect system specifications and that 80% of the total maintenance effort is spent on this category of errors. Hence, it is obvious to choose a method and preferably also a tool that meets the complete and correct system demands at a very early stage of the project.
The InDD tool has been developed for this purpose.

Basic model

Starting point is a standard business model extensively defined and stored in the computer, called the basic model. This model is divided into such functional units as Purchasing, Production, Sales, General Management, Finance and Administration, and Product Development. Within these units standard activities are defined, depending on type: Management, Control and Operations.
Activities, information carriers, data bases, directives and external triggers are defined per unit and for all units mutually related within the basic model.
The basic model aims to be a model of a business organization par excellence.
The tag of the basic model is named "as it should be". The model does not reflect the situation of the company to be investigated, but serves as a reference model.

Project start

Special attention should be paid to the circumstances under which information gathering will take place within a company. Questions like purpose of the investigation (information and computerization plan), scope of the investigation, explanation of the method, must be explained to the staff.
During the investigation the involvement of top-level management is absolutely necessary. It must motivate and give active support. It will be asked to make decisions as to whether bottlenecks encountered during the investigation should be eliminated or not.

Interviewing

A selected group of staff members will be interviewed by business analysts. The staff members will be asked to fill in standard questionnaires with support being provided by the business analyst. The staff members will suplly answers to the information needs necessary to perform their tasks. So answers to questions like:
- What kind of information do you receive?
- How do you process your information?
- To whom do you send your information?
- What is the contents of the information?
- What is the frequency of the events?
- How do you plan/direct this?
- Do you receive information you do not need?
- What else do you want?
- etc.

Business model

By filling in the questionnaires, staff members and business analysts gradually obtain the building stones for a second model, viz. the model of the business under investigation.
These building stones comprise activities, information flows (activating information), directives, data bases (reference information). By supplying the source and destination of the information used by staff members, it is possible to establish relations between the various information flows.
The information dictionary/directory (InDD) is the outcome of the activities undertaken so far and a model of the business investigated.
This is called the business model.

Completeness and correctness check

The InDD tool is now able to compare the basic model stored in the computer with the business model; so a comparison between "as it should be" and "how it was encountered". The tool gives a print-out of the differences. With this print-out the business analyst will go back to the staff members and discuss the differences mentioned.

A number of differences will be irrelevant, because the business investigated does not work as described in the standard model. Staff member and business analyst will agree on this and store their motives in the business model. In a further stage of the computerization project these differences will not cause any problems.
Other differences will be found to be relevant.
The conclusion will very often be that information was omitted.
This information will be included as yet in the model.
A third category of differences between both models may be characterized as bottlenecks (we know that it should be so and we would like to have it too).
These are bottlenecks in the organization which conflict with the efficiency and effectiveness of the information supply.
Management must discuss these bottlenecks and decide whether they should be eliminated or not. The business model will be adjusted, based in part on the elimination of the bottlenecks, thus gradually reflecting the required information supply.
The discussions between staff members and business analysts are supported by various reports from the InDD tool like:

- overview of all activities, data bases, documents etc. within the business model;
- report of all activities that form a bottleneck;
- report of all activities that are triggered by time;
- report of all activities that have a directive;
- report of all activities that deviate from the standard model, together with the reason why they do not appear in the current business model;
- overview of all activities with their in- and output;
- overview of all activities related to data bases;
- overview of all work places related to the use of information;
- overview of all functional units;
- etc.

All this information is directed towards a "management by exception" concept: the points on which there is mutual understanding need not be discussed; instead time and efforts can better be spent on the elimination of differences and bottlenecks. Thus, completeness and correctness of the information needs, and later of the system demands are ensured.

Process approach; information streams

After the business model is complete and correct, it is possible for the InDD tool to describe the processes within the model.
This is of great significance, for the purpose is to develop an information system that reflects correctly all the processes within the business organization.
Since the business model is a faithful representation of the business organization, it can be used for generating process models. These models are named information streams. Starting point for this generation is the presence of external elements that trigger the model. For example, orders from customers, quotations from suppliers, receipt of payments, etc.

A second type of information stream originates during a certain period of time: those activities that are triggered by time start up by themselves and lead to a process.
For example: coping with back-orders, payments of salaries, dunning procedures, etc.
The results generated by the tool are called trigger tables.
The business analyst will have to adapt the information to the information streams.

Architect models

By bringing in data in the business model, it will gradually evolve to a design model. If top management is able and willing to find solutions for bottlenecks, the business model can be changed. This means creation of a architect model based on the business model. The information streams derived from the functional business model remain within the process model. From the architect model process models can be made.
This is done in the same way in which the information streams are derived from the business model. Because of their relation to processes, these models are called process architect models.
An architect model is a business model which includes data elements.
An architect model is also a collection of process architect models.

A further examination of the models can or will lead to the coupling of one or more process architect models. This influences also the data base philosophy. These changes must be included within the architect model. As a result of all these activities a process architect model can be seen as a future information system which can be computerized.

During this stage further analysis of the activities of the business model will have taken place. If they occur within a process architecture model, they can be described as a transaction, divided into more transactions or form part of a transaction.
All the information relating transactions, quantities, the work places where transactions are executed, the physical location of buildings and departments, the physical location of the computer and data bases, is gathered during the interview stage and stored in the business model. The business model tool will gradually be able to report in greater detail about all the information stored.
These reports can be used as input for the network concept.

Priority, information systems planning

During the information analysis the business analyst will have collected a lot of wishes about information needs and processing. They will also have become acquainted with bottlenecks which may affect the problem solution to a greater or lesser extent.
Activities and their mutual relations form the structure of the business model. Since bottlenecks are inherent in activities, bottlenecks can also be considered in their dependence on the activity concerned. The tool reports about this in a bottleneck dependence table.

Connection with design tasks

If management authorizes to the execution of the computerization plan and its related projects, the design of the data structure, data bases, information systems, networks, transactions and dialogues must be executed systematically.
Part of the design activities can be executed parallel to each other; they also influence each other.

- Data base design

It is very important to establish a group called data base control and to involve this group in the future developments. Data base control must be supported by a data dictionary/directory system. All the data in the information data dictionary/directory must be taken over in the business data dictionary/directory.
For the design of data models and structures, the business analyst will interview the staff members a second time. The purpose is to collect all data elements in the business model, to recognize them and to relate them.
If the business model has already been developed into an architect model, all data elements will be related by means of information analysis. The result is a central data model. From this general model logic structures are made for the purpose of data base and/or file structures. Business policy is an important factor in the decision about the central or decentralized storage of data.

- Information system design

The input for these designs is formed by the process architect models which are described in the priority planning and are fundamental for the computerization plan. Input for this stage is also formed by the logic design structures from the data base control group. The design and building of information systems should be to started methodically. Dialogue simulation and transaction analysis must be part of the development method for the building of interactive systems. Also for information system design it is necessary to contact the end users again in order to gather the exact system wishes.

- Network design

To improve the network, design information is stored in the architect model of the InDD, such as number and location of work places, transactions involved and related quantities. The network design is based on this information. Also the location where information is stored, influences the network design.
Transaction analysis is part of the network design method.

Supporting aspects, advantages

It should be noted that InDD has not been designed to be an automatic analysis and information planning system. Perhaps methods of artificial intelligence will support this in the future.
The main advantage of the tool is to obtain the right and correct information about the system at an early stage of the project.
There is no guarantee that it will give the one and only correct information system, but it supports the business analyst in such a way that he can concentrate on on the most important parts of his tasks.
The comparison of models provides proof about forgotten and incorrect information streams. Such proof is a real new aspect of information engineering.
Of very great importance is the ease of use of the models when activities, information streams, data elements and time triggers are to be updated, added or deleted from the model.
In contrast to conventional manual production and amendment of schemes and typed print-outs, the InDD tool will deliver new reports very fast.
The advantages of the application are summarized in the five following points:
- enhanced completeness and correctness;
- short time needed for information analysis and rapid formulation of structured plans;
- automatic production of various (detailled) reports;
- quick, easy and consistent model manipulation;
- input for further development in design and computerization stages.

ARTESSA/3000

As mentioned before, I now come back to the application generator ARTESSA/3000.
In our efforts to automate computerization we have been searching continuously for tools to support this.
The starting point is that an application development tool must support the philosophy which we have for all the development stages. Commercially available tools did not meet our demands, of which the most important is that it must fit in with our PARAET methodology.
In 1982 we were able to obtain the rights for an application builder which, as a whole, did not meet our standards but could be changed with reasonable effort. We added a number of features to the basic concept, so that the tool has become very flexible, offering possibilities for a wide range of applications. Moreover, we were able to adapt the in- and output of the application generator to our standards.

Place in the development cycle

The use of ARTESSA/3000 means that in the logical design stage the information analyst is supported by the prototyping facilities. By registering all data elements once only, it supports the data base design. The total stage of the technical design in which analysis and programming take place is highly automated.

All screen lay-outs can be made as standard.
Only processes which have not yet been implemented within the building block concept must be described. Owing to the high level of abstraction the processes will have multiple applications.
Testing is very simple and fast, because the structure of the application is always correct and only the contents should be tested.
Changes and maintenance are executed at the highest level, resulting in a consistent system.

Why ARTESSA/3000?

We believe that ARTESSA/3000 is a good cure to fight the symptoms of the software crisis.
What are these symptoms:

- Backlog in application development. A computer department is always under pressure to deliver applications. The backlog in fulfilling user°s wishes is on average more than six months. There is also an invisible backlog (the computer department has no time to fulfil wishes, so wishes are not expressed). The backlog is also due to the lack of well trained staff.
- The lead time of projects is too long. At the time a project is finished, the specifications are out of date or no longer in agreement with changed business circumstances.
- Poor quality of developed systems. The user is confronted with a system that does not do what he expects. This is caused on the one hand by a lack of communication between end user and computer staff, and on the other hand the educational level of computer staff is directed towards system development techniques and too little towards the judgement of and thinking in terms of business structures.
- System development is often tied to one person, involving transfer risks, specially during the maintenance stage. In every EDP environment every professional seems to have a number of standards in his toolkit.
  Therefore, an an EDP department will always be confronted with a number of individuals who have their own way of working. So as long as a method does not compulsorily stipulate a specific way of working, we will continue to be confronted with problems in the area of transferable software. Therefore, a tool that requires a certain way of working will contribute to the standardization of the development of applications.

Characteristics of ARTESSA/3000

ARTESSA/3000 stores all data in a central data dictionary. This includes data elements, record lay-outs, file descriptions and screen lay-outs. For example, the change of the length of a data field automatically results in adaptation of all relevant update programs. As soon as all data descriptions have been loaded, prototyping can be started.

The end user should be warned that the application has not yet been built, because if he sees it working he will believe it is the real thing.
Processes which have not been described before are written in a high level language which is very user friendly to the EDP professional. It describes the "what" of an application and it is hardly necessary to describe the "how".
The applications developed with ARTESSA/3000 can work independently of the development environment. There is no "run time" tool necessary on the target computer. For a single fee it is possible to develop software in a centralized environment for subsidiaries.
This is due to the fact that ARTESSA/3000 generates ANS/COBOL 74.
As a result changes can be made in the COBOL source.
We urgently advice not to do so in order to prevent inconsistency of systems. Moreover, this compilation is only done to create independence of the development environment.
The COBOL that is generated is efficient and well-segmented.
ARTESSA/3000 can be used for the development of a complete application. It includes on-line, batch and report building blocks.
Because of the use of standard H.P. software like KSAM, MPE, V/PLUS, DICT/3000 and IMAGE/3000, ARTESSA/3000 can be implemented on all H.P.3000 models.
Interfaces to non-standard H.P. software like RELATE/3000 and COGNOS products are available or in preparation.
ARTESSA/3000 generates its own documentation. The level of the documentation can be adjusted by parameters. In the maintenance stage only amended parts of the system can be documented.

Functional building blocks

In our opinion it is impossible to offer a satisfactory solution for the variety of applications, using one or more models to generate complete program structures. As a completely free format means COBOL programming, our aim was to find a solution between these two extremities.
Therefore, we have chosen in favour of using functional building blocks related to a specific task. There are building blocks available for the menu structure at the start of an application, and building blocks to cope with update, input and delete manipulations of one or more records and/or files. These building blocks have been preprogrammed by us, using JSP. Owing to the use of this tool and the fact that these building blocks have been used in a variety of applications, we are confident that programming is 100% error-free.
So building blocks contain tested coding. The total application consists of a number of joined building blocks.
As mentioned before, there is a possibility of adding specific tasks to applications. This is done on so called invocation points within the building blocks. The processes to be invocated can be written in the programming language of ARTESSA/3000 or in COBOL.

Procedure

The whole procedure is based on entering the data in the computer only once. First a description of the data elements is made. Next, the file and/or data base structure is described by grouping data elements and appointing key elements. With the use of the screen painter it is possible to design the screens in cooperation with the end user. For the definition of reports user-friendly aids are available.
It is now necessary to structure the application and divide it into subsystems, applications and transactions. In this context we mean by transaction the lowest level at which a group of building blocks perform an update, input, delete or print process.
Applications consist of a number of transactions and describe an information stream. Subsystems consist of a number of applications which form together the information stream of a department or group.
After describing the structure of a subsystem, the applicable building blocks must be selected. To facilitate this choice it is important that already in the logical design stage the applicability of building blocks is taken into account.
The system makes it possible to generate a prototype of the application and to show this to the end user. The only thing that is missing are the calculation rules to get the result in certain data elements. Addition of these calculation rules is the last step before an application can be generated completely.
Documentation has already partially originated during the generation and can now be completed.

ARTESSA/3000 during maintenance stage

Changes in an application can be made in the system definition. After generating, new software will arise. Amendment of a generated COBOL source is possible, although the relation with the system definition is corrupted. Therefore we strongly recommend the former appraoch.
By implementing changes at system definition level, the aim of transferable software is ensured. This is one of the great advantages of using ARTESSA/3000, because it prescribes the way of working. Besides a saving on labour which is realised during system definition, the use of ARTESSA/3000 will also involve savings during the maintenance stage.
Minor changes can be made quickly. The decision to reconstruct a system can be taken more quickly and with greater cost effectiveness. In the maintenance stage it is possible to opt for completely new documentation, or for documentation of the parts of a system that are to be changed. Also the level of detail of the documentation can be controlled by parameters.

Conclusion

I have tried to give you a broad outline of our approach within the ASSYST-RAET GROUP, and our ideas about developing software and the aids which make it possible to automate the process of computerization.

Biography

Hans van der Leeuw

Product manager for A-R Products (owned by the ASSYST-RAET GROUP).
His computer experience began in 1967 with IBM 360 equipment.
Within the ASSYST-RAET GROUP he has been working for 6 years as management consultant in the International and Special Projects group.
Before his present employment he had the function of data centre manager.

Application building with ARTESSA/3000, description of concepts and facilities.

Chris Lewis

Summary

The intention is to make you familiar with the general concept, to give a summary of the features, how the product can work for you and your organization, to describe the benefits.

ARTESSA/3000

ASSYST-RAET°s consultancy and education services in methods and tools for software development have always been on the leading edge of the state of the art in software engineering and have been recognised as such by the market place.

The system development methodology PARAET, developed in the early seventies by RAET, are adopted by many large and/or international operating corporations. These corporations have formed their own user group in support of these methods and tools out of the PARAET concept. Since early 1970 research and development in software engineering has been a continuous process within the ASSYST-RAET GROUP.

The urgent needs of the ASSYST-RAET GROUP to find fundamental solutions to the software crises in support of its own services to the market place has lead to the development of the application generator ARTESSA.

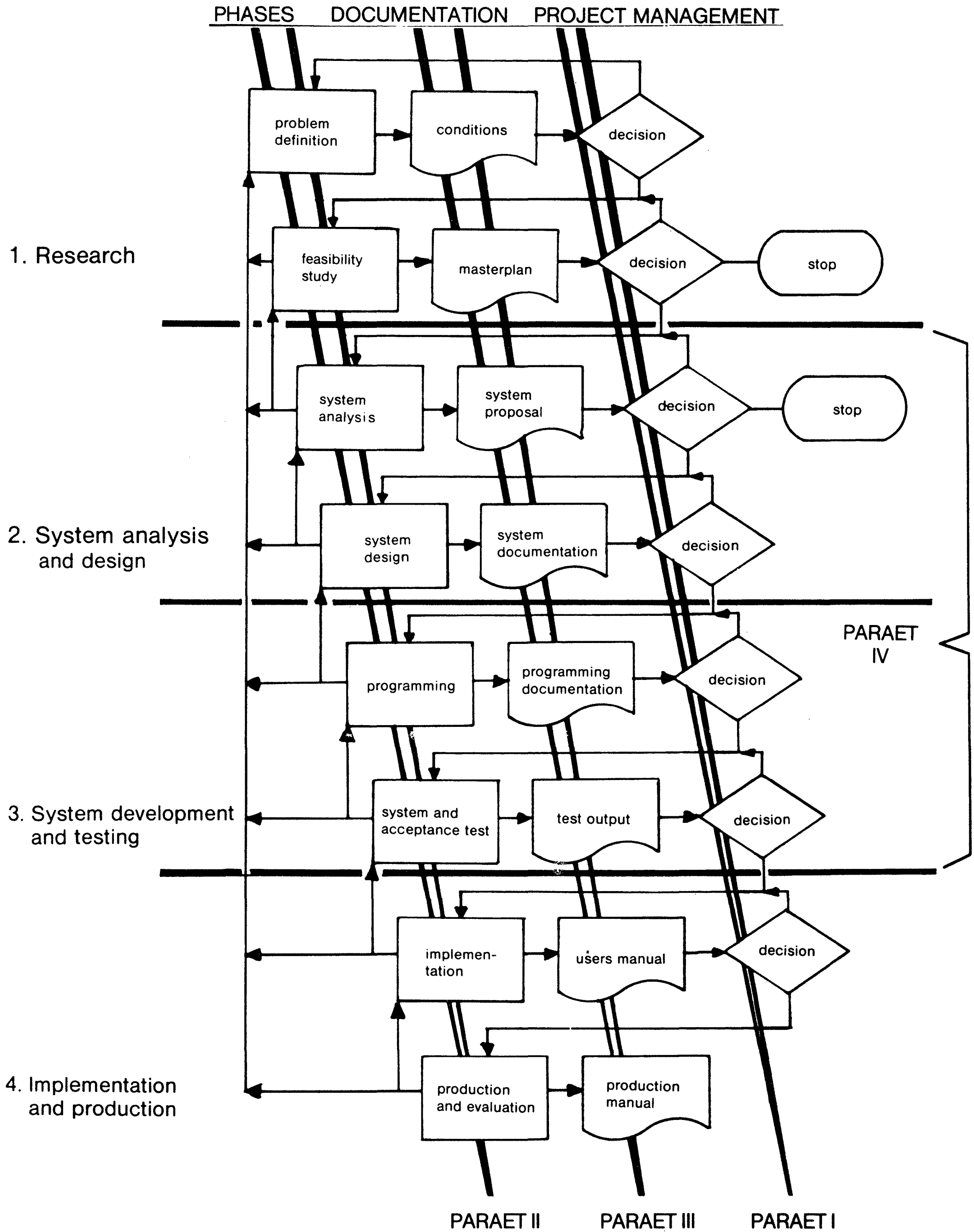
The first prototype of a product leading to ARTESSA was operational during early 1982. Out of this a fourth generation toolset has been developed for use within the ASSYST-RAET GROUP. This toolset has been in full operational use since September 1983 resulting in large gains in productivity and flexibility of the applications developed.

The result of the packaging of our toolset to the specific environment of the HP 3000 range is ARTESSA/3000. With this action the ASSYST-RAET GROUP offers its latest experience and capabilities in system and software engineering to the market place.

Application generators as part of the total development cycle

By examining the logical stages of development within any computerisation project, as shown in the PARAET basicmodel flow diagram, the traditionally time consuming and thereby costly aspects become easily identifiable.

BASIC MODEL OF A COMPUTERIZATION PROJECT



1. Research - the problem in this phase is ensuring that the user has defined (or in some instances recognised) the problem and need, completely and correctly.
2. System analysis and design
   - an itterative process with research, through system proposals.
3. System development and testing
   - a time consuming and skill dependent process.
4. Implementation and production
   - this phase is totally dependent upon the quality of the system developed in both business and technical terms.

From the above it may be seen that no one phase stands apart from the others but improve the quality and standards of phases 2 and 3 and immediate benefits are obtained in the time necessary for phases 1 and 4. If in addition the productivity levels within phases 2 and 3 are dramatically improved then the real benefits are obtained.

It is therefore to the system analysis and design, and the system development and testing phases that ARTESSA/3000 primarily addresses itself to obtain the greatest gains in quality, productivity and flexibility whilst minimising the effects upon creative application design.

Main characteristics of ARTESSA/3000

When considering the characteristics of an advanced software product such as ARTESSA/3000 it is best to examine the problems which confront those responsible for the use of computers, be they Managing Directors with overall concern for cost and business efficiency or Data Processing Managers with their specific concerns for the quality, efficiency and consistent reliability of the software developed and maintained by the computer facilities under their control.
In the majority of cases the concerns are the same and in every case the solution is the same, ARTESSA.

The cost of software development has escalated to the point of being unacceptable.
- ARTESSA cuts development time by factors thereby reducing costs.
- ARTESSA dramatically reduces the need for expensive software "specialists" thereby reducing costs.
- ARTESSA provides powerful prototyping facilities at the earliest stages of the system design reducing the time cycle for business design acceptance; thereby reducing costs.
- ARTESSA moves the emphasis of development from programming to system designers and analysts increasing individual contribution and efficiency and thereby reducing costs.
- ARTESSA eradicates logic and structure faults dramatically reducing the time necessary for testing of individual programs or entire applications, thereby reducing costs.
- ARTESSA interfaces completely with all existing HP system software and utilities thereby preserving and capitalising upon previous investment and experience.

The development cycle of software from definition to user acceptance is too long.

- ARTESSA reduces the time required for the initial design phase by providing the system designer with:
  . prototyping capabilities;
  . screen proposal and design facilities;
  . report generation;
  . integrated data dictionary support.
- ARTESSA reduces the time required for the application development phase by providing:
  . integration of data registration and logic/process definition;
  . standardisation of application structures through predefined logic building blocks;
  . removal of dependency on individuals for specialist software skills;
  . high level fourth generation language for user defined processes, generating syntax error free COBOL code.
- ARTESSA reduces the system and acceptance testing cycle time by:
  . complete applications are developed free from logic, structure and COBOL syntax errors;
  . all supporting technical documentation is produced at the same time;
  . all JCL necessary for installation and live running is produced at the time of generation;
  . the end users have previous experience of the application from the design stage prototypes, thereby reducing the time necessary for acceptance testing.

The introduction of new development software can be costly in terms of training, inefficiency during the initial period of use and redundancy of current skills and experience.

- ARTESSA is based on COBOL to ANSI 74 standard therefore easily understood;
- ARTESSA logic and design concepts are based upon the widely accepted and used Jackson Structured Programming method;
- ARTESSA is easy to use being totally interactive with "fill in the screen" approach;
- ARTESSA is extensively documented, every stage being fully supported by "help" screen facilities and comprehensive supportive documentation;
- ARTESSA makes full use of all the excellent HP 3000 software and features like IMAGE, DICTIONARY/3000, V/PLUS etc.
  Additionally interfaces are available to leading third party software such as POWERHOUSE and RELATE/3000;
- ARTESSA generated applications are processed without the support of any non-standard, non-HP runtime environment, and the application runs independent of the development environment.
  There are therefore no requirements or constraints imposed upon the operational environment.

The costs associated with maintenance of conventionally developed software are ever increasing, for a number of reasons:

1. bad specifications resulting from insufficient user involvement during the initial stages lead to logic design faults which invariably do not show up for some time.
2. applications are invariably developed without standardisation and are very often author or skill dependent.
   . ARTESSA encourages end user involvement at the earliest stages of development with the powerful PROTOTYPING capabilities.
   Fully working, menu driven, applications can be produced quickly and easily enabling the users to SEE and USE the basic system before a single line of code has had to be written.
   . ARTESSA eradicates logic and structure faults resulting in more reliable software.
   . ARTESSA standardisation and use of pre-defined logic building blocks removes the reliance upon individuals or specific software skills.

Frequently, enhancements or modifications are required to existing applications. In many cases because of the inflexible design redevelopment of the entire application is the only practical course of action.

- ARTESSA°s approach to user involvement at the earliest stages reduces the frequency of and necessity for modifications or enhancements.
- ARTESSA°s integrated data and process approach enables modifications and enhancements to be made easily and quickly whether it be to a single program or the entire application.

Applications development using fourth generation tools require the development environment to be resident before they will run.
In a multiple CPU environment the cost of licenses can be prohibitive.

- ARTESSA generates COBOL/ANSI 74 applications which are therefore totally independent of the development environment and fully transferable between HP 3000°s from the model 37 to the 68.

Summary of ARTESSA characteristics

- Greater efficiency at all DP staff levels;
- greater user involvement;
- removal of dependency on highly experienced programming staff to make full use of HP 3000 hardware and software;
- greatly increased productivity, by factors saving time and money during development;
- greater reliability and flexibility reducing maintenance costs;
- a system totally independent of the development environment
  . therefore fully transportable between HP 3000°s.

Biography

Chris Lewis

is salesmanager for Assyst U.K. which is partially owned by the ASSYST-RAET GROUP.
He has gained experience in EDP by working for his own management consulting firm. He also worked as sales manager for the Assyst subsidiary in Saudi Arabia for over 4 years.

TAKING THE MYTH OUT OF ARTIFICIAL INTELLIGENCE.
(Building and using Expert Systems)

Julio Moreno-Dávila.
IMEDE, Management Development Institute, Lausanne, SWITZERLAND.

CONTENTS:

1) Expert Systems (ES).

   - Knowledge Base.
   - Organized in rules.
   - Inference Engine.
   - Natural Language Interface.
   - Plausibility handling.
   - Explanation Window.
   - Debugging facilities.

2) Putting the Myth into Artificial Intelligence.

   - Algorithms (first order predicate calculus).
   - Programming Languages (LISP, Prolog).

3) Knowledge Engineering (KE).

   - ES Shell.
   - Making the human expert(s) explicit.
   - Knowledge base debugging.

4) Taking the myth out of AI.

   - BES (Basic Expert System).
   - Prolog-like Interpreter.
   - Build-in Inference Engine.
   - Base modification, debugging facilities, explanation window, etc.
   - Plausibility handling.
   - 350 Lines of BASIC.

5) Pedagogical example: Choosing Table Wines, a miniature ES.

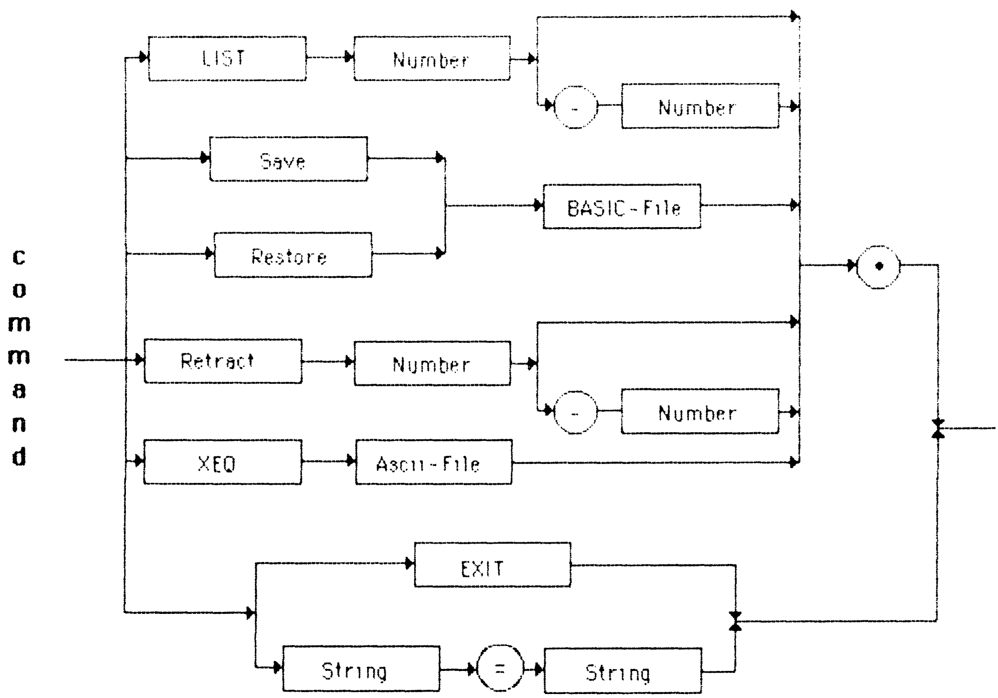
6) A more sizeble ES: Granville's new strategy of stock market timing for maximum profit.

Biography
unavailable.

```
] meal is not including meat:-not red.
] list.
 1      red:-meal is including meat.
] white;rose':-meal is including fish.
] red--white--rose'.
] list.
 1      red:-meal is including meat.
 2      white;rose':-meal is including fish.
 3      red;white;rose'.
 4    :-red,white.
 5    :-red,rose'.
 6    :-white,rose'.
] wine=drink
] ?-white;red.

 Is meal including meat ? why?

 I am currently trying to stablish if drink
 is a red wine and this information is
 relevant because of clause # 1

 Is meal including meat ? no

 Is drink a rose' wine ? no

 Is meal including fish ? why?

 It is firmly stablished that:

drink is not a rose' wine.

 I am currently trying to stablish if drink
 is a white wine and this information is
 relevant because of clause # 2

 Is meal including fish ? yes
 Inference! drink is a white wine. Plausibility = 1     0


 No (other) information could be inferred.

] exit.
```

Choosing table Wines: A miniature ES.

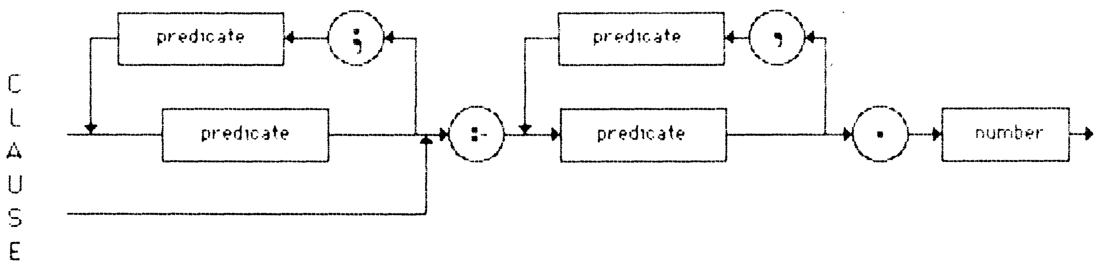
COMMAND SYNTAX.



Legal Examples

LIST	Display the current list of clauses.
Save Myfile	Save Expertise in "Myfile"
Restore Myfile	Restore the Status before Saving
Retract 7	Clause #7 is no more valid (beware renumbering)
Retract 1-44.	
XEQ Asciifil.	Accept Input from Asciifil
EXIT	Finish BES OPeration.
patient=Peter	Initialize Status and Gives a Name to Subject

CLAUSE SYNTAX.



, Stands for AND

:- Stands for IF

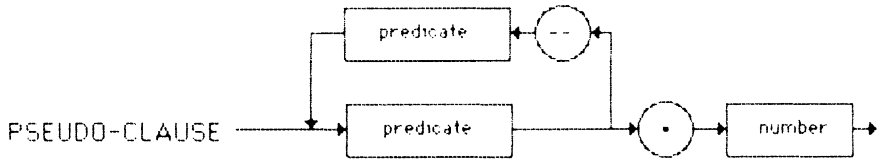
; Stands for OR

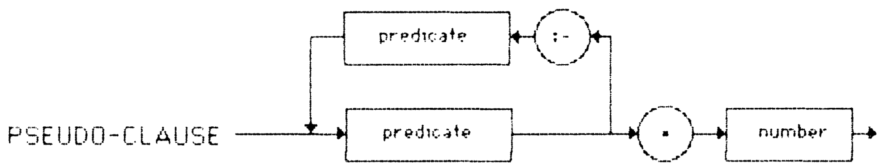
Valid Examples.

red,white;rose'.
red - meal is to include meat
white;rose' - meal is to include fish 95
:- $2k14 333

This is the plausibility!

PSEUDO - CLAUSE SYNTAX



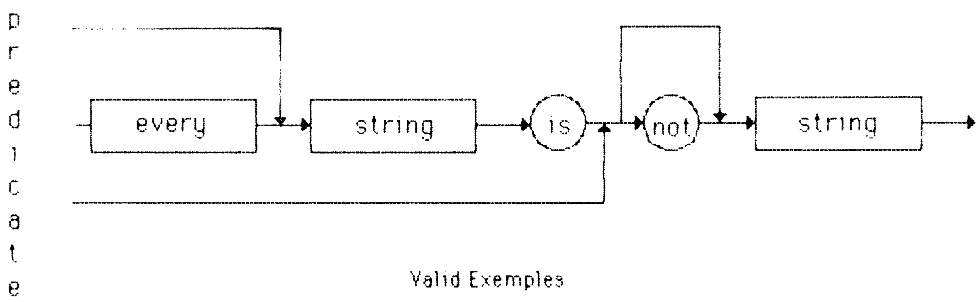


-- Stands for EXCLUSIVE OR
\- Stands for IF

Valid Examples

beaujolais -- bourgogne -- bordeaux
rouge - meal is to include cheese - dish is the last

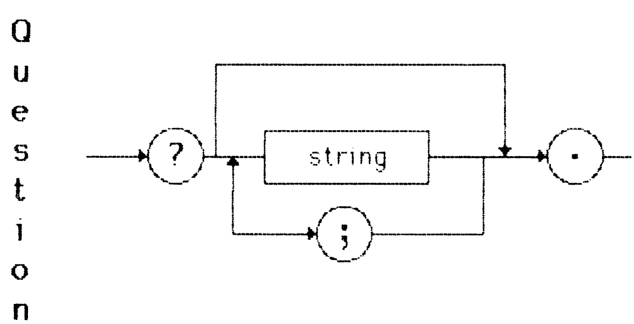
PREDICATE SYNTAX.



Valid Exemples

red
wine is to be bourgogne
k7 9xx !!! p
you should rather consider to use expert systems to choose your table wines

QUESTION SYNTAX.



Valid exemples.

?- (last question is implied)
?-red, white, rose'

OPPORTUNITIES AND DANGERS OF FOURTH GENERATION LANGUAGES

BY

ROBERT REMILLARD
INFOCENTRE LTD.
MISSISSAUGA, ONTARIO, CANADA

SUMMARY

The objective of this paper is to review the direction and definition of Fourth Generation Languages (4GL's) and address the impact that they will have on data processing departments and their users. We will look at several real examples, review the dangers and opportunities of 4GL's, look at their benefits and suggest practical tips on how to put them to work for your company. Along the way, we will look at the changes that are needed in data processing to accept and successfully implement 4GL's.

BACKGROUND

Before we go into the definitions and concepts of 4GL's, let us review the trends that brought them into existence in the first place.

A recent computer industry forecast predicted that the number of installed computers would increase by a factor of ten (10) in the next ten years. That was before the fantastic growth created by the micro-computer. This means that the demand for software would increase considerably.

In a Computerworld survey, it was indicated that the number of computerized applications in existing data processing departments is growing at a minimum rate of 45% per year. Again, this indicates that a lot of new software is going to be required from data processing staff.

Computerworld also estimated that the average cost of one debugged computer instruction is approximately ten dollars ($10.00). What this means for data processing managers is that their goal should be to reduce the number of lines or instructions required for an application in order to reduce the cost of development.

There is also a very interesting fact about our industry. Last year, North America produced more computers than programmers and science graduates combined. At this rate, we cannot continue to develop software in the same way as before.

As everybody that has attended an HP sales presentation knows very well the cost of hardware keeps going down for the same (or improved) software performance, while the cost of people increases all the time. This indicates that programmer productivity has to increase to keep up with the hardware trends in our industry.

A recent article in Computerworld indicated that the average application backlog in North American DP departments is between three and four years. This is only the visible backlog of documented requests from users. It does not include what James Martin calls the invisible backlog. This is the one where users do not even consider making a request for the applications they need because they know that it will take several years before they receive a response.

In summary what we see right now is that the traditional approach to systems development has to change drastically in order for every DP department to survive in the next decade. People productivity has to increase dramatically if we do not want every human being to be a programmer by 1995.

SOLUTIONS

Fortunately, there are solutions to the development bottleneck experienced today. The first solution that appeared in the marketplace is pre-packaged application software. This is especially true of the HP3000 family since there are many good packages available.

Pre-packaged applications usually offer a good starting point for the most common needs (ie accounting, manufacturing, etc).

Another solution is to let end users create their own solutions with powerful tools. This trend has been accelerated lately with the new type of software on micros, such as spread sheets, file managers and graphics.

Finally, another approach is to have systems analysts create applications, rather than write program specifications. The benefit of this approach is that the time required to develop an application is greatly reduced through prototyping, where the user gets instant feedback on how the system might look and operate. This avoids costly misinterpretations and the generation of applications to meet needs that changed long ago.

No single solution will necessarily be the only solution for your company. DP managers should always evaluate each approach for different requirements and make a decision on the best alternative(s).

HISTORY AND DEFINITION

We are hearing a great deal about Fourth Generation Languages. Let us first review some history and where they come from. The first generation was the machine language (binary). Then came the second generation of "Assembler" type languages. The third generation was that of the supposedly machine independent languages such as Cobol, Basic, Pascal, Fortran, PL/1, RPG, etc. This is the generation that most of us are familiar with. It provide more friendly interface with the machine, but is still driven by programmer logic.

My own definition of a fourth generation language is: "A non procedural data base language that produces results in one-tenth of the time than with Cobol, or less". This is also close to James Martin's definition of 4GL. What it really means is that it should be a language that is action driven instead of logic driven. With a 4GL, one should tell the machine what to do instead of how to do it, and the results should be user friendly. It means that data processing's productivity should be improved by a factor of ten or 1000% to qualify as a 4GL. Also, a language that works exclusively with indexed files instead of data base management systems would not qualify as a 4GL.

SOME CONCEPTS (BUZZWORDS)

There are several categories of software tools that fall in the 4GL family. Let us review them:

A) **Simple Query** facilities. They have been around since the first disc based systems. They allow stored records to be printed or displayed in an appropriate format.

B) **Complex Query** languages. These are like the Query on HP3000, allowing retrieval and joining of records based on certain qualifications or parameters. Traditionally these languages are poor performers on the machine.

C) **Report generators.** These are tools for extracting data from files and formatting it into reports, while allowing substantial arithmetic or logic processing before displaying or printing it. These tools have been around for a while in the HP3000 environment and are the first wave of accepted fourth generation tools.

D) **Dictionaries simply** are files in which is stored pertinent information about data and their use by programs.

E) **Systems or applications generators.** These represent the latest trend in 4GL's. They contain modules that permit an entire application to be generated.

The best application generators are created with the user in mind and will automatically generate user friendly applications that include menus, online help, error messages, security structure, etc. Systems generators, to be classified as fourth generation tools, should generate a non procedural code and definitely not a third generation code such as Cobol. Tools that produce third generation code are known as code generators and can be classified as third generation productivity tools.

F) **Documentation tools.** This concept is now expanding the scope of 4GL's. As we all know, the implementation of an online application is usually only as good as the user documentation manuals that come with it. The better 4GL's offer facilities that automatically generate these manuals.

G) **Parameter-driven packages.** The latest packages include parameters that can be changed by the users to make them fit their own environment. The spread sheets on micros are good examples of parameter-driven user applications. MM/3000 is also an example of this generation with the customizer option.

CASE STUDIES

Now that we all know the terminology, definitions and history, let us look at some real situations of companies and institutions that have benefited from using 4GL's.

These cases are documented internally at Infocentre, or have been taken from examples given by James Martin in his book, "Applications Development Without Programmers".

Case 1 : In the Chase Manhattan Bank an end user department itself created a complex system for online analysis of the Chase's management accounting data in ADMINS II. They accomplished this in 4 months with two people and claim that the DP department had estimated they would take 18 months with 20 people at a cost of $1.5 million.

Case 2 : Mark-Hot, a manufacturer in the province of Quebec, had a backlog of applications estimated at 3 years. Using SPEEDWARE on an HP3000, and the same number of people on staff, they got rid of that backlog of applications in 6 months.

Case 3 : At Infocentre, we have an application called T.O.U.R.S. It was written in VIEW/3000 and Cobol. The application needed six people to support it for a base of four clients. The listing printout was over 10 inches thick when on a table. The application was mostly re-written in SPEEDWARE and the listing is now less than two inches thick. We still have four people to maintain it, but our base of clients has grown to over fifteen sites.

Case 4 : Corfax Benefits System is a software supplier of pension benefits systems. The software is both sold as a package or rented on a time sharing basis to client companies in Canada. The application is mostly written in SPEEDWARE with some Cobol subroutines. They estimated the increase in the area of maintenance productivity to be over 2000%.

Case 5 : Mohawk College of Hamilton Ontario had two programmer/analysts do an evaluation of SPEEDWARE. As a test, the management gave to them what was estimated as a one year backlog of applications to play with. The two created all the applications from the one year backlog in three weeks. They are now using and teaching SPEEDWARE.

Case 6 : B.A.S.F., a world wide manufacturer of magnetic media has achieved an increase in programmer productivity estimated at 2500%, using SPEEDWARE instead of V/3000 and Cobol. Moreover, the DP manager tells us of improved morale in his department since users are now very happy with the turnaround time of new projects.

Case 7 : A world wide manufacturer with a division in London, Ontario had a purchasing control application to develop. The estimated development in V/3000 and Cobol was over 7 months for an application without user friendliness, menus and documentation. The same application was developed, tested, documented and implemented in 3.5 weeks, using SPEEDWARE. Moreover, the end result had online help and was controlled by menus.

CHANGE NEEDED IN DP

Hopefully by now we have convinced you that 4GL's can work and are for real. This brings the first major change needed in DP management for 4GL's to be accepted. There has to be an open mind to the new technology and concepts. As vendors of a 4GL, we often find that our single most important problem is that people do not beleive that we can do what we do. A lot of companies have decided that all development must be done in Cobol or another "passe'" language, deciding automatically against improving productivity. When Hewlett-Packard announces a new microchip that doubles the performance of the hardware, nobody seem to reject the claim (except IBM maybe). However, when a software manufacturer announces a new tool that can double programmer's productivity, most DP managers have serious doubts about it. Their view can be likened to that person in the 1950's, who was asked to get from Montreal to Los Angeles in half the normal time required. He took his car to a high performance shop, changed the engine, boosted the power, improved the suspension, added a bigger tank for fuel. When asked why he did not go by plane instead, he simply answered: "Those things, they will never fly!"

The second change needed in DP management concerns the wall between the user and the application creator. It has to disappear with 4GL's. This wall consists of written specifications that must be frozen, a multi-year backlog, slow programming, lengthy program documentation and the fact that most bugs are in analysis and design. By the time the application finally arrives, the original requirements have changed.

Prototyping is a new concept introduced by 4GL's. Programmer/analysts can now sit with the user and generate a prototype application that will serve as a start up systems design model. When the 4GL has a user documentation generator, the user can then take the documentation back with him/her and write the required enhancements on it. The users do their own analysis. Also, when in doubt with a new application, try a prototype instead of a formal analysis. The cost might be less.

Another change needed in DP is that we have to start managing information instead of code. After all, we are called Managers of Information Systems, not Managers of Code. Our role is to give the users access to the data that they require to perform this task. Our role has to become a guide to the information, by knowing where the information is and then telling the system what we need instead of how to retreive it.

The new applications created have to be user friendly and flexible enough so as to accommodate more than one need. If the application is friendly, you will not get numerous phone calls from users who do not understand how to use your application. A way to generate flexible applications is to always include substitution parameters in your applications. A report could, for example, ask a user directly for the sort criteria With one report program, you may be creating in effect 25-50 different outputs that will be defined at run time by the users. The reports become user-customizable and multi-functional.

We also have to avoid being a maintenance department. (see figure 1). The maintenance pitfall is very real when we develop more and more applications that we have to maintain later. 4GL's can help in that area, since the code generated is usually one tenth to one twentieth the amount of Cobol needed for the same application.

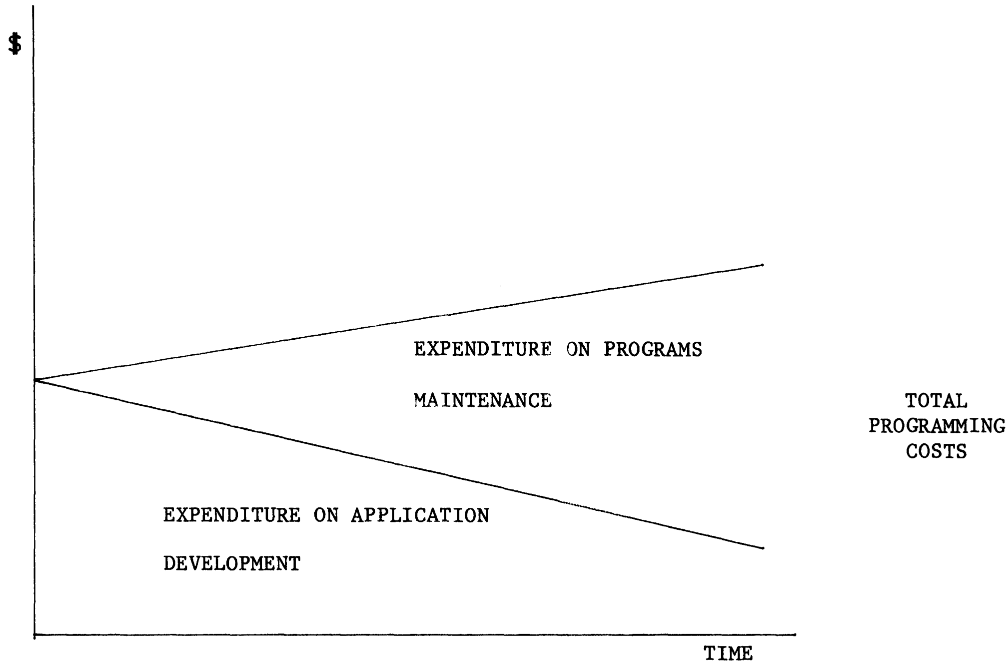


Figure 1: New application progress is often deferred by the rising costs of maintenance. Some companies spend 80% on maintenance and 20% forging ahead.

Finally, DP managers have to look for existing packages instead of trying to re-invent the wheel all the time. Today, there are a lot of very good application packages that can offer a good fit for 80-90% of the needs for certain applications.

DANGERS OF 4GL's

One of the most serious dangers associated with 4GL's is the power given to users. For example, users can write their own inquiries into large data bases. This can create a situation whereby, a user could read a 200,000 record data set sequentially to find a keyed customer number and then re-read the same data base again sequentially to extract the customer address. Obviously the system performance would suffer tremendous degradation. This can easily be avoided by retaining some level of control at the data processing level.

Another danger of 4GL's is to try to duplicate old systems with the new technology. For example, if you have an old application that was running in a certain manner and you try to redesign it using a 4GL, you should always try to take advantange of the new features and technology instead of duplicating what is there. 4GL's have tremendous benefits when simple guidelines are followed.

Control needs are greatly increased when a data processing department uses 4GL. Since many new applications would be implemented quickly, security measures around the system need to be revamped. There would be an increase in the number of data bases and files on the computer and this should be monitored in order for the data processing department to become an Information Centre.

One of the more unapparent dangers of 4GL's is at the implementation level. Since you will be creating a lot of new systems with this new technology, there will now be a shift in the backlog of application from the data processing shop, to the users. Your shop will now produce and try to implement more systems than your users ever dreamt possible.

What this means is that you have to start streamlining all of the implementation procedures. Although todays fourth generation languages are becoming increasingly powerful, you should not try to do everything with the 4GL. For example, some calculations could possibly be better done and/or simplified by exiting to a Fortran subroutine. With your increasing use of the 4GL tools, you will find those areas of application development that can be developed easier and faster, and performed more efficiently with a third generation call.

The last danger associated with fourth generation languages is their performance. Unfortunately, some 4GL's available today, while offering you a great improvement in the development cycle, are poor performers when put into production. However, this is not a generalized case and it should not divert you from going with a 4GL. It is however, one of the criteria which should be evaluated when choosing a tool for your system.

BENEFITS OF 4GL's

Obviously, the first benefit of 4GL's is the huge savings that can be had by companies that use them. As we have shown in our case studies, the increase in productivity can range from 700% to sometimes 2500% and even 5000%. That represents substancial savings in personnel costs.

With a 4GL, your backlogged applications, for which the development in a 3GL would take at least 10 times as long, can be put in to production a lot faster. This means that the savings generated by those same applications can start happening for the company a lot faster.

Another major benefit of 4GL's is the fact that through code reduction, maintenance is a lot less important. Your staff will now be able to concentrate on satisfying users needs with new projects, instead of maintaining old obsolete applications.

Happier users and increased morale in data processing is also associated with the use of 4GL's. Obviously, if you can serve your users better, they in turn will show to your management how satisfied they are with your services. This will give your people more pride in their work.

One of the most underated benefits of 4GL's is the automatic standardization that is achieved though their use. As you well know, some companies will pay hundreds of thousands of dollars to consultants to get standardized methodologies in their departments. Most 4GL's, or at least the good ones, will provide you with standards at the data entry level, at the reporting level and even sometimes at the documentation level.

This leads to another benefit, namely the computer generated documentation. As we pointed out earlier, the very best fourth generation languages will offer automatic creation of user documentation manuals. This feature enables users to be transparent to the system, by allowing their replacement if need be.

Also, 4GL's tend to generate user-friendly applications from day one. This includes online help, error messages, friendly menu driven user systems, and so on. These features should also automatically be standardized by the tools. The benefit of the data processing professionals is that implementation is greatly facilitated.

Finally, one of the strangest benefits of 4GL's for programmers is that it gives them more time to play computer games. I will not attempt to translate this into a benefit which management could understand.

HOW TO MAKE THEM WORK FOR YOUR COMPANY

The first step in making a 4GL work for your company, is obviously, how to select it. First, you should always evaluate, more than one 4GL. It is no different than evaluating a computer. If possible you should try to get a trial tape of the software, load it onto your system and begin to develop applications. Very soon in the evaluation process, you will feel your own level of confidence with the tools. This trial gives you, a feeling for the support given by the vendor. In this trial process think of your users. Any new generated system should be given to the users for evaluation of the user-friendliness. We often neglect them. Also, during this trial period, try to develop a feeling for the performance of the tools while in production. As we pointed out earlier, fourth generation languages vary greatly in performance, some of them being splendid performers while some of them are dogs on your system.

In order to make a 4GL work, you will have some selling to do. First you have to sell to your high management the increases in productivity, the faster response to their needs, and the overall improvement in your shop. The second step is to sell it to your own DP staff. They will fear the new tool as competition for their jobs. This should be dismissed early in the process if you want to be successful.

Finally, you have to sell your new tool to the users. They are the ones that will benefit from faster turnaround, friendlier systems and improved documentation.

When you implement the new tools in your environment, start with the easiest applications. This will bring early success that will show your management and your users the power of the tools very early in the cycle. It will also familiarize your staff with the new technology.

With a 4GL, you have to learn to take advantage of the new tools. You will have to start to change the way of developing systems, by using prototyping for example. As we mentioned earlier, do not try to duplicate old systems the way they were.

Your next step in a successful implementation of a 4GL, is to change your role to that of a consultant. You will now have tools to give your users access to the data. Your role should now be to guide your users on how to get to the data. You have to end the big mystery surrounding data processing. With a 4GL, you no longer need that mystique to retain your own security in your shop. Your results will now speak for themselves and your status will be further enhanced in the corporation.

The last paragraph described the consultant approach. This also means that you need good control on where the information is stored. You will now have to develop file references, or dictionaries to tell you where the data is stored. Again with more new applications coming online, this is critical.

You will also need to maintain a catalogue of applications and data bases. The reason for this is that you will create so many applications and programs that very often you will find a user requesting something that already exists. Good systems documentation will prevent duplication of systems.

You should avoid unproductive study of systems. When in doubt, prototype the application instead of spending time studying it. If the prototype works, (and you will out find very soon in the cycle), then you try to turn it to production right away. This will save a lot of time and fully take advantage of the new tools.

A last suggestion is to always keep good communication lines open with the vendor of your tool. As a vendor I can tell you that we know a lot about how to make our tools work in companies. Very often we find that our users don't use us enough as a good reference on how to use our tools. The vendor should be able to refer you to another clients who have solved similar problems.

This technology is very new and finally is being accepted. There are very few good reference books, manuals, or training on how to make them work for your company. So please use your vendor as a direct source of information.

CONCLUSION

Fourth generation languages are now here to stay and those that do not believe this fact will be left behind, like those who continue to use an abacus instead of a calculator.

Although there are some dangers associated with their use, their tremendous benefits make them the avenue to the future. So open your minds and fasten your seatbelts. You are in for one brand new era.

BIOGRAPHY

Robert Remillard has a bachelor of Commerce degree from University of Montreal. He is the Marketing Manager of Infocentre Ltd., and has been associated with data processing since 1976.

HOW RAPID/3000 CAN ELIMINATE THE "MOVING TARGET" SYNDROME

STEPHEN M. RUSH, VICE PRESIDENT, MARSH AND McLENNAN ASSOCIATES,
ASSISTANT PROFESSOR, GOLDEN GATE UNIVERSITY,
BOTH SAN FRANCISCO, CA, USA

SUMMARY: The classic system development methodology was evolved to bring some order and professionalism to the process of creating information systems. This paper illustrates, through the vehicle of a short business case, some of the weaknesses inherent in this classical methodology and explores some of the ways that the RAPID/3000 family of software products helps to overcome these deficiencies, thereby increasing the chances for success in systems projects.

CASE - THE MOVING TARGET

Joe Ericson, Director of Data Processing at the Wilson Manufacturing Company, was in his office when his systems manager, Mike Adams, stormed in.

"I'm so mad at Harry (the national sales manager) I could strangle him!"

"Hey, cool down and tell me what's up."

"We just delivered the fifth set of sample reports on that sales forecasting system we're writing for him and he still isn't satisfied! He wants some format changes, again, and he wants to add a field which will require changing the master file. My people are fed up with the project and our backlog is growing like you wouldn't believe."

"Gee, I didn't realize the situation was so bad. Why, everyone was excited about this system because it looked so easy and would really help the sales types do a better job. When Jim (the former sales manager) got promoted and Harry was brought in, the first thing we talked about was how much he needed the data."

"Well, he may talk about needing it, but he sure doesn't want to use what we give him. I'm ready to abandon the damn thing. Linda (the lead programmer on the project) says that if she isn't moved to something else, she'll start looking around, and I really don't blame her."

Just then the phone rang. It was the VP of Administration requesting that Joe, Mike and Linda attend a meeting in the boardroom in an hour to discuss the sales forecasting system. The VP of Finance and the national sales manager would also be participating.

KEY POINTS:

* Time Lag Between Request and Delivery
* Personnel Changes
* Lack of Communication
* Inflexible Software Tools
* Lack of Proper Project Monitoring
* Ineffective Management

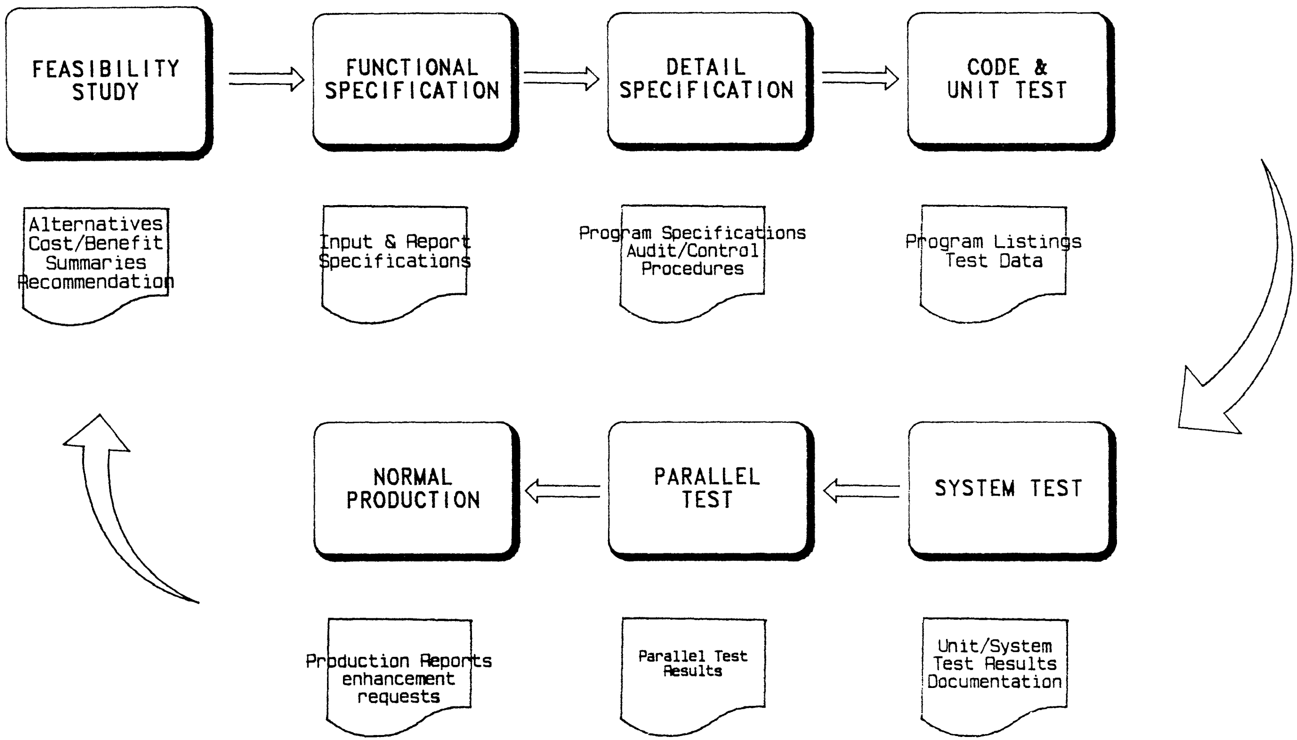
Let us examine how the classical system development methodology and the COBOL programming language contribute to some of the difficulties encountered in this system development project and how the RAPID/3000 family (or several other similar products) can help us avoid these pitfalls.

* Time Lag Between Request and Delivery

This is the single most important factor, in my opinion, in the failure of major systems development projects. In the classic system development methodology, (Figure 1), the user is asked (and expected) to have completely defined his needs at a very early stage in the process (Functional Specification). The programming staff then takes these requirements and codes a set of programs that will produce the screen displays and reports that the user has requested. This can take from several months to several years, depending on the complexity of the system and the skill and size of the programming resources assigned to create it. The user, however, has not removed his head and placed it on a shelf during this period. Quite the contrary is almost certainly true. The process of agreeing to a set of requirements at the completion of the functional specification phase has <u>started</u> him thinking in a disciplined way about what it is he needs. The result is shown in Figure 2.

Obviously, the more time that passes between the original system definition and delivery the greater the disparity between what the programmer delivers and what the user needs. To use an American expression, the programmer is then forced to play "catch up football", as shown in Figure 3, by embarking on an immediate program of enhancements and modifications to a <u>brand new</u> system. This of course takes time, delaying the availability of the new system and its benefits to the organization. It is also sad but true that some enhancements may not be economical to implement and the system is abandoned before it is used.

The use of tools like RAPID/3000 can help to overcome this problem in several ways. The feasibility of prototyping the system helps the user to understand and state his needs. Rather than a theoretical look at how the system will work, which is all he gets in the classical methodology, he gets a series of iterations and working models to help him identify the system features he needs. A model for system development using prototyping is shown in Figure 4.



THE SYSTEM DEVELOPMENT LIFE CYCLE

FIGURE 1

FIGURE 2 THE NEEDS GAP

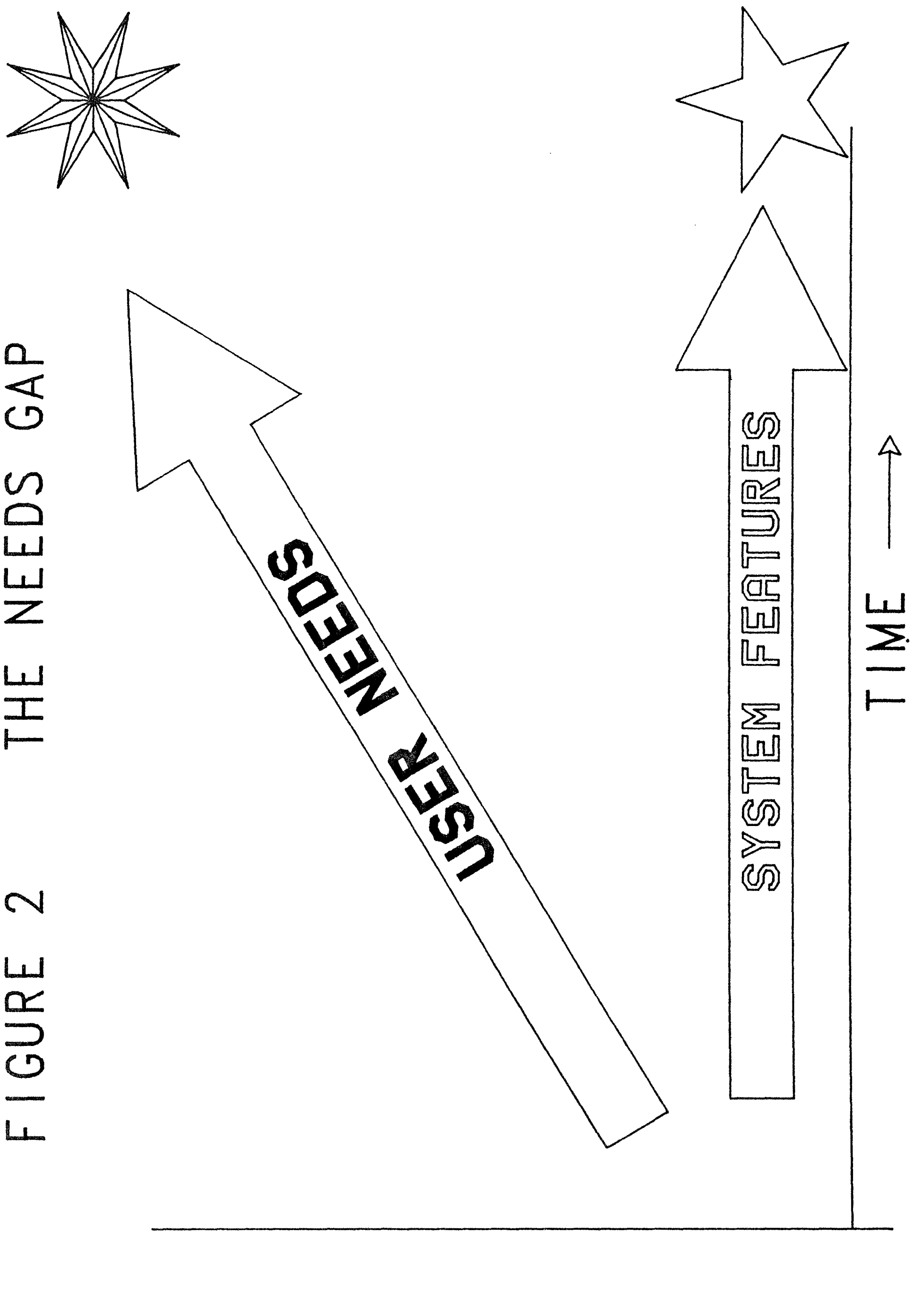


FIGURE 3 THE MOVING TARGET

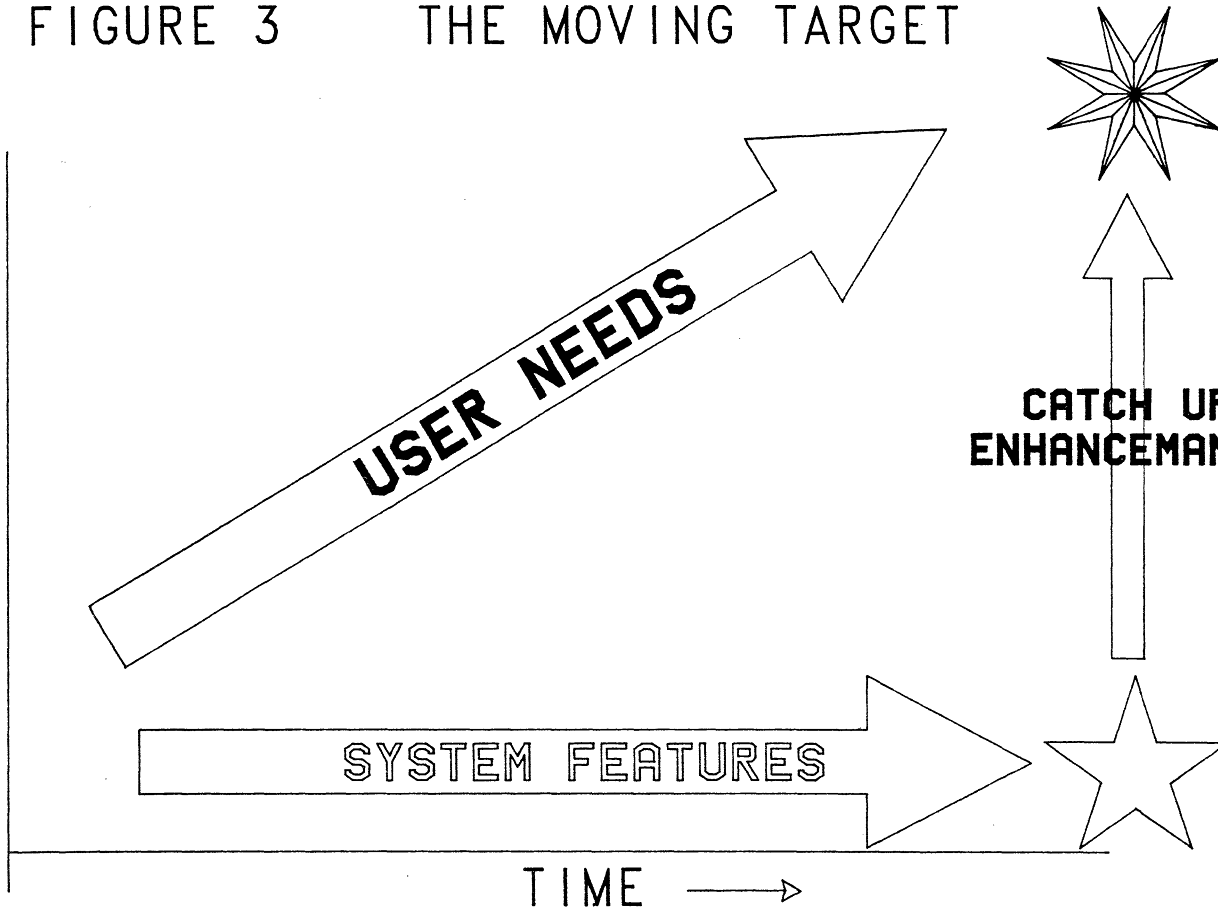
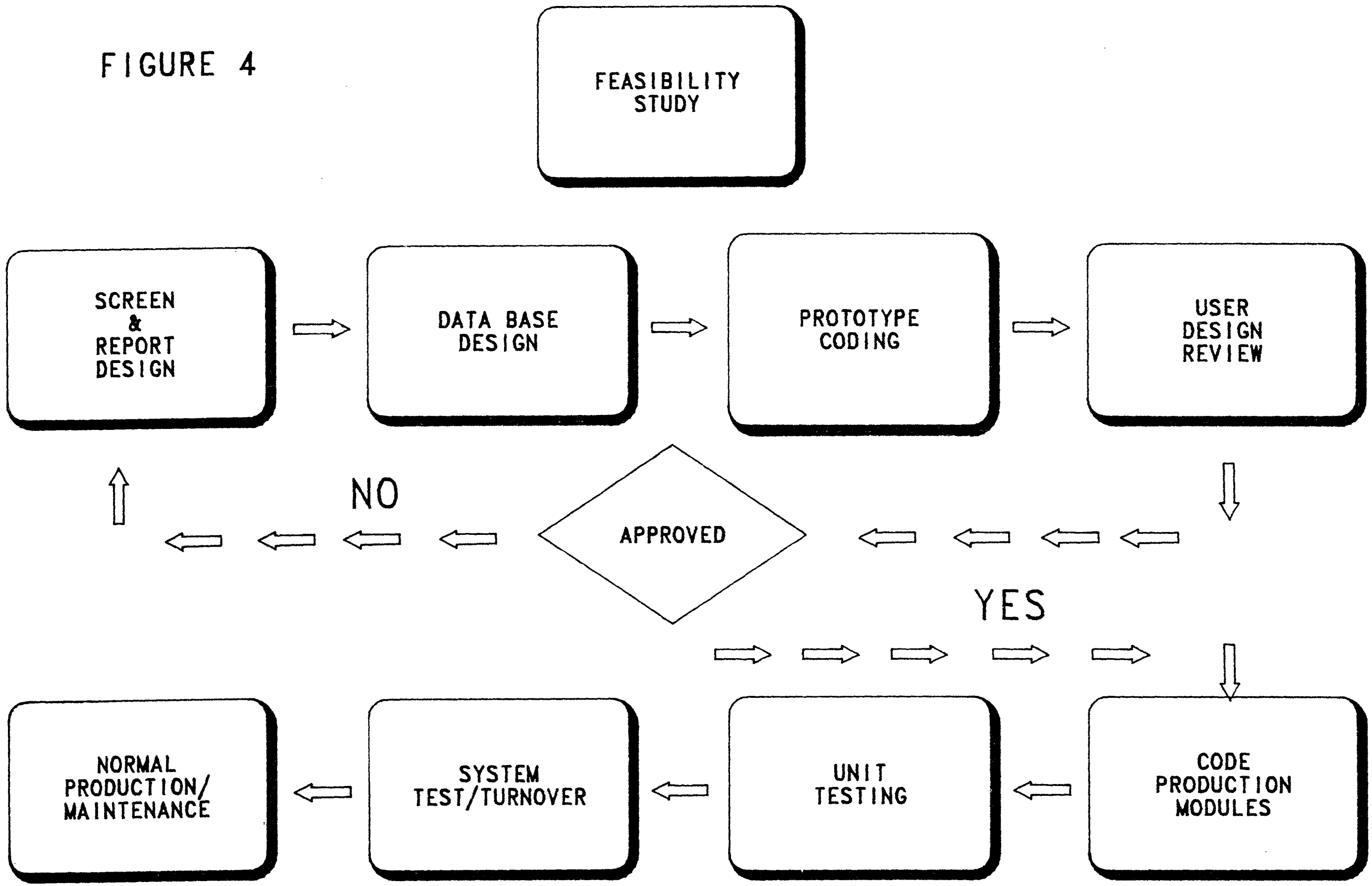


FIGURE 4



REVISED SYSTEM DEVELOPMENT METHODOLOGY

The ability to rapidly produce working models of the system allows the user to try out his ideas and to separate true needs from unnecessary wants very well. It changes the situation shown in Figure 2 to the one in Figure 5.

There is never a long time lag between a statement of user needs and a system model that lets him determine the soundness of his thinking at that point.

Specifically, the user and system designer build a series of V/3000 screens and sketch out major reports. The data elements are then entered into DICTIONARY/3000. Various IMAGE schemas are then created and analyzed using DICTDBC. TRANSACT/3000 modules are then prepared to exercise the screens and the data base. Reports can be produced via INFORM and REPORT. As the user sees the screens and reports, he can add, modify and delete data elements by merely indicating the changes he desires in those end products. The system designer can then work with the data base designer to ensure that the information is stored efficiently. Rather than consuming months or years, each of these iterations takes only a few days.

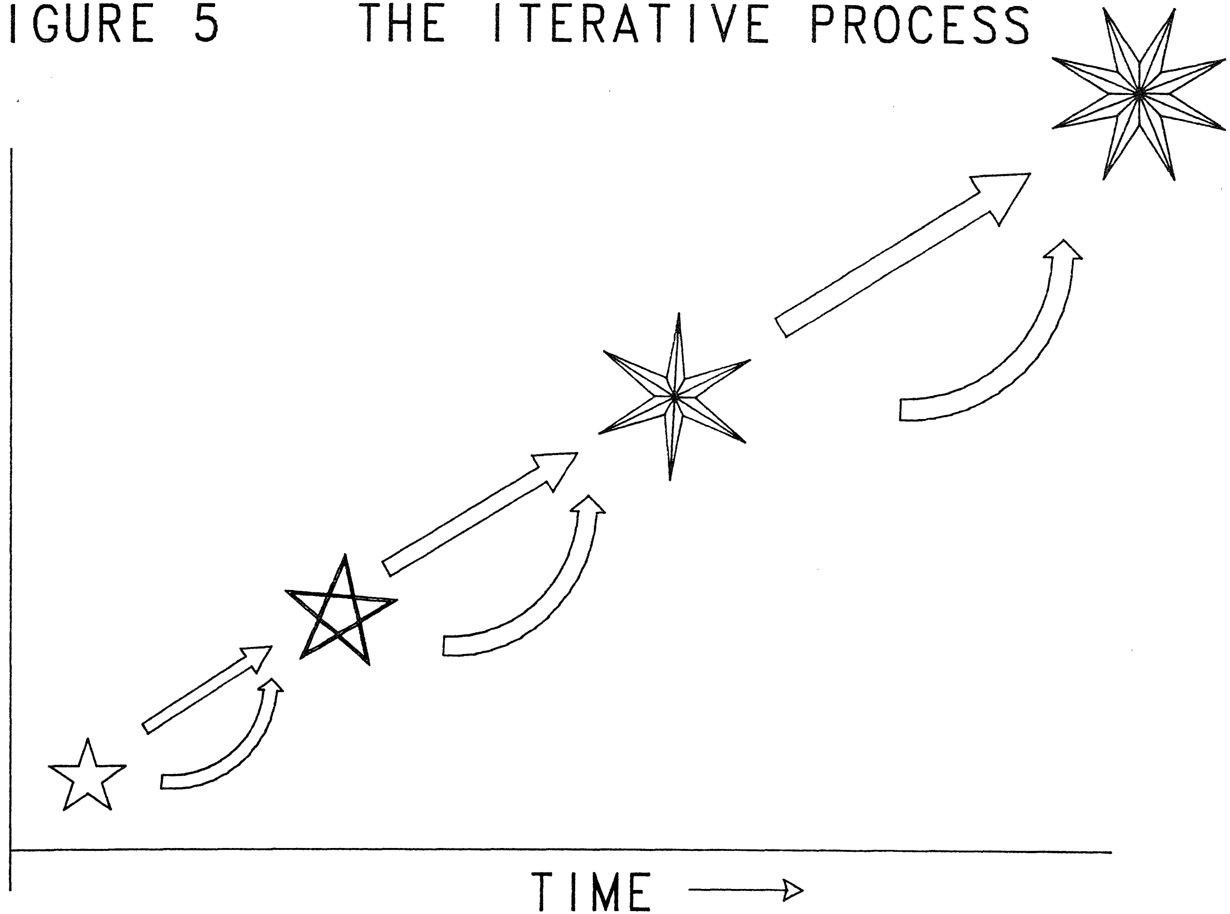
* Personnel Changes

In the case, we are told of one actual personnel change (the sales manager) and one potential change (the lead programmer). One obvious reason for the difficulties experienced in this project is the different sets of system requirements expressed by the original system sponsor and by his successor. The quick response capabilities of the RAPID family (and other fourth generation software) minimizes the disruption caused by the (inevitable) personnel changes that occur during the course of system development. As we can see from the case, these personnel changes occur in both the user and systems communities. The simplicity of these tools allows new technical people to step in and complete work that others have started without an enormously costly learning curve. When one considers the 25% - 35% annual turnover of systems personnel, this factor assumes the importance it deserves.

* Lack of Communication

This factor cannot be completely alleviated merely by the use of RAPID or similar tools. However, it is my contention that part of the communications gap that opens between systems people and users is the lack of a common set of language elements. The user is discussing money and inventory levels while the systems staff is thinking in terms of data base structures and transaction types. Tools such as TRANSACT, INFORM and V/3000 can bring the two groups together very quickly so that they can concentrate on the business problem at hand. Of course, proper user liaison is also required and it is clear that is it lacking this case.

FIGURE 5 THE ITERATIVE PROCESS



* Inflexible Software Tools

Those of us who have had to modify a COBOL program in order to accomodate a report format, or worse still, a file or data base change, can appreciate the frustration of the lead programmer in our case. COBOL, developed in the early 1960s from precursors written in the late 1950s is very cumbersome and wasteful of personnel resources. In its defense, it was a great improvement over what had gone before. However, the most common resource constraint today is not machines, but skilled people to use them. Therefore, we should use tools that maximize this scarce resource. The shorter development times that can be achieved with RAPID and its counterparts not only result in the user getting his system more quickly, they also mean that a fixed number of systems people can give more users more systems in a fixed time period. This can eliminate the backlog of user requests that plague most information systems departments. Getting back to our case, the new sales manager could be given a two-hour course in INFORM and he could then experiment with reports to his heart's content, leaving the technical staff free to concentrate on more substantive matters. An incidental benefit is likely to be improved morale among the systems and programming personnel and a greater feeling of system and data ownership on the part of the user community.

* Lack of Proper Project Management

This factor cannot be overlooked in assessing the reasons that this particular system project is in serious trouble. It is not unlikely that improper project management is a contributing factor in most failed systems projects. RAPID can help us even here, however. The very quick turnaround times combined with sound modular program construction standards allows a project manager to set a series of short duration tasks for his project staff. Instead of an assignment of several weeks to code an entire program, a programmer can be given a TRANSACT module due in two or three days. By breaking the job into very small work packets, progress can be monitored closely and corrective action can be taken quickly when required.

* Ineffective Management

The unbelievable lack of leadership by the MIS manager ("Gee, I didn't realize the situation was so bad.") cannot be alleviated by RAPID or any other software package. The point to be made here is that when good systems management is combined with committed users, a competent systems staff <u>and</u> powerful software tools, the information systems function can perform up to its potential. Although many systems have been successfully developed using "classical" tools such as COBOL, it is my opinion that information systems resources can be more effectively managed and utilized if development tools such as RAPID/3000 are adopted.

EPILOGUE - ONE YEAR LATER

Mike Adams, the Director of Data Processing at the Wilson Manufacturing Company, was holding his weekly project status review meeting with Linda Rogers, the firm's Systems Manager.

"How are we doing, Linda?"

"We're in good shape, Mike. I think we can accomplish our 12-month plan on time with the current staffing levels. Since we started using TRANSACT as a programming language and let the users design their own reports, our people have really become productive and the users feel like they are a real factor in the process. By the way, have you heard from Joe Ericson, lately?"

"Yes, he seems to be doing OK as the building maintenance supervisor. Now, how is the data base design for our strategic planning model coming ... ?"

So ends our parable. Although use of fourth generation tools such as RAPID/3000 is not a panacea, it can help the information systems function become an asset rather than a bottleneck in the success of any organization.

BIOGRAPHICAL SUMMARY

Stephen M. Rush is a Vice President in the Western Region of Marsh & McLennan Associates where he is responsible for Information Systems. He is also Director of Information Systems programs in the Graduate School of Management at Golden Gate University.

He holds a B.S. in Mathematics from the University of Rochester and an MBA from Golden Gate University. He also holds the CDP and is a Certified Management Consultant (C.M.C.) in the Province of Ontario, Canada.

Steve and his wife Heather reside in San Francisco.

PDS/3000 - A MAN-MACHINE INTERFACE

Ahmed Sharaf Eldin Ahmed
College of Engineering, King Saud University,
Saudi Arabia

Summary

This paper presents a system developed to help programmers and data entry personnel in developing programs and entering data on a HP3000 computer system. A preliminary version of this system was given in [1], however this new updated and enhanced version is named Program Development System - PDS/3000. The system is implemented on HP3000 Computer Systems. It is designed to minimize the user typing effort and increase the interaction with the computer. All the system commands are one character only. On-line help facility enhance the system. The modular design of the system allows expanding its capabilities easily. Five programming languages are supported by PDS/3000. Examples that illustrate the use of the system are given and discussed.

Background

One of the challenges in computer systems use is the man-machine interface. The only reliable means for program development and most of data entry is by keying in using a keyboard of a terminal or other data preparation equipment. It is true that some data could be read by other methods like OCR (Optical Character Recognition) or MICR (Magnetic Ink Character Readers) systems but it is safe to say that only a very small portion of data and programs are entered using these methods[2]. This could be due to the fact that character recognition is successful for printed english text with almost 100% success, while for constrained hand printed english the ratio of success is around 90% only. However, for handwritten english, no machines are available that can recognise unconstrained handwritten english. Most of other languages, like Arabic, do not have document reader equipment at all.

Another method for man-machine interface is the use of voice synthesizers and recognisers. However, these methods could be used in very specific situations and to do particular jobs. Another reason for the use of keyboards as the main man-machine interface is that they are still cheaper than other methods. In a word, the two main reasons that make keyboards the dominant man-machine interface are: (1) Current state of technology and (2) Economic reasons.

Program Development Cycle

Program development cycle is as shown in figure (1). The programmer uses a text editor to create a source module, then use a compiler to compile the source module and produce an object code. Afterwards, uses a link editor (=segmenter) to produce an executable code. If errors are found during compilation, which is very normal during program development, the editor is used to modify the source program and the cycle of re-compilation is done again until an error free compilation is realised. One exception of the above mentioned cycle is the case of interpreters like the BASIC interpreter where the source statements are checked immediately one by one. During program development using the text editor there are some facilities provided to help programmers in entering their source code. These could be summarised for the EDIT/3000 in the following[3]:

1. COBOL format for COBOL programs. Using this facility, the user is relieved from numbering his source statements, the cursor is positioned at column 7 automatically.

2. Free Format for FORTRAN programs. This facility is not an Editor one, but a compiler command, which directs the fortran compiler to accept the fortran source in free format mode. However, the resulting source program is not a standard one.

3. Some other editing facilities which may help indirectly in program development.

Since the text editor is a general subsystem which could be used for any text, it implies that it is not a program development oriented. Moreover, it is designed to work on all types of terminals (Teleypes, CRT, ...) which implies that more powerful terminals (CRT, Memory, ...) will not be fully utilised. One can summarise the following points as major difficulties in developing programs interactively:

1. No checking is done on the entered source statements. The program source is only checked during compilation. This will result in a time consuming process, since the user must first terminate the editing stage, keep his source text in a disk file, then compile the source text by using the compiler subsystem. To illustrate, assume that a COBOL program is developed and a division name was mis-spelled, then most probably, the first compilation will be of no use except for the correction of the mis-spelled name.

2. No shorthand could be used automatically, for instance; if a text like this "CALL ADOMULT (X,Y)" is to be repeated several times in a program with different arguments, it should be entered as such each time.

3. To test your source program, terminate the editor subsystem and start compiler subsystem. If error occurs and correction is required, then, you will have to terminate the compiler subsystem, enter the editor subsystem copy the course text in a work file again to start editing which is a time consuming process.

Program Development System Objectives

The Program Development System is an interactive system designed to help the users in developing their programs. Its main functions are to minimize the user input, make the program keying in easier and more efficient. It is not intended to replace the Edit/3000 subsystem but to complement it as a subsystem oriented to program development and data entry. PDS/3000 consists of a family of commands which facilitate program and data entering, make some validation of program structure (in COBOL). The computer prompt is ">". The command set is chosen to be one alphabet character. The user needs not to press the return key each time he enters a command, because PDS/3000 will automatically start the command processing as soon as a key is pressed. PDS/3000 makes extensive use of the 8 soft keys of CRT terminals. Up to eighty characters could be assigned to each key. These assigned values are permanently displayed on the screen, so the user need not to memorise which is which.

PDS/3000 supports five programming languages, "BASIC, COBOL, FORTRAN, RPG II, and SPL (The HP 3000 System Programming Lanugage). Default values for the eight keys has been defined. The default values are selected according to a statistical study for a big collection of programs written in COBOL, FORTRAN, and BASIC, while the other two languages, RPG II and SPL, were chosen arbitrarily. In this statistical study a program was written which counts the keywords in each program and the relatively freqnency of each of these keywords in the program text is calculated. The results were as follows:

```
BASIC   :  [PRINT, GO TO, GOSUB, INPUT, IF, THEN, FOR, NEXT].
COBOL   :  [GO TO, MOVE, TO, PERFORM, WRITE, DISPLAY, ADD,      ].
FORTRAN:   [GO TO, IF (, FORMAT (, READ (5, WRITE (6, CONTINUE,
            CALL,         ].
RPG II :   [ADD, COMP, DSPLY, EXSR, GO TO, MOVE, TAG, XFOOT].
SPL    :   [BEGIN, END, PROCEDURE, ARRAY, GO TO, ASSEMBLY, BYTE, PCAL]
```

PDS/3000 Commands

15 commands are available in PDS/3000. These commands can be classified into the following groups:

(1) Key assignments:

The values assigned to keys are defaulted according to the selected programming language. However, these values can be modified to best accommodate the user requirements using the >M command. Using this command the user can either retain the current key value by entering a "/" or change the value by typing in the new value to be assigned to key and then the terminator character '/', If the key is required to be defined as an executable key, the '\' is used as a terminator. The values assigned to keys can vary from 1 to 80 characters. It is possible to save these values in a file, PDS file, using the >A command. Subsequently these values can be retrieved again using the >G command. PDS files are normal files formatted by PDS.

(2) Executing Programs:

One of the features of the PDS/3000 is the ability to execute programs under PDS control. Four commands are given as follows:

(i) >C: This command is used to run the appropriate compiler i.e. to compile a source program. Since PDS knows which language is currently selected by the user, it means that the correct compiler will be invoked. The user is asked whether to get compiler listing on printer or screen. In case of BASIC, the BASIC interpreter will be invoked. Having the 'C' command will save the programmer time, since it is not required to terminate the PDS subsystem to check the Program Symtax. Moreover, checking of program segments is possible in an easy way using the 'C' command.

(ii) >S: This command will invoke the EDIT/3000 subsystem.

(iii) >Q: This is used to invoke the Query/3000 program.

(iv) >R: This command is used to run any program provided that the correct access is granted. Usually this command is used to run user programs. However, it is possible to run subsystems like Edit/3000 or Query using this command. The user is prompted for the name of the program to be run. If the program is not in the user logon group; the group is required. If it is outside the user account, then the full qualified name is required.

(3) Communication with the operating system - MPE

To invoke the PDS/3000 the user type: PDS and to terminate the PDS the PDS command >E is used. However, it is possible to terminate PDS while keeping the values assigned to keys; this is done by using the >K command. In order to execute a MPE command while in PDS the >: command is used. The user is prompted for the required MPE command. It should be noticed, however, that not all MPE commands could be executed using this command. The ability is dependable upon the programmatical execution of the required MPE command[4]. In all cases, a message will be displayed showing the cause of any errors found.

(4) PDS utilities

Five general commands are provided; these are:

(i) >H: An on-line Help is displayed on the screen that lists the available PDS/3000 commands and their functions.

(ii) >F: This command is used to fix the values assigned to keys permanently on the terminal screen. F is a PDS default. However, this will decrease the number of available lines on the screen. The key values are displayed in three different formats according to the max. length of keys.

(iii) >U: This command is used to Unfix the key values from the screen. This is the contrary to >F. Using >U, the full screen will be available to the user.

(iv) >L: To change the currently selected programming language. The user can at any time change the current programming language among the currently supported languages.

(v) >I: This command is used for COBOL programs only. It will initialize the 'standard' part of a COBOL program automatically. The user is prompted for the file name that to hold the COBOL source program. PDS will copy to this file the standard part of COBOL program. Although the COPY verb could be used within COBOL programs to copy portions of cobol programs from a COBOL library but using the >I command is better for the following reasons:

(1) No need to build a Cobol library for standard parts.

(2) Initialized Cobol file will contain all the 'static' parts of a Cobol program. These information can be changed if required.

EXAMPLES

A) A FORTRAN Example:

```
      READ (5, 11) X, Y
   11 FORMAT(2F8.3)
      Z = X + Y
      WRITE (6, 21) X, Y, Z
   21 FORMAT (2X, 3F10.3)
      STOP
      END
```

Entering this program using PDS, then the keywords: 'READ (5,', 'FORMAT(', 'WRITE (6,', and all leading spaces will be done by one keystroke only. Now the number of key-strokes required to enter this program in the standard way are: 10 upper case letters, 45 spaces and 65 lower case letters. While the number of keystrokes required to enter the same program under PDS are 5 upper case letters, zero spaces and 38 lower case letters.

The comparison is clear and shows a reduction of about 63% of the required keystrokes.

B) A COBOL Example:

```
IDENTIFICATION DIVISION.
PROGRAM-ID. SAMPLE.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.
  HP 3000.
```

```
OBJECT-COMPUTER.
  HP 3000.
DATA DIVISION.
WORKING-STORAGE SECTION.
77 X PIC 9(3).
77 Y PIC 9(3).
77 Z PIC 9(4).
PROCEDURE DIVISON.
P1.
  ACCEPT X ACCEPT Y.
  ADD X Y GIVING Z.
  DISPLAY Z.
```

Entering this program using PDS, the keywords: 'PIC (9', 'ACCEPT', 'ADD', 'DISPLAY' and all leading spaces will be done by one keystroke only. The number of keystrokes required to enter this program in standard way are 6 upper case letters, 162 spaces and 226 lower case letters while the number of keystrokes required to enter the same program under PDP are 3 upper case letters, 10 spaces and 43 lower case letters. The comparison shows a reduction of 85% of the required keystrokes.

C) A DATA ENTRY APPLICATION: Student registration

PDS could also be used to facilitate data entry functions. This is particularly useful in data entry programs in system like: inventory, personnel, student registration and all applications with huge data entry. This could be accomplished by assigning frequently used entries to keys, thus, such entries could be entered by one keystroke only.

As an example; in an online student registration system designed by the author we define frequently selected courses as key values. Thus if at a certain moment we expect that majority of students will select courses names like: GE 304, MA 501, CE 302, EE 313 ... etc. Then students choose one of these courses will be processed easier and consequently faster. Moreover, two terminals in this system are used for on-line query purposes. Assigning helpful query commands to keys and using the >Q command will make retrieving of information much easier.

D) A DATA Entry Application: Inventory System

Usually items in inventory systems are grouped together. Thus item number will comprise of a group-id and an item number within this group. Assume that in an inventory system the items to be processed are all of the group GA01 and that the descriptions of these items contain keywords like: "PIPE", "DIAMETER", "INCHES" and "LENGTH" and assume that the units of measure used are: "METERS", "UNITS" and "KGM".

Now consider the following inventory data entries:

Item No. = GA01 001

Description = PIPE 3 INCHES DIAMETER, 6 METERS LENGTH

Units = METERS

Typing the above data in the standard way requires 55 keystrokes. However, using PDS/3000 the number of keystrokes will be reduced to 15 only, i.e. reduced by 73%.

Notice also that the key assignments could be changed at any time to suit the data being entered.

E) Executing Several Similar MPE Commands

Sometimes it happens that a user would like to execute a number of MPE commands several times but with different parameters. This type of operations is greatly simplified using PDS/3000. Suppose that a user likes to 'clean' his file domain (group). The "PURGE ∇" command can be used several times to do that. Instead of typing it each time or typing "REDO" and edit the file names it is much easier to assign "PURGE " to a key and use the ":" PDS command. If the family of files to be purged share common subword then it is possible to assign this subword to another key or the same key. To illustrate suppose that a user would like to purge the files:

WORK001 WORK002 .... WORK015

Then key could be defined as:

"PURGE ∇ WORK0" and deleting any file is done by pressing that key and the numeric suffic of the file i.e. 01, 02, ... etc.

It should be noticed that the :REDO command is not as efficient in such operations due to two reasons. First, REDO itself needs typing; it is a command, and needs editing to form the correct command and pressing of the RETURN key to terminate the editing. Thus the number of keystrokes is still high (sometimes more than re-typing the command itself). Second, REDO is valid for the last command only; thus repeating a block of commands is not possible. Although the use of UDC" could be an alternative sometimes but it is not as easy as using PDS.

Conclusion

A program Develpment system has been presented and demonstrated. The system is programmed in SPL and FORTRAN/3000. Most of the code was prepared using PDS itself. The users' reactions to this system in our University is encouraging further developments and enhancements.

Acknowledgement

The solores program, solar air conditioning project, King Saud University are deeply appreciated for their help during preparation of this manuscript.

References

(1) A. Nouh and A. Sharaf Eldin, "Program Development Package", 6th Nov., 1980.

(2) J. Ullmann, "Advances in Character Recognition", in Applications of Pattern Recognition, ed. K.S.Fu, CRC Press, 1982.

(3) EDIT/3000 Referecen Manual, HP Publication.

(4) MPE Reference Manual, HP Publications.

Biography

Ahmed Sharaf Eldin Ahmed
is the System Manager, Computer Center of the College of Engineering, King Saud University. More than 18 years of experience in computers. Developed several types of software for business, scientific and engineering applications. Obtained the Cairo University Prize in 1971. A co-inventor of a programmable calculator 1980. Published several technical papers. Authored a book "Fundamentals of Data Base Systems" in 1984.

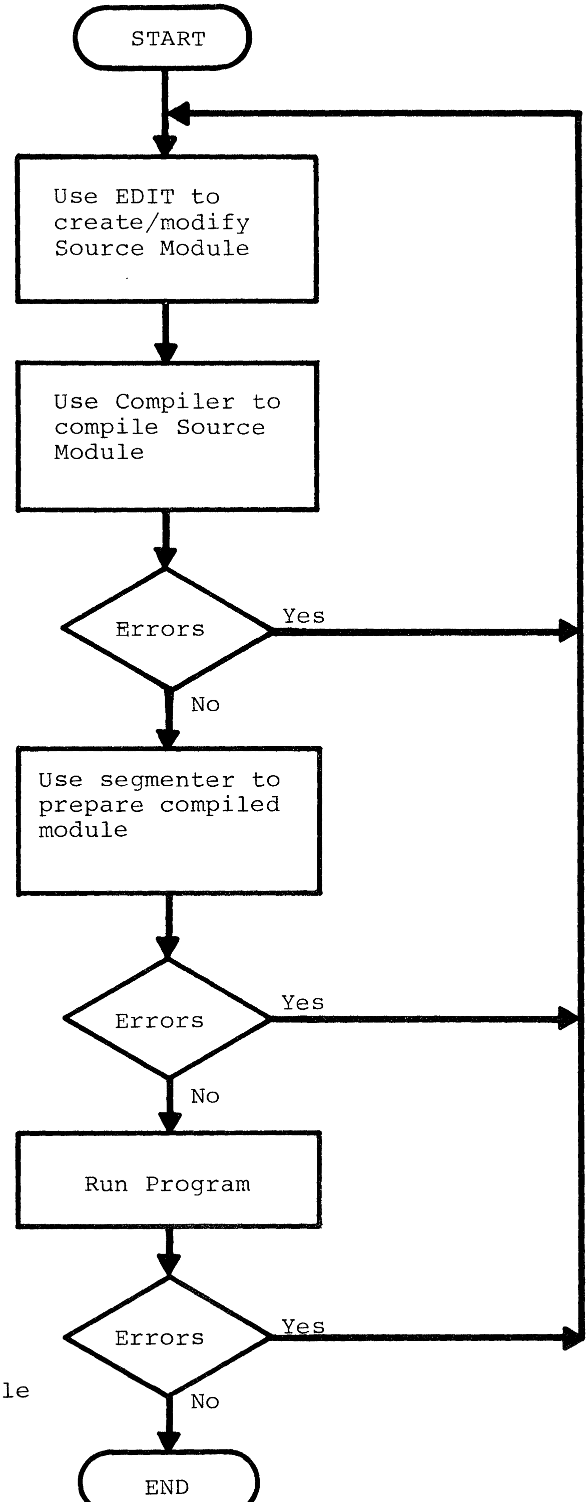


Figure (1).

Program Development Cycle

```
IN THE NAME OF ALLAH MOST MERCIFUL, MOST GRACIOUS

Program Development System        WED, DEC 26, 1984,    8:30 AM

                         P  D  S

(C) Ahmed Sharaf Eldin Ahmed.     SEPT 84.
Type  >H for aid
Available Programming Languages are:
B=BASIC                C=COBOL
F=FORTRAN              R=RPG II
S=SPL
Type in language code (B,C,F,R or S)?
```

Key Values

```
f1=/F FSTUDENT\                     f2=/R RSTUDENT\
f3=/F FCLASS\                       f4=/R RCLASS\
f5=/F FCOURSE\                      f6=/R RCOURSE\
f7=/F INFORMATION.DONE-BY="/        f8=/;END\
```

```
>Q
LOCKWORD: QUERY.PUB.SYS?

HP32216A.04.03 QUERY/3000  WED, DEC 26, 1984,  8:50 AM


QUERY/3000 READY


>
```

Key Values

/GO TO /	/IF (/	/FORMAT(/	/DISPLAY "/
/WRITE(6,/	/CONTINUE\	/CALL /	/ /

THE AVAILABLE COMMANDS AND THEIR FUNCTIONS ARE AS FOLLOWS:

```
>:   EXECUTE MPE COMMAND UNDER PDS CONTROL.
>A   ASSIGN KEY VALUES TO A PERMANENT DISC FILE (PDS FILE).
>C   COMPILE A SOURCE PROGRAM UNDER PDS CONTROL.
>E   END THE PDS SUBSYSTEM.
>F   FIX THE VALUES ASSIGNED TO KEYS PERMENANTLY ON THE SCREEN.
>G   GET KEY VALUES FROM A PDS FILE.
>H   HELP THE USER BY INFORMATION ABOUT THE OPERATION OF PDS.
>I   INITIALIZE A COBOL PROGRAM.
>K   KILL THE PDS SUBSYSTEM WHILE RETAINING THE KEY VALUES.
>L   CHANGE THE CURRENTLY SELECTED PROGRAMMING LANGUAGE.
>M   MODIFY THE VALUES ASSIGNED TO KEYS.
>Q   RUN QUERY UNDER PDS CONTROL.
>R   RUN A PROGRAM UNDER PDS CONTROL.
>S   SWITCH TO THE EDITOR/3000 SUBSYSTEM.
>U   UNFIX THE VALUES ASSIGNED TO KEYS ON THE SCREEN.
```

WHAT ARE EXPERT SYSTEMS?

Hitesh Vekaria

VRS Software Ltd., London, UK

Summary

Usage of computers until recently was restricted to a limited domain of our lives, i.e. businesses and large research projects. The contributing factors were the emormous hardware and software costs for a machine with very limited functional features. However, the ever decreasing costs (at least in hardware) of more versatile and powerful computers opened doors to the corner of the market which was as yet untapped: namely, small businesses and the average person on the street. History is about to repeat itself. This time it is due to the achievements made in the software field. These achievements in software techniques can now help to develop the largest market which is the representation of human expertise, whether it be in the medical field, architecture, law or basic car maintenance tasks. This briefing is all about such software techniques, better referred to as Expert Systems or Knowledge Based Systems.

Introduction

At present throughout the computer society Expert Systems are very much in the spotlight, representing a new and important concept in the future design of computer based systems.

They are the most viable commercial 'spin-off' from the field of artificial intelligence, a system that embodies organized knowledge concerning some specific domain of human expertise and is capable of performing as a skillful and cost-effective consultant - specialist knowledge at an arms length for the 'man in the street'.

This briefing will describe the basic architecture of Expert Systems and how they are constructed and programmed. It will also provide information on current off-the-shelf systems; how they might offer solutions to problems that cannot be solved by conventional methodologies.

The beginning

There have been three significant events to-date in the history of mankind - first, the creation of the Universe, second, emergence of the first living organism, and the final, the idea of constructing an intelligent artifact.

Artificial Intelligence (AI) has long been the pet fascination of man, stretching as far back as the legends of the Pygmalion, followed by various blood-curdling tales of golems, talking heads and notably Frankenstein's monster (see Aldiss, 1975; McCorduck, 1979), and most recently, the obsession of a machine with human attributes (intelligence, innovation, etc), attributes which separates us as superior beings from the ones with whom we share this planet earth.

The research of AI has never been so intensive as in the last 30 years, due to the coming of the computer age. The areas of AI spanned from early days of Neural Nets (Rosenblatt, 1957) to Heuristic search (Ernst and Newell, 1969), but always with very limited success. This was mainly due to the technological barriers - as claimed by AI researchers. Several having realized the mammoth task of creating a mechanized Frankenstein's monster reverted to concentrating on less ambitious and domain specific problems, thus the coming of the 'Knowledge Representation' paradigm - building computer systems in limited domain of expertise with knowledge elicited from human experts. The great granddaddy of them was the MYCIN system developed at Stanford University in the 1970's (Shortliffe, 1976) whose characteristics are still to be found in systems emerging at present.

So what is new?

Expert Systems or Knowledge Based Systems as they are often referred to, are classified as; systems which embody organized knowledge concerning human expertise combined with its inference mechanism, which enable them to use this knowledge to perform as effectively as a human expert to offer advise or take decisions.

It can be argued that expert systems are not the first computer systems to aid decision making. There have been various statistical packages developed with mathematical techniques (Bayes Rules) for medical diagnosis, educational assessment, personnel selection and resource planning with great many successes. However, what differentiates such systems is that expert systems emphasise qualitative, logical reasoning not quantitative calculations as in statistical packages.

Following are some of the features which have become hallmarks of expert systems:

- Organized knowledge representation
- Explanation of its workings
- Management of uncertainty
- Reformulation.

a) Knowledge Representation

One of the vital land marks in devising expert systems is the knowledge representation formalism. Earlier systems such as MYCIN encoded knowledge in the form of production 'rules' and simple lists of 'facts'. The rules relate to, possible conditions to associated interpretation.

Example: For more details see Alvey (1983).

STRONG-ANALGESICS

TRANS: (my list of strong analgesic drugs)
value: (DEXTROMORAMIDE DIAMORPHINE .... PHENAZOCINE)

RULE 034

If: 1) The current drug is one of: my list of strong analgesic drugs, or
2) A: The current drug is pethidine, and
B: The route of administration of the current drug is on of: im iv

Then: It is definite (1.0) that the current drug is a strong analgesic drug.

Fig 1: EMYCIN stores knowledge in the form of production rules and simple list of facts. EMYCIN is an expert system shell developed from the successful MYCIN system.

These rules are tested against available data or data requested from the user. If appropriate, it tries to infer the truth or falsity of conditions from other rules. In the above example it may refer to conditions which deal with variables im and iv.

b) Explain its workings

One of the characteristics of an expert system that set it apart from conventional systems is that it can justify and explain its reasoning. A user can interrogate the system to explain why certain conclusions were drawn or to the nature of why a question is being asked during a consultation. This in effect, is a by product of the formalism (rule-based) use to encode the rules and facts. Whenever an expert system is asked to explain itself all that will be required is to reiterrate descriptive trace of the facts and the rules which led to the questions.

The following example illustrates an interaction between an expert system and a user for providing advise and information to potential claimants of social security benefits. The system was constructed using a Shell (APES) developed at Imperial College London, using PROLOG (logic programming language).

Example: Interaction between user and an expert system, see Hammond (1983) for more details.

Is (peter is-entitled-to sup-ben)
What is peter's sex? Male
How old is peter? 23
Is peter in full-time education? Why

At this stage the user queries the system to explain what relevence does the question bare with the original goal.

Initial explanation ...

If peter is not in full-time education
then peter's educational status is OK

It then goes on further to explain what relevence is the question to the original goal ...

If peter is not disqualified by sex and
peter is not a juvenile and
peter's educational status is OK and

peter needs financial help and
peter is not disqualified by being in a trade-dispute

then peter is entitled to sup-ben

which is what you originally asked about

> Having explained the reasoning it rephrases the question ...

Is peter in full-time education? No

How much saving does peter have? 2600

> It finally arrives at an answer ...

No, I cannot confirm peter is-entitled-to sup-ben Why

> The user wished to know why it cannot confirm ...

peter is-entitled-to sup-ben could be deduced from the rule

<person> is-entitled-to sup-ben if
not <person> is-disqualified-by sex and
not <person> is-a-juvenile and

<person> needs-financial-help and
not <person> is-disqualified-by trade-dispute

> Having shown the rule as encoded by the expert, it continues to show the related result of each condition in the rule ...

peter is-disqualified-by sex is untrue
peter is-a-juvenile is untrue

but peter needs-financial-help is not true Why

> At this stage the user wishes to find out precisely why the condition is set to what it is ...

peter needs-financial-help could be deduced from the rule

<person> needs-financial-help if
capital of person LESS max-capital for sup-ben and
requirements of <person> at-least resources of <person>

```
but capital of peter LESS max-capital for sup-ben is not true Why

peter capital 2600
sup-ben max-capital 2500
2600 LESS 2500 is not true
but LESS is a built-in relation, so I cannot help further
```

c) Management of uncertainty

Experts frequently have to operate in conditions where they do not have all the required data or the knowledge for its application. In such conditions they have to make do with whatever resources are present to derive at a 'Good Estimate' of the solution. If expert systems are to tackle worthwhile tasks, then they must also face the problem which arises when incomplete or incorrect evidence is presented. Early researchers realized the necessity of integrating such an option to deal with uncertainty. The most influential model for reasoning with uncertainty was employed in MYCIN. It uses a scheme devised by Shortliffe based on 'Certainty Factors' on a scale of +1 (definitely right) to -1 (definitely wrong).

```
Example: A typical rule in MYCIN with uncertainty of any degree is
         stated as follows:

Rule 0-47

If:     (1) The route of the administration of the penicillin is
            oral, and
        (2) There is a gastrointestinal factor which may interfere
            with the absorption of the penicillin.

Then:   There is suggestive evidence (0.6) that the route of
        administration of the penicillin is not adequate.
```

Such 'Certainty Factors' associated with both rules and facts are propagated through an inference network to evaluate a measure of belief in hyptheses.

d) Reformulation

Some of the critical tasks experts perform which are now being approached in expert systems, is to take a problem stated in some arbitary initial form and convert it into the form appropriate for processing by the expert rules. These can be forms of data collected to find appropriate patterns and act accordingly in the domain concerned by the system (Mostow, 1983).

Components of Expert Systems

The Knowledge base and inference engine are usually the two attributes of expert systems much talked about. However, the majority of expert systems consists of:

- Knowledge base
- Inference engine
- Working-Storage
- Man-Machine Interface (MMI)
- Justifier
- Knowledge Acquisition Module

a) Knowledge base

Knowledge encoded onto the knowledge base is separated into two categories, procedural knowledge (rules) and declarative knowledge (facts). There also exists the third kind, control knowledge, which is beyond the scope of this paper. Rules are given in implicational form. In production rule formalism, they are expressed in familiar IF-THEN format. Rules typically express general knowledge about a particular subject area.

Example: Rules for stating relationship between various family members can be expressed as production rules.

Rule 01

If: Father-of <X> is <Y>, or
Mother-of <X> is <Y>

Then: Parent-of <X> is <Y>

Rule 02

If: Father-of <Z> is <X>, and
Parent-of <Y> is <Z>

Then: Grandfather-of <Y> is <X>

Facts on the other hand are not expressed as implications. Typically they represent specific knowledge relevant to a particular case. Facts could be stored on the knowledge base or volunteered by the user at the consultation stage.

Example: Continuing our example of family relationship, we can express simple facts which could be used by the rules to derive more complex results.

Facts

Father-of JULIE is JOHN
Mother-of MARY is JULIE

Given the rules and facts, we can deduce through a simple inference mechanism that Grandfather of MARY is JOHN, since Parent-of MARY is JULIE and Father-of JULIE is JOHN.

There are a number of other alternative formalisms to express knowledge. The most popular ones are, decision tree, semantic net and predicate calculus, each suited to the nature of knowledge being expressed.

Example: PROLOG makes use of predicate calculus to express knowledge.

Facts

Father-of (John, Mary)

Rules

Grandfather-of (X, Y):- Father-of (X, Y), Parent-of (Z, Y)

b) Inference engine

An inference engine is a mechanism which interrogates the knowledge base and infers deductions based upon the stored knowledge and also upon the information volunteered by the user. The process of the interrogation involves interpreting the rules which in turn could lead to other rules (nodes) on the network or the decision tree.

In a goal-driven system (backward chaining), it starts at the node representing the most definite category of the unknown object and tries to find sub-category of the current node to become the next node. In order for a rule (sub-category) to succeed, all its conditions must be satisfied. The process is continued (descending a tree) until an end-node (leaf of the tree) is reached representing the successful classification of the starting category or no further inference steps can be made from the current data.

A number of alternative methods are also used besides backward chaining. These are, forward chaining, where the evidence supplied by the user leads to a hypothesis; combined forward and backward chaining, this method is widely practised in systems where the network can be hierarchically partitioned.

c) Working-Storage

Working-storage (blackboard) is used for recording intermediate results during the consultation. This could be in terms of hypotheses and decisions that the expert system manipulates. It could also record various strategic information, such as recommendations for following the primary hypothesis, refining and elaborating each of the hypothesis until one best hypothesis emerges.

d) Man-Machine Interface

MMI is a processor for problem oriented communication between the user and the expert system. Typically the processor parses and interpretates user questions, commands and volunteered information. Conversely the processor formats information gathered by the system, including the answer to questions, explanations and justification for its behaviour and request for data.

e) Justifier

Primary function of the justifier is to explain the system's behaviour. This involves interrogating the knowledge base and the working-storage for information which could be useful to explain to the user why certain conclusions were drawn and maybe also to reveal midway through the consultation the plan to test various hypotheses.

f) Knowledge Acquisition Module

One of the real bottlenecks which the knowledge engineers face when developing expert systems is the elicitation of knowledge from human experts. Many researchers in this field have approached the problem by using the induction process, whereby, the users supply samples of problems with known solutions. The system will try to derive some kind of pattern which leads to the stated solution and will then devise a general rule for it.

The task of knowledge acquisition module is to accept samples in some meaningful form which it can then interrogate.

What is a Shell?

As the fourth generation software is seen as a tool to develop application systems - proto-type, minimize programming burden, quick turn-round of system changes - an expert system shell (Shell) is seen in much the same light to the development cycle of expert systems. A number of vendors supply such shells which provide the inference and control mechanisms necessary for an expert system but are empty of knowledge. This provides an inexperienced user with a quick and convenient method for developing expert systems. All that will be required of the user is to encode the domain knowledge onto the knowledge base in the formalism (rule language) stated.

Example: Hayes-Roth, (1983) reinstate the formalism used by EMYCIN - MYCIN expert system but empty of knowledge - at the first approximation using Backus-Naur Form (BNF) as:

```
                <rule> ::= (IF <antecedent> THEN <action> (ELSE
                           <action>))
          <antecedent> ::= (AND {<condition>}+)
           <condition> ::= (OR {<condition>}+) /
                           (<predicate> <associative-triple>)
  <associated-triple> ::= (<attribute> <object> <value>)
              <action> ::= {<consequent>}+ / {<procedure>}+
          <consequent> ::= (<associative-triple> <certainty-factor>)
```

The user of such a Shell will encode the rules and facts elicited from the expert in the given format onto the knowledge base. The Shell has not only inherited the advantages of the fourth generation software but also its disadvantages. Namely, the inflexibility of declaring knowledge in other than the stated formalism, degradation of system performance and many others which we are all too familiar with.

The following lists shells supplied by various vendors:

TIMM - Developed and marketed by General Research Corporation.

ADVISER - Developed jointly by International Computers Ltd and System Programming Ltd International.

EX-TRAN7 - Developed and marketed by Intelligent Terminals Ltd. Interesting point about this package is it generates Fortran-77 code for production system.

Uses of Expert Systems

Expert systems have recently been exploited in a wide variety of tasks. The following are the most successful ones.

Interpretation - Whether it is one spoken language to another or signals intercepted via receivers.

Diagnosis - A number of computer manufacturers have proto-type models which have been successful in trials to debug operating systems.

Design Planning - Architects use it as an aid to design planning.

Monitoring - Monitoring flow of fluid through complex piping network, to ensure constant flow of fluid and no pressure build-up at any point in the network.

Why bother?

In this information technological era it can be said that 'Knowledge is power' preceeding popular beliefs of the 1970's during the oil crisis 'Gold is black' therefore it must also be 'power'. Knowledge (especially expert knowledge) is an attribute which is acquired by humans through the process of 'burning the midnight oil', which only a few can either endure or have the ability to consume. They therefore have become a scarce and expensive commodity which only a few could afford - take for example, the person who wrote HP segmenter.

These are the precise reasons why institutions which once depended heavily on a single person can now turn to an alternative method of acquiring such expertise. Expert system technology gives us the tools of encoding knowledge of such scarce beings onto a microchip and supply it in bulk to whatever part of the world it might be required. And maybe only then we should be able to strive towards a society where poverty, unemployment, no inflation, less HP bugs outstanding ... will be a thing of the past.

References

Aldiss, B. (1975) - Billion Year Spree, Corgi, London.

Alvey, P. (1983) - The problems of designing a medical Expert System. Expert Systems 83, Cambridge.

Ernst, G. and Newell, A. (1969) - A Case Study in Generality and Problem Solving, Academic Press, New York.

Hammond, P. (1983) - Representation of DHSS Regulations as a Logic Program, Expert Systems 83, Cambridge.

Hayes-Roth, Waterman and Lenat (eds) (1983) - Building Expert Systems, Addison Wesley, London.

McCorduck, P. (1979) - Machine Who Think W.H. Freeman, San Franciso.

Mostow, D.J. (1983) - Machine Transformation of advise into a heuristic search procedure, In R.S. Michalski, J. Carbonell, and T.M. Mitchell, eds., Machine learning: An Artificial intelligence approach, Tioga Press, Palo Alto, California.

Biography

Hitesh Vekaria
is a Consultant with VRS Software Ltd, UK, involved in system development for clients, product development and support with responsibility for co-ordinating user requests into KNOWLEDGE, feasibility studies into the application of expert systems by companies together with the development of software tools for expert systems construction. Hitesh is actively involved in various associations for expert systems.

BS - BUSINESS SYSTEMS

USER-DIRECTED APPLICATION DEVELOPMENT

Stephen Belford
Rosenberg Capital Management, San Francisco, USA

Summary

This paper discusses the techniques used to develop a long-range user-oriented System Plan in the Financial Services Industry. The process details gaining user and management commitment, and active participation of 20% of the full-time non-DP professional staff. It emphasizes the orientation of end-users in analytic roles, utilizing techniques to enhance and streamline the development process in a highly understandable manner. The resulting applications developed, tested and implemented will serve to illustrate reduced development times, more ergonomic systems, consistency from one application to another, and most importantly, applications that meet the end-user's expectations.

Introduction

I joined Rosenberg Capital Management (RCM) over a year ago. At that time, RCM had been a highly successful business for fourteen years. We are in the business of Investment Management. There has been some form of automation at RCM for the last ten years. We currently have an HP 3000/48. HP hardware has been utilized at RCM for the last five years. There are in excess of 100 employees at RCM, 70 of whom have access to the HP. We exclusively utilize IBM PCs, XTs and ATs as terminals. All these devices have access to the HP through the use of terminal emulation software called PC2622A. All access occurs through the use of our Rolm Data Network accessible through a direct-connect of the individual's telephone.

When I started at RCM, my prime charter was to enable the firm to effectively utilize technology as a tool to enhance their growth and decision-making processes into the future. In order to identify the magnitude of this charter, it became clear early on, that this task required the creation of a Long Range System Plan that would encompass all areas of the business. I recommended that we undertake this project with a team which consisted of the following personnel:

- 2 Professional System Analysts (one of whom had 10 years of experience with RCM).
- 2 Programmer/Analysts.
- 1 Technical Documentation Specialist.
- 1 HP System Manager.
- 16 Business Users (including 5 of 17 Principals of the firm).

In order to gain broad-based User commitment as well as full support at the Senior Management level, the expected results were delineated as follows:

- Increased understanding of the business by the technical Data Processing staff.
- Long-Range Action Plan defining Systems, Priorities and needs.
- Comprehensive documentation of the business.
- A method of quantitative evaluation of the proposed new systems and enhancements.

After gaining the required commitment, the details of the methodology to be utilized was developed.

My background has included eighteen years of Data Processing experience. I have worked primarily in the area of system development, ardently advocating the techniques of Structured Analysis and Design since 1974. It was clear to me that the techniques of Structured Analysis were not in themselves sufficient to produce the desired results on a scale as large as a whole business' operation. In the areas of Long Range planning, I was familiar with the Business Systems Planning methodology developed by IBM. Although this methodology was an excellent tool for planning, it did not encompass the tools to do some of the analytical steps needed to produce the action plan. Intuitively, I also recognized that the greatest underlying problem to be solved was in getting the data of this business organized in a logically structured manner. I therefore developed a methodology which combined the strengths of the techniques of Structured Systems Analysis with those of Business Systems Planning.

Getting Started

With the methodology chosen, the Study team selected and the System Plan approved, it was now time to orient the team in the methodology. Only one member of the team, beside myself, had experience with Structured Analysis; none of the other members had exposure to the Business System Planning methodologies.

The Manager of System Development, who had Structured Analysis experience, was selected as the Team Leader. We spent a significant amount of time reviewing texts, and selected two as the basis of our educational effort. They were:

- Structured Systems Analysis: Tools and Techniques by Gane and Sarson -- McAuto
- Information Systems Planning Guide IBM Corporation

We developed the project plan and the task descriptions. It was understood that the first step was to define the processes that made up the business. We further realized that these processes should be independent of the organizational structure which existed within RCM. I had been at RCM less than two months, and he had less tenure than I. The use of processes to define a business is, at best, an abstract concept. It therefore was essential that before we began any working sessions with the Study Team we produce a preliminary list of processes and their definitions, knowing full well that the business user members would find deficiencies. This path was chosen, because there was no other means of communicating this abstract concept, and it would serve as a basis for discussion and involvement on the part of the business end-users in a process that was at first foreign to them.

The initial education process involved meeting with small groups of Study Team participants, first broken up along organizational (division) lines so that they would feel more comfortable. These initial meetings provided an introduction to the objectives of the project and the first introduction to the techniques of this methodology. The components of both methodologies chosen were really techniques which have historically been the purview of Data Processing professionals. I have long felt that the primary tasks in the process of Business Systems Analysis could better be carried out by those who intimately understand the nature and nuances of the business, whose skills can be enhanced with the discipline of the rigors of System Analysis. Data processing professionals generally must learn these aspects of the business, the subtleties of which may take many months or even years to master, before their analytical skills are at a level to provide significant value added to the process. These sessions introduced participants to the terminology and symbols of Structured Analysis Data Flow Diagrams by utilizing a subject which all team members could identify. Several weeks beforehand, MIS was faced with a problem in which a process which had been unique to a single division of the firm, was now being utilized by another division of the firm. When we were asked to change programs to accommodate the new use, we made most of the required changes. After the changes were implemented and the process utilized by the new division, problems began to develop. These problems were caused by an incomplete understanding of the process and of the interfaces required to the new division. We proceeded to chart in DFDs, the current use of the process and then create a DFD of the new use of this process by the other division. We created a third DFD to illustrate what a visual presentation of the process would be like, if there were only one process and it had the facilities to service both divisions. Armed with these DFDs, we began our education process. After discussing the meaning of the abstract symbols of DFD and their usefulness in graphically representing a complex set of relationships, we were faced with some rather blank stares. We then introduced the DFDs we prepared to illustrate the abstract concept in a practical manner, and we suddenly had a room filled with highly interested users. Our introduction was a success!

Our next step was to reorganize the Study Team participants in groups which were not tied to functional organizations, but rather would represent a complementary mixture of the various divisions of the business. Each group was set up to include one member from MIS, and three or four Business end users. Each group would be assigned certain processes to analyze. There was usually at least one team member with the expert knowledge for each process. The MIS member would serve the purpose of educating the other team members in the methodologies in use, and at the same time be educated in the functions of the business. This was clearly an objective of the process -- to create a MIS staff with an understanding of the business for which they would be called upon to develop new Systems. An additional objective of the process was to educate Business End Users from one division of the firm in the business functions of the other divisions, thus creating a foundation for the introduction of functional process view of the business, as opposed to the current organizational view of the business. One overriding concept which this process instilled was that the processes which make up the business would not change even if the organizations responsible for a process were to change its structure or members. Each team had one member who would function as its leader. The team leader was responsible for assuring that the assignments of that team were completed on schedule and for assuring the quality of the output of that teams products. All team leaders were Business End Users.

Defining and Documenting Business Processes

We presented in excess of 50 processes to the Study Team members. Individuals with expert knowledge of various processes objected to our terminology or our descriptions of the processes. This dialog was a particularly helpful one. Through it, we were able to encourage active participation of the team members and correct and refine the processes which we had originally posed.

With the processes initially defined, we divided the processes equally among the teams. Each team would now be responsible for preparing a high-level DFD for each process assigned to them. They were told that they should start with whatever expertise they had available within their team. If that were not sufficient, they should go outside the boundary of their team, and utilize the expertise of other resources within RCM. This process was a lengthy one. The first DFDs were not done in a complete manner. The results were reviewed with each team and they tried again. Some problems were directly related to the understanding of the methodology, others were related to lack of precision in the identification of data, etc. The review process was conducted in a way that would reinforce the learning experience of all involved. Early on, we were faced with an inevitable problem -- continuing revision to content of the DFDs required the time-consuming task of redrawing them. To reduce the impact of this problem, we developed a rather simple mechanical enhancement to the process. Each team member was given a standard size cardboard easel to which was attached a piece of clear cellophane. They were also given a number of paper cut-outs corresponding to several of each symbol used in DFDs. These symbols were placed on the cardboard easel under the cellophane. They utilized easily erasable pens to write on the cellophane. The diagrams were then drawn on the cellophane, and there was no longer resistance to change resulting from the review process. The finalized DFD was then photocopied and sent to the Technical Documentation Specialist, who prepared them in final form.

When all processes had been prepared in this first round of documentation, we prepared a large Process Flow Diagram which showed the interrelationship between the various processes and documented significant flows of information between processes. We entitled this the RCM Business Model. This document, together with brief descriptions of the processes was reviewed by the Senior partners. Their reaction was positive. After internal review of the output of this phase of the System Plan study, it became clear that the DFDs were not sufficient in and of themselves to communicate the steps and data involved in each process. It further stimulated us to recognize the specific direction towards which the study was leading us -- that the first job to be tackled was that of identifying logical data groups. We further recognized that it would not be worthwhile to proceed with further detail levels of DFDs until managing partners outside the Study Team could verify and validate the work completed to date. As a result, a decision was made to have team members prepare a detailed narrative describing each DFD which would then be reviewed by the team member with the greatest expertise in the particular process. Our ultimate objective within this phase of the study was to have the narratives and DFDs reviewed by the non-team member partner(s) under whose jurisdiction the process fell and to receive comments on their accuracy.

Understanding the Data

With the process descriptions underway, we embarked in a direction to enhance the understanding of the data. Our first job in this task was to gain a thorough understanding of the data utilized by the existing 1750 programs and 5800 files. This task had been going on in parallel, led by the System Manager. This task was made even more challenging by the fact that our library of programs included many originally written in RPG and many which used KSAM files rather than IMAGE. We created a database of the data elements and the programs using them with a product called DataEase on IBM PCs. From this effort, in conjunction with the output of the DFDs and the business model, we went through a process which resulted in the identification of a variety of data classes.

The next step was to validate the data classes. For this task, we utilized the previously completed business process DFDs and narratives. This process reviewed the identifiable data elements and assured that they were members of the data class appropriately designated for use by the process. In this process a definition of the relationship between classes (eg. one to one, one to many, or many to one) was developed and validated. This set of relationships was then depicted graphically and called a "Logical Data Model".

The Review Process

The review process entailed reviews of all our output to date by Senior Executives who were not participants in the Study. Each executive was given a package containing:

- o RCM Business Model
- o RCM Business Process Definitions
- o Detailed DFDs and Narratives of each process the Executive had primary knowledge and/or interest
- o RCM Data Model
- o Data Class Definitions
- o Review Guidelines

The guidelines for review suggested that each individual answer for themselves the following questions:

- o Is this an accurate description of the business process as you understand it?
- o Is the data used to do the job complete and accurate?
- o Are the steps depicted completely and in the correct sequence?
- o What are the critical success factors in the business process?
- o What are the greatest problems that exist in the business process?
- o What has prevented you from solving these problems?
- o What value (in work-hours saved, dollars saved, or better quality product produced) would better information have?

Each executive was interviewed by myself and the Team Leader for ninety (90) minutes during which these questions were addressed for each business process being reviewed by that individual. The results were positive, and all interviewees were particularly enthusiastic. The reason for this enthusiasm, I believe, was the fact that they were dealing with a concrete depiction of the business processes which they perform or for which they have responsibility. We were able to take a subject which has been in the realm of professional Data Processors, namely Systems Analysis, and present the findings in a way in which business end-users could easily grasp. We presented things in their language, with minimum references to abstract concepts such as data and system. The result of this effort was to achieve a significantly more open communication path between members of the MIS Division and other divisions of the firm. MIS was beginning to be perceived as an integral component of the business.

Reporting the Results

After the external review process described above had been completed, the internal MIS effort of producing the Study Plan Report began. The expected contents of this report had previously been defined to include:

- o Executive Summary
- o An Introduction -- a brief discussion of the study's purpose, method, scope and participants.
- o Business Perspective -- a description of the mission, objectives, services and environment of the business.
- o Findings and Conclusions -- a summary of all the components which had been reviewed by senior executives (i.e. data

classes, business processes, analyses of data class vis-a-vis process and organization, organization vis-a-vis business process--including current systems support).

- Recommendations
- Action Plan

It is worthwhile to explore the section entitled "Recommendations" in depth. As we gleaned more and more information in the study, it was becoming increasingly apparent that the structure of the data was an overwhelming deficiency. Systems had been developed over the past decade to address the solution of one problem or another. During the course of this past decade, a new division was added to to the firm. Although it was distinctly different, it had many elements in common with some of the existing divisions. The systems developed to support this division were independent, causing some duplication of function in comparison with other functions. Additionally, most individuals interviewed identified the inability to access information known to be stored within the HP as a major source of frustration. All of these factors taken together led us to develop two key components that would provide the basis for our recommendations. These two components were a) the Information Systems Architecture and b) the Architecture Priority System.

The Information Systems Architecture is a means by which each business process is grouped into a System Group. It was essential that there be few System Groups, and that each be clearly differentiated from the others in regard to function. We developed five System Groups:

- Portfolio Management Systems
- Research/Analytical Management Systems
- Client Information Systems
- Administrative Systems
- Business Management Systems

Business Management Systems was unique in that it did not contain any business process, but rather was seen as a Decision Support System which had access to information required to make informed business decisions. It needed access to the firm-wide RCM data from some or all of the business processes.

Architecture Priorities was the most difficult factor to finalize. All business processes were candidates for new system development. Given limited staffing resources, computer resources and time, it would not be possible to address all the system needs with equal priority. Our recommendations would be implemented in a 3 - 5 year period. Our prioritization recommendation decision therefore had to be the one with the highest payback in cost reduction and/or increased productivity. For these reasons, we recommended that Business Management Systems be the first area addressed. This required that RCM's data be totally reorganized with relationships identified and redundancies eliminated. All data now would be resident on an IMAGE database. All the I/O processing for each existing program would have to be changed to access the new database. We believe that the database is the essential ingredient in a viable system redevelopment process. Even though the basic processing of the existing systems would not be significantly altered (in regard to logic) their maintainability would be significantly enhanced. In addition, all components of the RCM organization would be able to access data currently not easily or readily available to them. As we had already defined how the individual systems would relate to each other and how the sub-systems interfaced, we had a high level of confidence that the remaining architectural priorities could be equally easily accommodated, leaving the subsequent choices up to the users.

Action Plan Implementation

The Action Plan tried to realistically address the needs of RCM. First, all current needs could not be ignored. Secondly, we would have to begin work on the second priority while implementing the Database redesign. Our Action Plan therefore allocated resources to each of these areas. We are currently in the midst of our first year of the system plan implementation. Our proposals were well understood and accepted throughout the organization. Most importantly we have a road-map of where we will be going, and a set of common expectations and commitments between MIS and our business end-user community.

Biography

Stephen Belford
is the MIS Director of Rosenberg Capital Management, an Investment Management firm located in San Francisco, CA. RCM has 100 employees and manages Investment Portfolios for many large corporations. Stephen's experience has included more than 17 years in the Data Processing profession, with more than two-thirds in the financial and securities industry. His expertise is in the area of analysis and system development.

A THESAURUS OF INFORMATION - DOCUMENT STORAGE AND RETRIEVAL

Franz-Josef Boll, Consultant

Kreuzstr. 25, 6052 Mühlheim am Main, West Germany
Tel.: 01608 - 71580

Summary

Document storage and retrieval might be a widespread application of computer-technology in the next years.
There is a large potential of applications, for dedicated information systems on a specific subject and in connection with word-processing systems.
Information-retrieval systems as a major section of non-numeric data processing require special technologies. We will discuss two aspects briefly : data base technology and keyword (descriptor) definition and selection.

Three existing systems will be presented shortly :

for mainframes GOLEM (by SIEMENS)
STAIRS (by IBM)

for the hp3000 THESAURUS/3000

The article contains product information on the last-named program.

Physics Abs hits two million

Physics Abstracts have published their two millionth abstract. It took 65 years to reach the first half million, a further nine years to make the first million. The second million took only 10 years indicating the accelerating growth of physics literature.

The Problem lies in a continuously increasing amount of information available, an information "avalanche", that comes in faster and more extensively every day.
And the need not only to handle this information, but to handle it quickly, to be up-to-date.

Another aspect is the increasing usage of word-processors, the fact that more and more documents are stored in computer-readable form from the beginning.

The answer is a new type of information systems, so called "Retrieval Systems". Programs that handle a special type of data, unstructured data, usually texts, documents without a fixed format. These programs deliver a special type of information, usually, they answer questions like : which documents are available on a specific subject, which subjects are stored at all and so on.

Retrieval systems might be dedicated systems, designed e.g. to store all kind of information on a subject, for example environment protection.
Or they might be used together with word-processing systems allowing you to keep track of your correspondence.

IR-Relations

Retrieval systems handle a special type of data - texts or unstructured data in general and require special data base techniques.

Retrieval systems can be understood as a special type of relational data bases, where the relation between a large number of texts and a large number of descriptors has to be stored. It may be defined as a two-dimensional **array** storing the values TRUE or FALSE.

Descriptor \ Text	1001	1002	1003	1004	1005
VARIABLE	F	F	F	F	F
VEKTOR	T	F	F	F	F
VERBINDUNG	F	T	F	F	F
VERKETTUNG	F	F	F	F	F
VOLLSTÄNDIGKEIT	T	F	F	F	F
VOLLTEXT	F	F	F	F	F
VOLLTEXTRECHERCHE	F	F	F	F	T
VSAM	F	F	F	F	F

Excerpt from a sample IR-data base

The direct implemenation of this data structure would yield a very large bit map, where by the way, most bits are set to FALSE, the table is nearly empty.

Other technologies in use are so called "inverted lists".

An index-sequential access to the descriptor-list is required.

Obviously is the efficient and straightforward implementation of this structure one of the musts for a good retrieval system.

On-line access should be possible not only for information purposes, but also for the update of the data base. Response-time is, of course, one of the main criteria in selection a retrieval-program, and many retrieval-systems in use have the fame to be "resource-hogs", to eat up computer-resources like Bert (in Sesame street) eats up cookies.

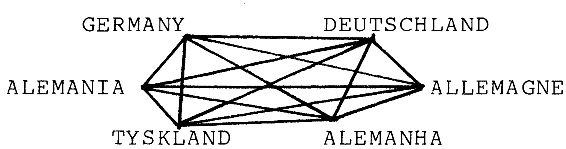
Descriptor Selection

The quality of the information in your IR-data base is of course decisive for the qualitiy of the information. Therefore another crucial point of a retrieval system is the selection and definition of keywords, descriptors.

An optimal keyword-thesaurus would be :

* complete - no information that is required would be omitted
* irredundant - no obsolete information is stored
* definite - descriptors are always stored in the same terminology

A number of additional problems have to be solved :
The descriptor list should be structured, not "flat", for example the following synonym relation may be relevant in a multilingual environment :



Some languages make frequent usage of word-compositions, the "German" word for computer might for example, occur within the following word-forms:

Computer
Computertechnologie
Dialogcomputer
Computers

Not always exists a physical relation between text and descriptor, for example a text about automatic language translation is also a text about "Artificial Intelligence", even if the word does not occur within the document.

Another problem : Some words might be keywords only if followed by another keyword : NEW YORK
ARTIFICIAL INTELLIGENCE

Other descriptors might be only relevant in a certain context, words like TIME, THING, WORD might be relevant in certain circumstances, but usually they have very little significance.

There is no straightforward, simple solution for this problem, because on one hand you want to achieve no or little operator intervention for the definition process, on the other hand some kind of intelligence is required to achieve a thesaurus of good quality.

This is a subject of investigation, which points into the direction of artificial intelligence. A number of approaches are known, none of them is perfect, of course. The most advanced solution, at least for German, offers probably the PASSAT program for automatic keyword selection, that is offered as a part of the GOLEM System.

What is available also is a number of "thesauri", keyword data bases for specific subjects, e.g. medicine. These "thesauri" usually contain typical descriptors for the subject and a set of structural information onto the descriptors.

The Retrieval System GOLEM

GOLEM (**G**roßspeicher**o**rientierte, **l**istenorganisierte **E**rmittlungs**m**ethode) is a product of SIEMENS in Germany.

Development started in the sixties already, today GOLEM is a very powerful information retrieval system.

It is driven by a command language, that is very powerful, but not trivial to understand.

Some of the features are :
- Retrieval by keyword (operators are AND, OR, AND NOT)
- generic keywords and ranges are available
- a structured thesaurus can be defined and used for retrieval, e.g. synonyms, generic terms
- keywords can be combined with structured information, e.g. numeric values
- a set of retrieved documents can be scanned sequentially for the occurrence of strings

GOLEM uses a special-purpose data base, where B-trees and compressed bit-maps are among the technologies in use. Concurrent retrieval and thesaurus update is not possible, GOLEM is in that respect still batch-oriented.

GOLEM is offered together with a system called PASSAT that serves for a highly automated thesaurus maintenance. PASSAT is based on linguistic research, it is designed to analyse and process texts, to remove endings, separate meaningful from meaningless words, recognize words within composits, even some semantic analysis can be done. PASSAT is available for German, English, Italian and French.

GOLEM Data Base Layout - from SIEMENS Schriftenreihe data praxis

(1) Organisation file containing structural information about data base structure, users, access etc.

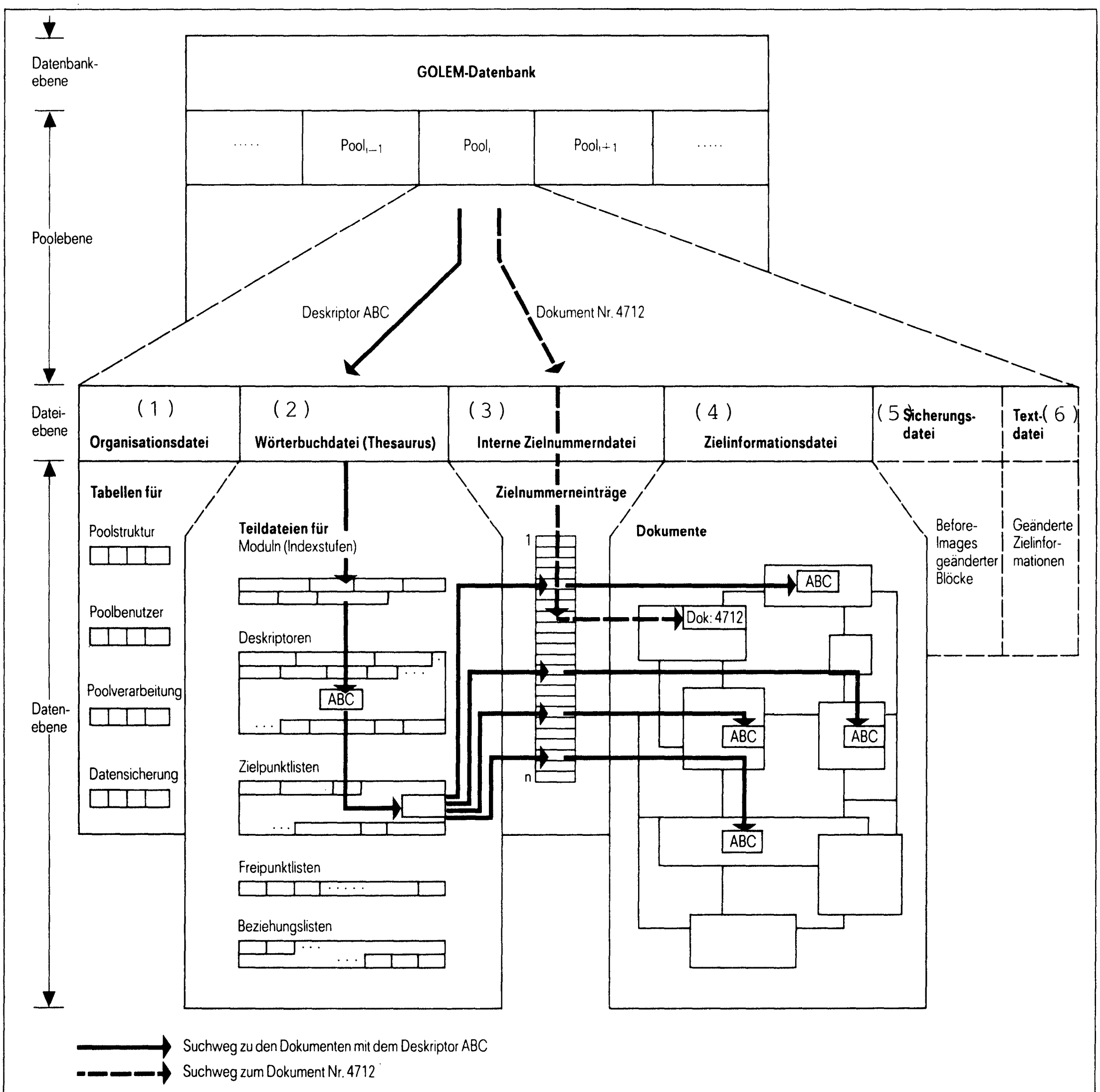
(2) Thesaurus containig descriptors, bit maps (IR-relations), internal information , relations between descriptors

(3) Internal Document Number File stores numbers and pointers for each document

(4) Document Data Base

(5) Log File

(6) Text File



STAIRS/CMS

STAIRS (**St**orage **A**nd **I**nformation **R**etrieval **S**ystem) is a product of IBM, in a way it has become an industry standard.

STAIRS is a very large, very expensive system, it is said that it requires a tremendous amount of computer resources.

STAIRS uses a more mathematical, "relational" approach in comparison to the linguistic features of GOLEM.

STAIRS uses all words within a text (apart from so called "stop words") as descriptors, a simple solution that will, however, store very many meaningless descriptors (the thesaurus data base becomes very large). This approach is not able to handle some of the word-structures in German, Swedish and so on, composit words are a big problem, word flexions also.

Some of the features :

- Retrieval by keyword (operators are AND, OR, AND NOT XOR, WITH, ADJ)
- generic keywords and ranges are available
- macros may be defined (they are called synonyms, but I think that is not quite correct)
- Online help function
- structured information serves as additional criteria

STAIRS uses standard DMBS like VSAM, online update of the thesaurus is not possible.

STAIRS uses a complicate command language that is probably more useful for an expert than for non-dp personnel.

A major problem is the absence of a structured thesaurus, i.e. no true synonyms, no secondary keywords may be defined.

THESAURUS/3000

THES/3000 is a retrieval system available on the hp 3000. It was developed by myself with the objective to offer an easy-to-use and relatively inexpensive general purpose retrieval system.

The main features are :

- runs on hp3000 computers (block mode terminals)
- retrieval by keyword (operators are AND, OR, AND NOT)
- generic keywords
- synonyms
- retrieval by strings and by secondary criteria (sequential search)
- menu driven, currently four local languages, softkeys
- 8 languages can be processed
- interface to MPE-files and to HPWORD-documents
- semi-automatic (but user-controlled) keyword selection

The system uses an IMAGE data base with emulated index-sequential access and inverted lists. Concurrent thesaurus update and retrieval is possible.

Document	Section
WIRTH NIKLAUS WIRTH ALGORITHMS AND DATA STRUCTURES ELECTRONIC DIGITAL COMPUTERS PROGRAMMING DATA STRUCTURES COMPUTER SCIENCE PASCAL ALGORITHMS	1 DATA TYPES SET ARRAY FILE RECORD
	2 SORTING INSERTION SORT STRAIGHT INSERTION TREE SORT STRAIGHT SELECTION PARTITION SORT STRAIGHT EXCHANGE POLYPHASE SORT MERGING
	3 RECURSIVE ALGORITHMS RECURSION BACKTRACKING EIGHT QUEENS PROBLEM STABLE MARRIAGE PROBLEM OPTIMAL SELECTION PROBLEM

Scope of descriptors
as it may be defined with THESAURUS/3000

References

(1) Biller,H.; GOLEM BS 2000, Ein System zum Speichern und Wiedergewinnen von Informationen; SIEMENS-Schriftenreihe data praxis/München 1979

(2) Boll,F.-J.,Geffken,J.; Electronic File Cabinet - Online Information Systems for unstructured Data; Anaheim Proceedings 1984

(3) Hoffmann, D.Mössler,A.; Automatische Selektion von Stichwörtern aus Texten; SIEMENS-Schriftenreihe data praxis/ München 1979

(4) IBM Deutschland;Storage and Information Retrieval System - Benutzerhandbuch; IBM Corporation 1980, 1981

(5) Mresse,M.; Information Retrieval - Eine Einführung; Stuttgart 1984

Biography

Franz-Josef Boll was born in 1951 in Mühlheim am Main near Frankfurt, Germany. After high school, he studied in Frankfurt to become a teacher for mathematics and social science.
After one year working as a teacher, Mr. Boll joined the computer community to work as programmer and system analyst. Working independently since 1980, he has developed a multilingual wordprocessing system for the hp 3000 (called IDT), the retrieval systems IDT-EF and THESAURUS. He also does linguistic research.

Hobbies : music, voyages.

HEWLETT-PACKARD 's INTERNAL DATA PROCESSING;

A WAY TO OFFICE AUTOMATION

Do van Drunen
Hewlett-Packard
Amstelveen
The Netherlands

Contents

1. Summary

2. The tools

3. The strategy; 6 major components

4. A case example; The Netherlands

1. SUMMARY

"Internal data processing by H.P. has been played a key role to success at H.P."...... (John Young, president at H.P.) Experience has shown that the most effective information systems mirror the style and organization of the company in which they are installed. Consequently, H.P. systems come in various flavours depending upon where, how and for what purpose they are used! This hand out explains, in general, each of the six major components of the international I.S. strategy namely:

1. Applications architecture;

to explain the company wide strategy with emphasis on the relationship between corporate and local information systems.

2. Integration standards;

address the development of modular software that can be integrated.

3. Application technology;

to examine along with hardware, operating systems, data communications, and interfaces.

4. Software life cycle;

shows you step by step the development methodology at H.P.

5. User interface guideline;

is a standard for all software applications to ensure that they present a homogenous image to the user.

6. Data standard section;

for the standardization of data related to distributed and networked products.

Finally it explains the role of a country organization like Holland having local responsibilities for tactical decisions whereby H.P.'s world wide telecommunications network provides the foundation for all systems.

2. THE TOOLS

Hewlett-Packard installed its first computer system, for internal data processing purposes in 1960 Palo Alto U.S.A.. From that time on a fundamental change took place almost every 5 years.

* 1960	stand alone main frame in one location.
* 1965	batch oriented computer centers in different locations.
* 1970	R.J.E. in most important locations first step up data transmission.
* 1975	decentralized systems set up; distributed information processing.
* 1980	network will full distributed data processing
* 1985	personal information processing
* 1990	- - - - - - - - - - - - - - - - - - -

Today to serve 80.000 employees we use:

500 x HP3000
200 x HP1000
20.000 x terminals
10 x mainframes in both the corporate as well as the European H.Q.

they are distributed as follows:

* Worldwide minicomputer network

on HP-1000 computers for mainly data communication and front/end data processing.

* Corporate Headquarter in Palo Alto

consisting of big main frames, HP-3000 systems and HP-1000 systems to provide:

- central processing and bureau processing for the bay area divisions.
- corporate data processing.
- U.S. timeshare facilities.
- central control of minicomputer communications network.

* European headquarter in Geneva

uses main frames, HP-3000 and HP-1000 computers to consolidate European information, batch processing and network control.

* Regional/Country sales offices

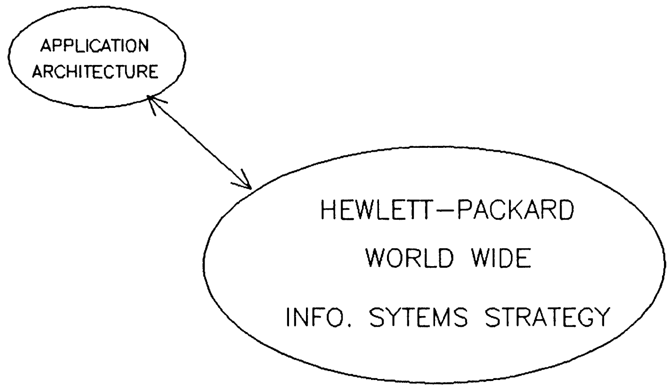
using interactive applications, local batch processing and network communications.

computer hardware concept

	BIG MAIN FRAMES	MAIN FRAMES	BIG HP-3000/ HP-1000	SMALL HP-3000
Corporate H.Q.	X	X	X	X
European H.Q.		X	X	X
Region H.Q.			X	X
Country H.Q.			X	X
branch offices				X

3. THE STRATEGY in 6 components

APPLICATION ARCHITECTURE



An information architecture resembles a road map. It identifies the major processes used by a business and shows the flow of information between them. These processes are broken down into modules that support functional business activities, such as purchasing, inventory management, shipping and cost accounting. By specifying the flow of information between the modules, the information architecture helps define responisibility for data origination and use. The road map helps managers understand the flow and usage of information within their organization.

HP is strongly committed to an environment that nurtures creativity. We consider it essential to maintain a "one company image" in a few important areas - customer relations, vendor relations, quality, personnel and accounting. Not only does this approach help us provide direction for our company-wide acitivities; it also helps us cope with government, legal and taxation requirements. The information systems needed to support HP's decentralized but unified environment can be classified as global, shared or local.

Global systems help us operate and control activities that management has decided to handle uniformly on a company-wide basis. These activities include sales and service, procurement, quality, personnel, accounting and telecommunications. HP global systems use a distributed network of computers located a over one hundred geographically separate locations tied together with a worldwide telecommunications network. Since these systems mirror HP's unique management philosophy and policies, they tend to be HP specific.

Shared systems resemble global systems since they are installed in more than one location but they differ because they are not tied together with a multi-site network. These systems provide the backbone of support for local activities such as materials and production management and cost accounting. Since many HP divisions need the same system features, shared system can provide substantial savings in design, enhancement and support when they are used by multiple entities which use central support resources.

Local systems are designed to satisfy unique needs or to provide prompt response to changing requirements. They can be stand-alone or can be interfaced to global or shared systems. Many local systems are currently being implemented on HP150 personal computers. Local systems are used in a wide variety of applications including word processing, "What if?" modeling using spreadsheet analysis, forecasting and budgeting.

What Is An Information Architecture?

The next figure shows Hewlett-Packard's information architecture. An information architecture of this type is an excellent way to identify and explain the systems modules that are used to support business processes and the information relationship between them.

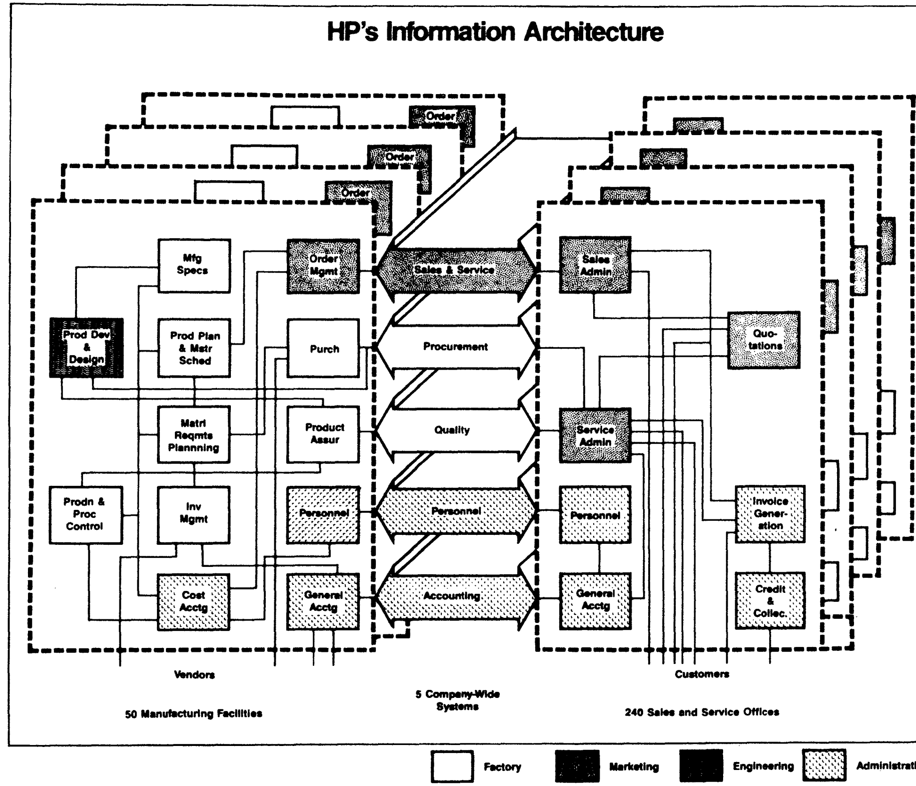
This somewhat oversimplified architecture appears as two decks of cards connected by five double-pointed arrows. Functional procedures used by the manufacturing facilities are on the left, and those used by the sales and service offices are on the right. In concept, each of the 50 manufacturing facilities and each of the 240 sales and service offices is represented by a card. The five global systems that provide the information structure to help HP's managers operate the company as one entity are shown in the middle.

Since each of the global systems interfaces with all manufacturing facilities and with all sales and service offices, they provide a powerfull mechanism to help unify company-wide activities. HP's worldwide telecommunication system provides the foundation for the global systems.
\

The product development module appearing within the manufacturing area is used to support divisional engineering activities. Adminstration functional procedures appear both in manufacturing and sales and service entities since personnel and accounting activities are required throughout HP's entire organization.

The important concept of leverage is illustrated at the upper right corner of the manufacturing "deck." Many of the systems requirements for all manufacturing entities are essential the same. The order management module shown is a shared system used by all entities. This type module offers an opportunity for the HP divisions to improve their profitability by creating leverage in the development, enhancement and support of their systems through sharing.

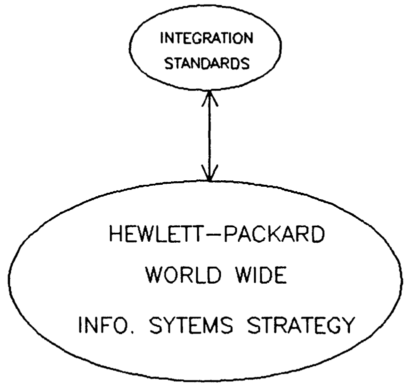
An important aspect of systems integration is multi-functional responsibility for systems. For example, cost accounting systems are designed and operated by the administration function, but the primary users of the information they provide are manufacturing personnel. Manufacturing managers need to take a leadership role with regard to cost accounting systems to ensure that the information provided meets their needs.



HP's Information Architecture
Order
Order
Order
Mfg Specs
Order Mgmt
Prod Dev & Design
Prod Plan & Mstr Sched
Purch
Matrl Reqmts Plannning
Product Assur
Prodn & Proc Control
Inv Mgmt
Personnel
Cost Acctg
General Acctg
Sales & Service
Procurement
Quality
Personnel
Accounting
Sales Admin
Quo-tations
Service Admin
Personnel
Invoice Gener-ation
General Acctg
Credit & Collec.
Vendors
Customers
50 Manufacturing Facilities
5 Company-Wide Systems
240 Sales and Service Offices
Factory
Marketing
Engineering
Administration

Figure

INTEGRATION STANDARDS



The success of the MPN Network depends on our ability to procedure integrated, but modular software. This means that every module or product must be able to run independently or integrated with other modules and customers can grow their systems architecture in a cafeteria fashion depending on their business needs.

Integration can be viewed from three perspectives and integrated modules require all three. They are:

Horizontal Integration - This is the functional integration of business applications. Functionally integrated systems have features like single-point data entry, no duplication of functions between modules, and common logical storage of key data items.

Vertical Integration - This is the integration of business applications with the underlying technical tools and facilities. Vertical integration features efficiency of operation and resiliency to change of things like database management systems, operating systems, screen handling utilities, etc.

User Integration - This is the integration of the way applications appear to the user. Features of this type of integration are common screen formats, common report formats, help facilities, etc.

Product Boundaries

The previous section on Application Architecture describes the work done to date on establishing product boundaries. These architectures are key to horizontal or functional integration as they clearly describe where different business fucntions are to be supported.

These architectures are the starting point, not the finishing point. Product plans need to be validated against these initial architectures. These architectures need to be living architectures which change as business and market needs change. However, significant changes to these architectures need coordination, communication, and management approval across the organizatio

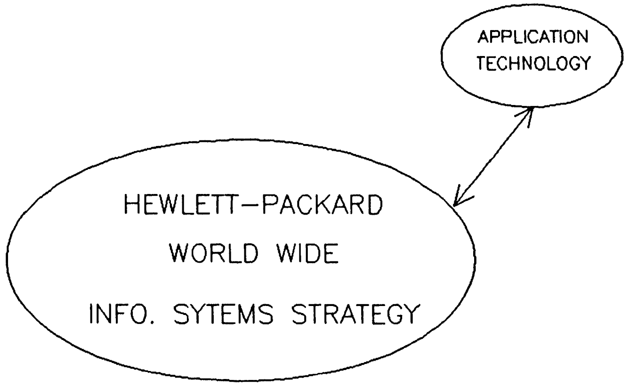
Database/Dictionary/Network Directories

Even with defined product boundaries and data that is both standardized and validated, the modularity is still dependent on the technology of the databases and networks. Vertical integration of our systems hinges on this.

Without superior technology in this area, the chore of interfacing modules will be left to the customer and it will become an increasingly significant task. The result being poor performance and high cost of system operation.

Ultimately, the customer will purchase not only two but many applications. For the entire network there will be many databases. However, if the data and the structure are standardized, we may think of them as a simple logical database.

APPLICATION TECHNOLOGY



Development

Our long term strategy is to achieve maximum integration and standardization among applications being developed using the current technologies with those that will be developed using the next generation of technologies.

1. Integration / Standardization

Initial steps toward integration and standardization should be taken by using existing subsystem for software development. For exemple, V/PLUS should be used for development of interactive applications for forms storage and terminal I/O. IMAGE databases should be used for data storage whenever possible.

2. Databases

IMAGE is recommended and encouraged for develoment of all applications whenever possible. Databases I/O should be modularized, put in generalized subroutines, and isolated from the rest of the application code so that later changes in IMAGE or migration to another database management system will have minimal impact on the application itself.

3. Distributed Systems/Data Communications

Standard DS capabilities should be used.

4. Report Writers

There are currently two major report writers within HP; 1) RAPID (REPORT) from MPD; and, 2) REPORT FACILITY from APO.

A RAPID interface is also available, for applications that allows downloading of data formats, relationships, and structure APPDIC to DICTIONARY/3000.

5. Operating Systems/Processors

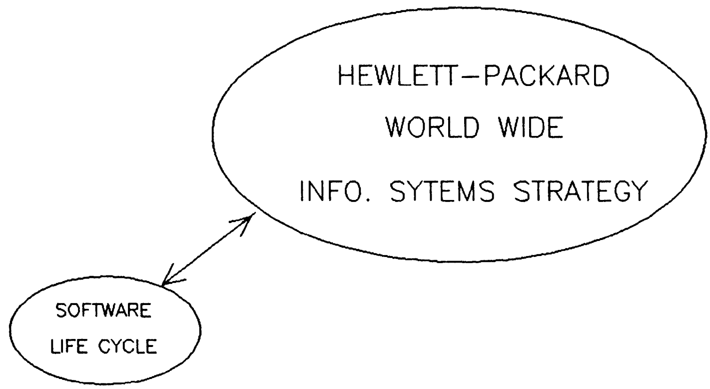
HP3000 using the MPE operating system.

6. Languages

Use COBOL whenever possible. Avoid the use of non-standard features, i.e., those not supported on other HP hardware to which porting may be desired. HPTOOLSET can be a valuable aid for editing compiling, debugging and version control of COBOL coded programs.

TRANSACT should be considered for prototyping. It lends itself to quick development of application models and throw away code.

SOFTWARE LIFE CYCLES AT HP



Many organizations within HP develop and support software. At product divisions, in sales regions and in corporate departments software is developed for internal use, for sale as part of a hardware product or for sale as a stand-alone software product. Software is developped by people with a wide variety of backgrounds, experience, and objectives. As a result, the development and support of software at HP has evolved organically, with little concern for consistency of approach, for commonality of terminology or for evolution toward a standard methodology.

PHASES OF THE LIFE CYCLE FOR SOFTWARE

1. Investigation

The Investigation Phase is used to define the requirements of the software in order that it will be successful when used by its target user. As such, its target user must be defined, the product objectives in terms of functionality, usability, reliability, performance and supportability must be stated and finally, any limitations of the product or any areas of high risk must be defined.

2. Design

The Design Phase has two primary objectives. The first is to specify the product's external interfaces - that is, the user interface and any interface to other systems. The second is to define the internal structure to be used. Since the internal and external design of the product must be planned together and since a decision affecting one often affects the other, the Design Phase is not complete until both are finished.

3. Construction

The Construction Phase is the place of coding and testing the modules of the product and bringing them together for system level testing. During this phase; most of the ancillary work required to produce a complete product is also done; user documentation and training, internals documentation, test package creation, the structure of the product components for manufacturing, and planning for the user support of the product.

4. System/User Testing

After through testing by the development team, the product is ready for test by its target user. This phase is needed to ensure that the product meets its objectives and that it is supportable outside of the development environment. In addition, the integrated system is subjected at this time to intensive testing in the areas of performance and varied configurations to minimize the possibility of unexpected results after wide distribution.

5. Release

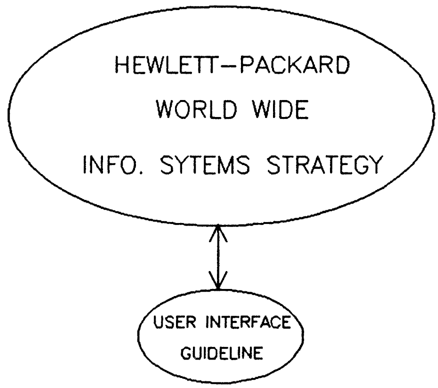
The release phase for software includes transfer of the software and its user documentation to the organization responsible for the delivery and installation of the product. The activities at this phase vary widely depending on the method of software distribution and support used by the entity, but it usually includes version control, final review of quality status, final cosging, pricing and so on, and introduction to the market and/or internal users. The release of a software product is based on the expectations for user satisfaction and the proven supportability of the product.

6. Support

After initial release, most software products undergo revisions for a variety of reasons: enhancements are added, new hardware or software products require new interfaces or support code, bugs are found which need to be included.

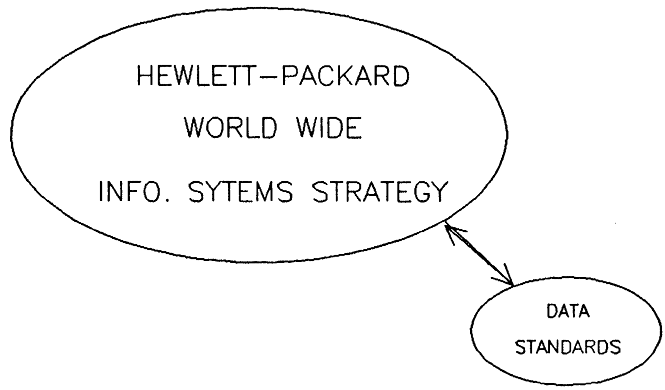
During the support phase, version control, testing, control of releases and supported configurations are major issues. Regression testing becomes essential to ensure that changes to the product have not created problems in code which worked previously. At the end of its supported life, a software product may be obsoleted. If there is an obsolenscene process used, it defines the archival and update of the product as required by the support strategy. Usually, bug fixes or other changes will not be made unless they are of a critical nature.

<u>USER INTERFACE GUIDLINE</u>



Recognizing the need to give a consistent presentation to the computer system user, we have spearheaded an effort resulting in a document titled <u>Screen Standards</u>. This standard will be used on all development projects. The aim is to procedure software products that provide a user interface that is consistent, friendly, easy to use, with uncluttered screens and terse prompts.

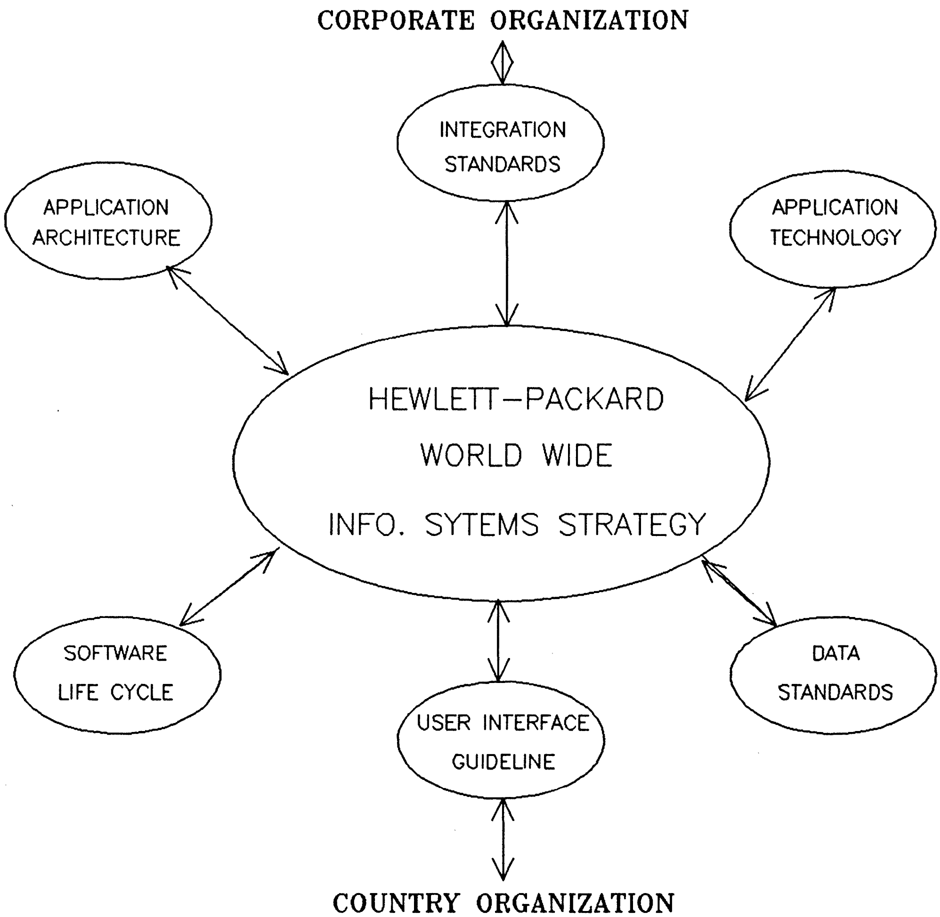
DATA STANDARDS



In order to procedure modular, distributed and networked products, the standardization of data becomes a key requirement. Future dictionary products will allow the gathering and disseminationof standardized data.

The dictionary metadata involves two standards, data definition and data format. Data definition refers to a standard way of describing data, and data format is related to the contents.

Although the prototype will be used mainly by the product producing entities, the information will be rolled forward into the future products. Under this plan, the internal users will get a new tool already containing a basic library of information. It has not been decided whether the tool will be implemented separately or through a systems dictionary. It either case, its functions will include logical data modeling, automated system design and information management. Implementation of the tool will provide a basis for easily sharing applications and contributing to the overall effectiveness of HP's investment in information systems.


CORPORATE ORGANIZATION
INTEGRATION STANDARDS
APPLICATION ARCHITECTURE
APPLICATION TECHNOLOGY
HEWLETT-PACKARD
WORLD WIDE
INFO. SYTEMS STRATEGY
SOFTWARE LIFE CYCLE
USER INTERFACE GUIDELINE
DATA STANDARDS
COUNTRY ORGANIZATION

4. CASE EXAMPLE; HOW DOES THIS CONCEPT WORK IN PRACTICE

H.P. THE NETHERLANDS CHARACTERISTICS

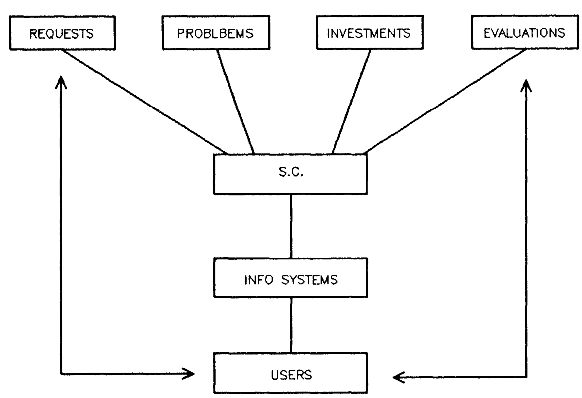
This is a sales organization with the sales headquarter in Amstelveen, 2 branch offices in Capelle a.d. Ijssel and Eindhoven, 1 service antenna in Meppel, 1 traffic location in Diemen and houses about 500 employees.

We are using 3 important aspects to our local strategy:

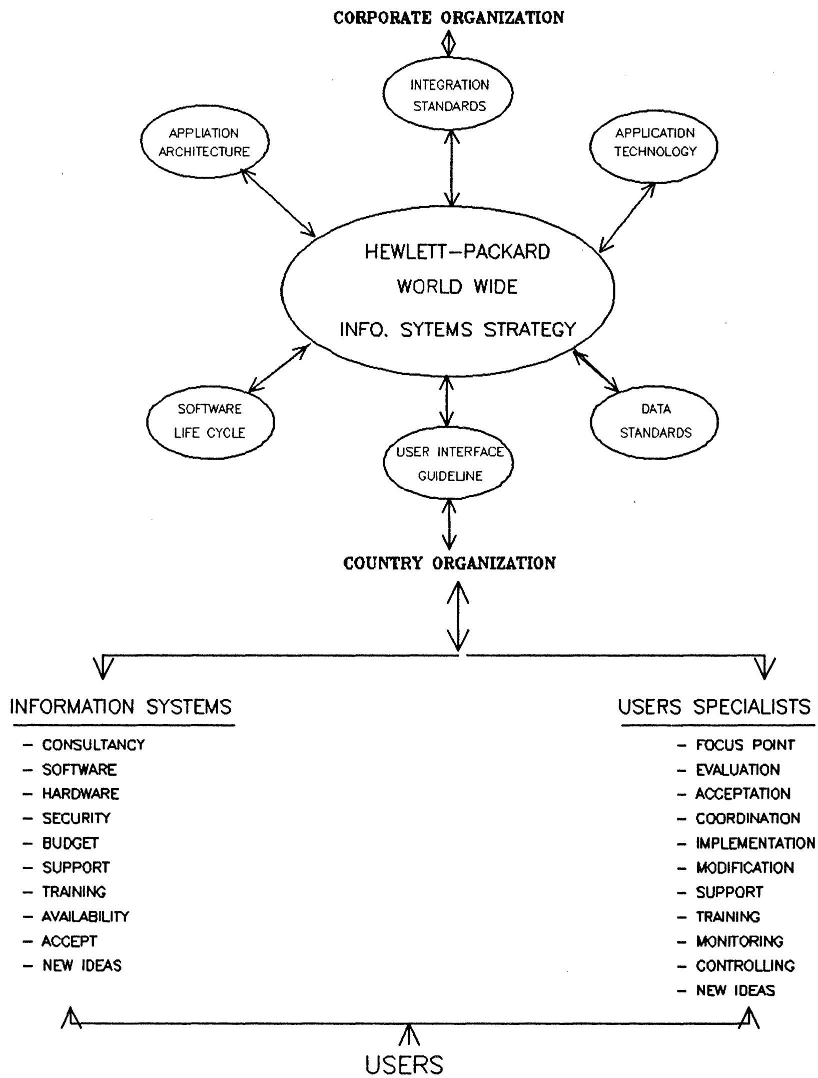
1. Procedure
2. Organization
3. Hardware

1. Procedure

Working within the framework from the international H.Q.'s we have set up an organization steered, by a steering committee which was made up of major department managers who would meet on periodic basis and review and evaluate the different requests related to investment, hardware development evalution and organization aspects.



In order to optimize usage of the application software in a decentralized environment we agreed the next global task definitions.

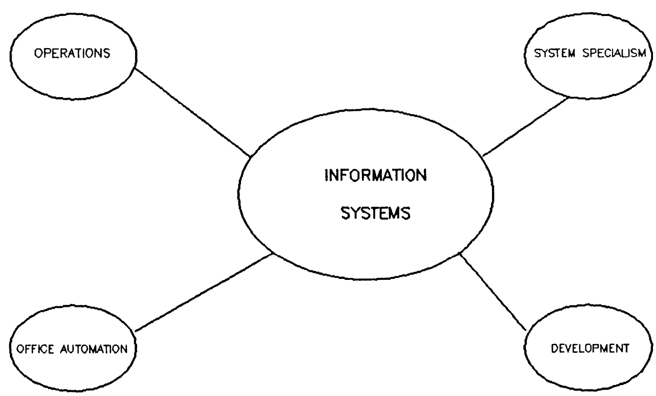

CORPORATE ORGANIZATION
INTEGRATION STANDARDS
APPLIATION ARCHITECTURE
APPLICATION TECHNOLOGY
HEWLETT-PACKARD
WORLD WIDE
INFO. SYTEMS STRATEGY
SOFTWARE LIFE CYCLE
USER INTERFACE GUIDELINE
DATA STANDARDS
COUNTRY ORGANIZATION
INFORMATION SYSTEMS
- CONSULTANCY
- SOFTWARE
- HARDWARE
- SECURITY
- BUDGET
- SUPPORT
- TRAINING
- AVAILABILITY
- ACCEPT
- NEW IDEAS
USERS SPECIALISTS
- FOCUS POINT
- EVALUATION
- ACCEPTATION
- COORDINATION
- IMPLEMENTATION
- MODIFICATION
- SUPPORT
- TRAINING
- MONITORING
- CONTROLLING
- NEW IDEAS
USERS

2. Organization

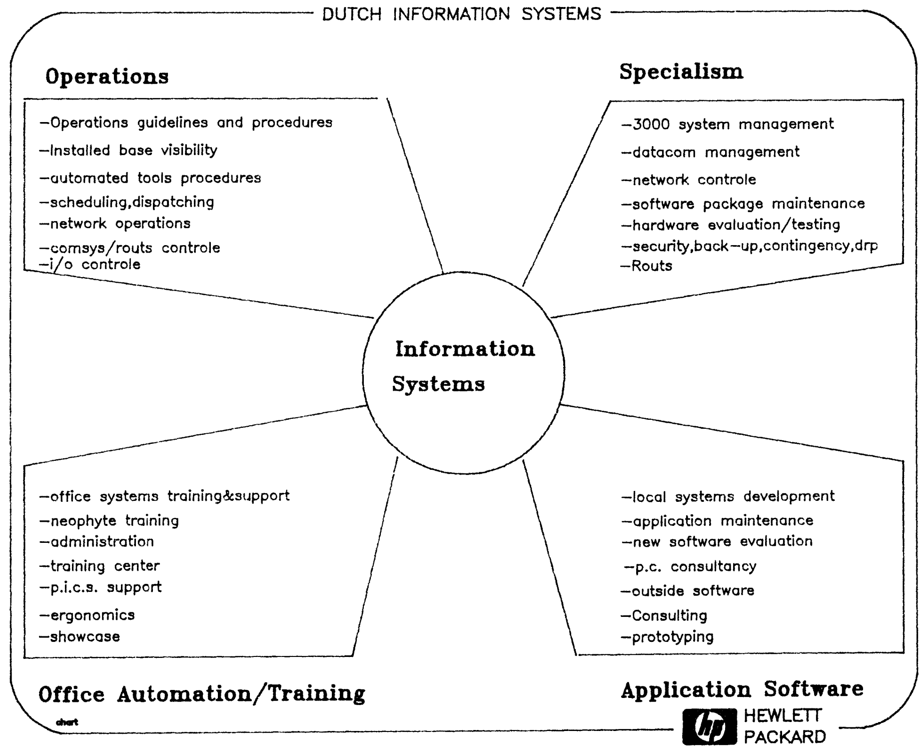
Like any other "classical" d.p. organization the Dutch organization has moved from an earlier position as overseen of a huge technical empire furnishing processed data to the end user. The new I.S. organization is moving toward a organization that is primarily one of consulation and institutional support.

The new style D.P. organization, I would call it "information center" is one with a good interactive relationship between "users" and d.p. staff, that leads to an "office automation environment".

As a result of this new way of I.S. structure careful attention must go into the definition of each area and sub area for which responsibilities must be assigned in varying areas. The different areas are:



Each of these areas must now be broken down into their items of responsibilities.

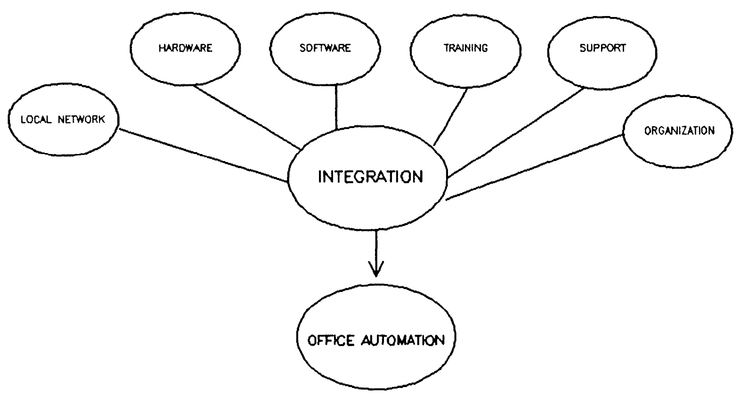

DUTCH INFORMATION SYSTEMS
Operations
-Operations guidelines and procedures
-Installed base visibility
-automated tools procedures
-scheduling,dispatching
-network operations
-comsys/routs controle
-i/o controle
Specialism
-3000 system management
-datacom management
-network controle
-software package maintenance
-hardware evaluation/testing
-security,back-up,contingency,drp
-Routs
Information Systems
-office systems training&support
-neophyte training
-administration
-training center
-p.i.c.s. support
-ergonomics
-showcase
Office Automation/Training
-local systems development
-application maintenance
-new software evaluation
-p.c. consultancy
-outside software
-Consulting
-prototyping
Application Software
HEWLETT PACKARD

HARDWARE

Hardware standardization is the fundament to support our information systems in both local as well as corporate systems. Mainly we are using HP-3000 systems as central interactive/batch processing systems connected with simple terminals. There is a trend towards 150-110 micro computers, to offload work from the HP-3000, for mainly text processing, graphics and spreadsheets.

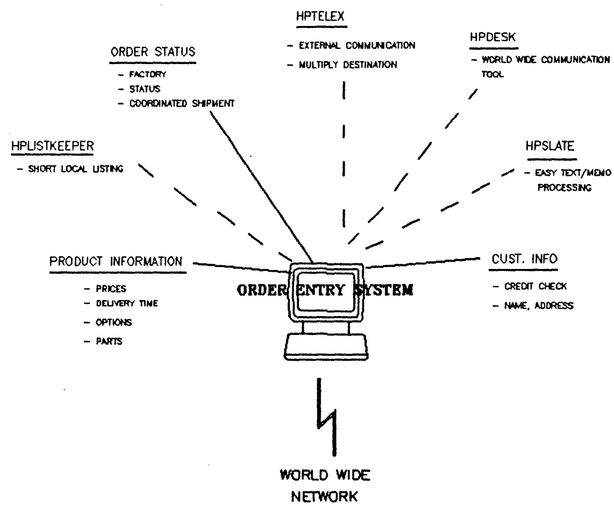
As print facilities we use, in a centralized environment, the HP2680A laserprinter for huge and advanced output, and several line printers. In the office environment we use the 2688A office laserwork stations for advanced quality "output and the 263X and 256X family for "check" printing used by scretarial clerical and professional people.

Finally to optimize usage of computer systems we have installed a local network to give any user access to any application at any computer at any time on all locations.



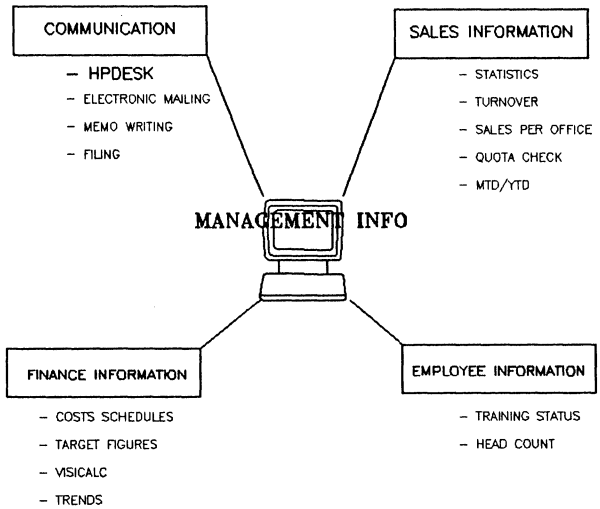
Office automation an example:

An orderprocessing employee is responsible for monitoring orders; providing information to customers about order status, communication with factories, checking orders on completeness, product and customer information All information is available on line; daily updated.



Management information

A manager needs quick, up to date easy to read "information" for decision taking both on line as well as hard copy. This is what we can offer them.



Thank you for your attention.

BIOGRAPHY

My name is Do van Drunen and I am manager of Information Systems for the Dutch sales organization of HEWLETT-PACKARD. My particular mosaic of qualities and experience include a degree in sport teaching and health, 2 years in the army as a marconist, then getting a degree in computer technology. 12 years ago I entered the D.P. world as a operator/ programmer.

My first 5 years in data processing, I spent working in large scale computer environments as systems analyst and project leader in an international structure. Along the way I set up a simple international computer data communication network, which is my speciality today.

Seven years ago, when I joined H.P., I had my first taste of management and working with H.P. equipment in a fast growing environment. This was leading from easy "bee hives" on a HP2100 with simple application towards high degree of automation with complex integrated databases in both a local network as well as a world-wide high, speed data communications network. I am still pleased to be one of the builders!

MR - A SHOP-TIME REPORTING SYSTEM ON THE HP-3000

Ron Frijda

Thomassen & Drijver Verblifa N.V., Deventer, Netherlands.

Summary

The MR (Man-hour Registration)-system is a clock-in/clock-out registration system as well as a houskeeping system which produce all kind shop-time information on a plant and provides the interface with the central payroll and the personal system.

Background

In the plant-environment of Thomassen & Drijver we have more and more flexible working-hours, short-work, partime-work and all kind of shifts with a different time-schema. The handling with time-clock-cards by the personnel department was growing to be a tremendous job. So we decided to implement a EDP-system for the registration of time-stamps records to avoid manual handling. But not only we want to be informed about the presence, we also want as much as possible information concerning the absence and the reason of absence of all personnel.
This paper describes the MR-system where MR is standing for 'Man-hour Registration'. The system design/programming of the system was put into contract with an outside software house. Nevertheless the EDP-department has invested a lot of work to get the system workable, based on the original userrequirement.
We have installed the system two years ago at our plant Leeuwarden on a HP-3000 III and with the two years experience we are now planning to introduce the MR-system in the other plants of Thomassen & Drijver Verblifa.

The MR-system, what is it ....

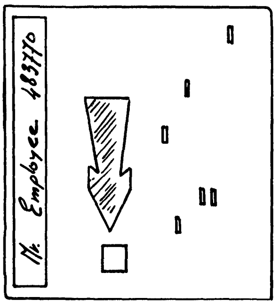
The MR-system can be divided into 5 subsystems

1. the registration of time-stamps
2. the processing of time-stamps to get daily-results
3. the maintenance of all kind of tables and files with information such as personal-record, deviations, shifts and calendars.
4. the information reporting for the production-leaders, the payroll department, the personnel department and so on.
5. the interface with the central payroll-system at Deventer.

In the way we breakdown the MR-system you can see that there is only place for so called shop-time registration. Every employee has to make a time-stamp going in or out the plant. The kind of work the employee does is not on record; shift-leaders and production-leaders are responsible for the job that has to be done.
In the processing of the clock-in/clock-outs you must be informed about who is going to be in or out. Making decisions by having on the one hand the time-stamps and on the other hand the time-schema of presence of all employees is a difficult problem. It is not easy to solve this decision-problem, considering the amount of possible deviations on the expected presence. Deviations are for example illness, holidays and holy days, tour of duty, short absence, day off for one's own account, external courses and studies. The difference between the in- and out-time stamps, attendance time, is compared with a time-schema and stored into a daily-result per employee. This is the center of the MR-system and (of course) we use here a IMAGE/3000 data base as file organisation. From here information is produced for several departmants.
This paper describes the most interresting parts of the MR-system: the registration of time-stamps and the processing leading to the daily-result.

The registration of time-stamps

To accept the clock-in/clock-out's we installed two Time Report terminals (3077A) at the doorway of the plant. For each employee is a badge with his personal identification number punched into it like in the old days the punch-card. The badges are placed in a rack close to the time report terminals. In a normal situation the time report terminals are connected with a Micro 9915A (see fig. 1) which has a delicated protocol-application to entertain the on-line communication with the HP-3000. Time-report records be made up of the items personal-id-nr, date and time and a clock-number (fysical devicenumber of the report terminal). This records are collected in a buffer of the micro and send by the protocol to the receiving program at the HP-3000 putting the records into a wrap-around file. The sole function of the wrap-around file is to collect the time-report records in sequence not interrupted by other processing on this file. A wrap-around file is a formatted file with the caracteristics of a circular-file having the last added recordnumber in record zero and the last read recordnumber in record one.

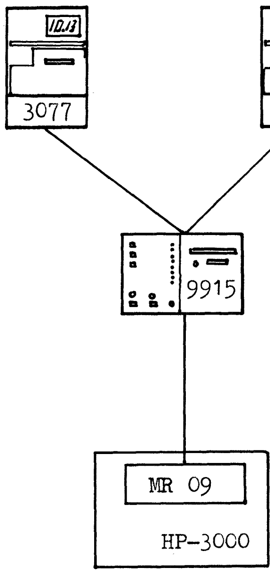


'badge'

There where three problems we have to solve because of the user requirement to record the time-stamps day and night, summer and winter.

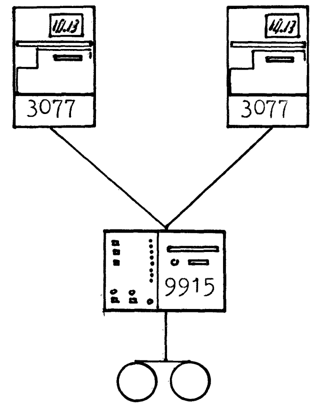
a) what to do when the communication between the micro and the HP-3000 is cut down in case of system-dump or computer breakdown
b) what to do by a break of power-supply
c) what to do with a breakdown of the micro

So we have made a lot of thinking to get the right back-up situation. The micro has a tape cartridge for loading initial the protocol-application program. But on the moment the micro discovers a disconnect with the HP-3000 this tape cartridge is used automatically as a back-up. As long as there is no-connect



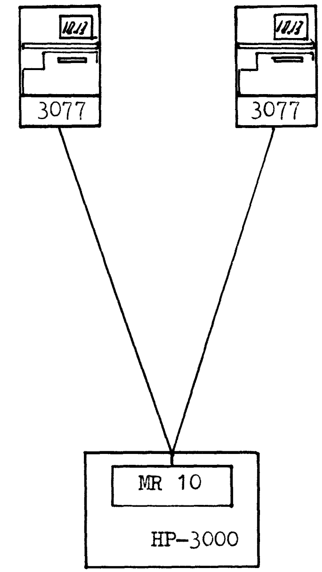
Normal situation

- init time to HP-9915
- MR09 reads
- HP-9915 sends 10 loggings
- MR09 receives, message of right communication
- MR09 writes wrap-around file



HP-3000 / MR09 down

- HP-9915 checks MR09 down
- autom. loads data cartridge
- HP-9915 checks on MR09
- HP-9915 sends data at its convenience
- HP-9915 resumes normal situation



HP-9915 down

- MR10 reads
- MR10 recovers loggings
- MR10 writes wrap-around file

fig. 1

between the micro and the HP-3000 time-report records are stored on the data cartridge. When the HP-3000 is back in the air and there where no clock-in/clock-out's for a 5 minutes the micro send first the logged time-report records to the HP-3000. So we have made the registration of time-stamps independent of the situation at the HP-3000.
Then we placed the two time report terminals and the micro at a power-breakdown installation which gives his own power-supply for 30 minutes in case of power-dip or power-break.
So far, so good, and we never had a fault in the micro for quit a time, but you never know... . The third back-up situation is created by connecting the time report terminals directly with the HP-3000. This is a situation we try to avoid because of the capacity requirement of the time-report record collecting-program, which some times gives provoking respons times on the HP-3000 at rush-hours.

The central MR-data base

To understand the central process of the MR-system there must be first a introduction of the principal sets and tables of the MR-data base. (see fig. 2)

Personal-data

To lay down some personal data we build a set on key personal-id. Data fields are name, shiftcode, sectioncode a.o. The amount of holidays rest for the year and overwork-hours to compensate are to maintain in this entry.

Shift

A shift is a batch of employees working at the same section and with the same work-hours. A chain by key is made to the personal data entries. The shift entry be composed of the items shiftcode, shift description and items to put in the stage of a calendar.

Calendar

The calendar is a combination of a shift and a time-schedule for each day.

CENTRAL DATA BASE principal sets

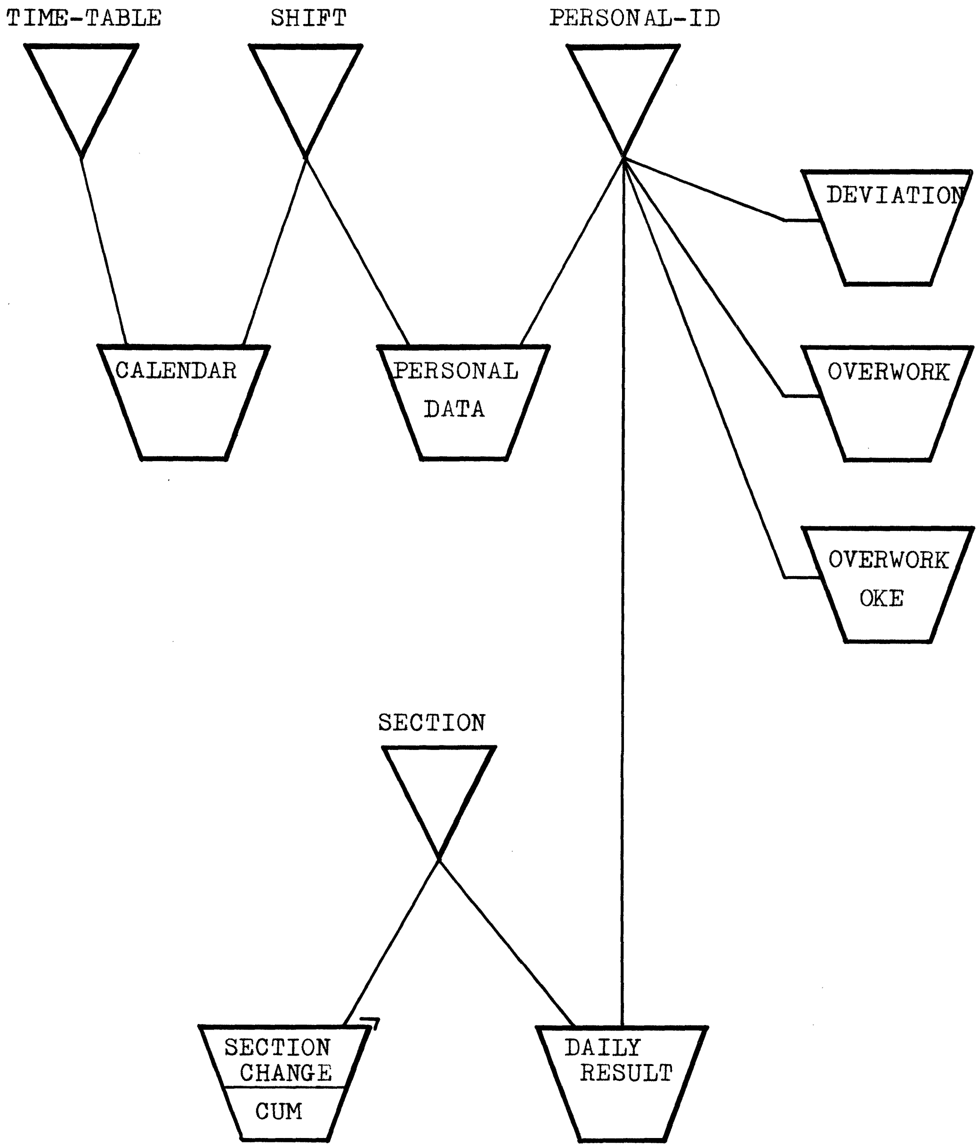


fig. 2

Time-table

Each possible existing schema of work-hours of a shift workman is code into a time-table:

Code TT	:	a four digit code	
start-time	:	fixed time to start (if no block-times)	
start-block	:		
start-rest	:	pause	if flexible work-hours this
end-rest	:		is the minimum presence-time
end-block	:		
end-time	:	fixed time to leave	
computer process-time	:	time that clock-in/clock-out's of an employee according to this schema are processed	
min-clocks	:	minimum amount of time-stamps per employee.	

Daily-result

The information about presence or absence of an employee is stored into this set by key personal-id and key pers.-id-date. There are positions for 6 times a result (hours,minutes) and a resultcode (performance).

Deviation (absence)

Per employee a maximum of four type of absence per day are possible of which two need regristated with start and end-time. The exception is the deviation illness having a start-date and being operative until the end-date is filled in by mutation or by the central-process accepting clock-in/clock-out's from the healthy employee again.

Overwork, overwork oke

If the total presence time exceeds the time according to the time-table it is considered to be overwork and placed by key personal-id-date into a overwork set. Once a week this overwork is approved by someone in authority and rewritten in a overwork-oké set.

Monthly the overwork-oké set is read to generate automatically input to the payroll-system.

The working of the system or ....

Who is in and who is out ....

Within a two hourly interval time report records are read from the wrap-around file, to be processed. With key personal-id and sequence number those records are placed into the (KSAM) Transaction file.
Within a similar time interval of two hours a central program reads the transaction file and makes decisions about presence/ absence of those employees which have equal time-in and time-out recordings. Furthermore only those employees are handled who are supposed to leave according to the time-schedule, even if they had to do overwork. The excesstime is recorded as being overtime. This rather difficult process can be described a little more specific as follows:
For each time-schedule of presence in the time-table the daily expiration-time is put into the item "computer process time". This time is chosen as being 'shift end-time + 4 hours (max allowable overtime)'.
If the central program at its two hours search passes this timemark, it collects from this time-table and the calendar-date all shiftcodes which are ended. From shiftcode to personal-id is only one step and all employees are selected to be processed by the system. The central program with the personal-id reads the Transaction file. It has one the one hand all the 'required working time' at the other hand the actual in- and out-times of the personnel in consideration.
This program calculates and decides on the presence and/or absence and record the result completed with deviations at the 'daily result set' in the data base. (see fig. 3)
Management information concerning presence and absence can be produced at will.

TIME-TABLE

- code TT
- start time
- start block
- start rest
- end rest
- end block
- end time
- computer process time
- min clock's

CALENDAR

- shiftcode
- date
- code TT

PERSONAL-DATA

- pers.-id
- shiftcode
- name
- etc

TRANSACTIONS

- pers.-id
- seq nr
- date
- clock-time

RESULT

- pers.-id
- pers.-id-date
- shift
- clock-time 1
- " 2
- " 3
- " 4
- result
- result-code

(result, result-code: 6 times)

fig. 3

Some MR-reports

The MR-system provides a lot of management information. To get an impression I mention a few reports.

- Listing of daily-results
  This survey gives daily and personal-id wise a review of the daily-result records. The period to be selected can be specified as large as the user wishes.
- Illness-list
  This list gives section-wise a review of the hours-illness, divided in short- and long-time illness.
- Balance list
  This list gives employee-wise the number of not-used vacation days and extra-hours to compensate.
- Presence plan
  This list gives for the coming week, section and shiftwise, the available hours and employees. Possible deviations with regards to the working hours are already processed.
- Gross and net-hours list
  Monthly a calculation of gross and net-hours of presence is made per section. Out of this report the bookkeeping-process takes place and the report is also used for the benefit of production registration.

The payroll-interface

On a monthly bases the MR-system selects all data relevant to the monthly payroll run. This data are transmitted to the head-office computer in Deventer, to be incorporated into the central payroll-system. Before the introduction of the MR-system the personnel-administration had to do a lot of (hand)calculations and data-input, which is taken over by the MR-system.

Biografy

Ron Frijda
is a system-designer and project-leader in the EDP-department at the head-office of Thomassen & Drijver Verblifa, Deventer, Netherlands. For 15 years he started his EDP-career at Philips. Being now for 10 years at Deventer he was responsible for the development of a lot of application-systems and with IMAGE/3000 as his specialism he was the pioneer for the introduction of RAPID/3000. At present his job is the installation of the MR-system in the other plants of Thomassen & Drijver.

AT LAST - RELIEF FOR FRUSTRATED REPETITIVE MANUFACTURERS

Laymond Harrier, CPIM, Price Waterhouse, Cincinnati, Ohio U.S.A.

Ralph Rinesmith, Cincinnati Microwave, Inc., Cincinnati, Ohio U.S.A.

SUMMARY

Cincinnati Microwave, Inc., a repetitive manufacturer, purchased a discrete (MM-3000 and PM-3000) manufacturing software package in August of 1982 and attempted for the next 18 months to implement it. In March of 1984, Price Waterhouse was engaged to assist in completing the implementation. This article describes why the implementation was delayed, how a discrete package can be modified to meet the requirements of a repetitive manufacturer, pitfalls to avoid in implementing a manufacturing system and the significant benefits achieved through successful implementation.

INTRODUCTION

Cincinnati Microwave, Inc. (CMI) is an $80 million publicly held corporation. It is the designer, manufacturer and marketer of the ESCORT and PASSPORT radar warning receivers. Additionally, the corporation is involved in the design and manufacturing of satellite receivers for television reception.

The company, like many other firms, has had a history of computerized manufacturing systems. Initially, a simple inventory management system was tried but the results were less than desirable. Because of this, people worked around the system and developed manual methods to get the job done. These were effective, but not efficient. As the company grew, (over 100% per year) it became apparent that the manual system had to be replaced.

MM-3000 and PM-3000 were selected as the Manufacturing Requirements Planning (MRP II) software. Classes were taken and data was keyed in, but the target date for successful implementation passed, and the system was still not operational. People continued to use the manual system. Why? What went wrong?

No Requirements Definition

Preparation of a requirements definition is one of the most important steps in the implementation of an MRP II system. This was not done at CMI. If initially, they had taken the time to identify the requirements, they would have better understood the current formal and informal system, identified potential problems, educated many of their users to the functions and features of an MRP II system and would have been better prepared to select the software that best met their prioritized needs.

Lack of User Knowledge

Several employees were sent to vendor training classes to learn how to use the system. They didn't learn the concepts behind the development of the system or the interrelationships of the various areas effected by it. Additionally, they did not learn to analyze reports nor the importance of master scheduling. More simply put, they did not understand what to ask of the system or how to utilize its output.

No Real Commitment

Management authorized the purchase of the MRP II package but did not understand that their commitment to its implementation was more important to the success of the implementation than actual purchase of the software. CMI is not the first company to make this mistake nor will they be the last. A company must be committed, in all departments, to make it work. Top management must convey the importance of this project to the entire organization. They must also periodically review the status of the project, obtain education to better understand what the new system will and will not do for them, provide required resources to ensure the successful implementation and insist that specific goals be established and obtained.

Implementation of an MRP II system involves a major effort in many areas and it will not get done in one's spare time. A project leader was not assigned responsibility for the successful implementation of the project. An implementation plan was not developed since it was felt that the system could be installed with each department implementing its portion of the system.

Repetitive vs. Discrete Manufacturing

Perhaps, the biggest problem encountered by CMI was the inability of a discrete off-the-shelf software package to meet the needs of a repetitive manufacturer. Exhibit I illustrates the difference between discrete and repetitive manufacturing. Repetitive manufacturing is the continuous fabrication, machination and assembly of a standardized product units produced (in mass) volume, or of products assembled to order in volume from standard (mass produced) options. The products involved are discrete in nature and different from fluids, powders and processes involving chemical change. They can be handled and called in each.

What really makes repetitive different from classical work order manufacturers is primarily the type of product, the production process requiring specific equipment and their arrangement, and a large volume of units produced. Manufacturing output may be accomplished on lines of work stations set in assembly sequence or at a single bench assembly station.

The remainder of this article will focus on methodology utilized to convert the discrete software to meet the needs of the repetitive manufacturer. Additionally, it will focus on people aspects of an implementation and the benefits that can be achieved through the successful implementation of an MRP II system.

PROJECT METHODOLOGY

Management Commitment

The first step in this project was to obtain a sincere management commitment. This commitment is the key to the successful implementation of any MRP II system. Top management conveyed the importance of this project to their entire organization, provided the required resources to ensure successful implementation and periodically reviewed the status of the project. In addition, they appointed members of their staff to the implementation task force. Each of the following areas were represented on the task force: Data Processing, Engineering, Finance, Manufacturing, Marketing, Inventory Control, Purchasing and Production Scheduling.

Requirement Definition

Preparation of a thorough detailed requirements definition is one of the most important steps in the implementation of a system. Since a requirements definition had not been completed, this was the logical place for the project team to commence the project. A group interview schedule was developed and included responsible staff members from each functional area. At these meetings, the functions, features and data elements required in CMI's MRP II system were identified and documented using the format in Exhibit II for each application identified in Exhibit III. A summarized report was prepared and approved by all users.

The project team utilized the requirements definition in their review of HP-3000 manufacturing software packages. They could not find a suitable repetitive manufacturing package available and, therefore, decided to modify the existing software. It was determined that the following enhancements were required:

- Negative quantity capability with daily monitoring.
- Elimination of the routing file with incorporation of the routing into the B.O.M.
- Elimination of work orders and provide for traceability of work-in-process by part number.
- Improved physical inventory costing capability.
- Incorporation of backflush capability to allow component deductions based upon receipt of parent.
- Improved cost accounting capability that would allow standard cost change variance measurement, scrap costing, material price variance measurement, and monitoring of unplanned issues.
- Capability to monitor parts substitution.

- Dollarized perpetual inventory/general ledger integration.
- Traceability of inventory at outside processing.
- Integration of the accounts payable/receiving/ purchasing systems.

An implementation plan was prepared to accomplish these objectives.

SOFTWARE MODIFICATIONS

Negative Quantities

Changing the MM-3000 software to allow for a negative on-hand balance was the most difficult modification. To accomplish this we utilized the user exit capability to establish an additional field at the location level and at the part level to maintain negative quantities. (Note: Maintaining negative quantities at the part level alone would not provide accurate on-hand inventory balances.)

The example below clearly illustrates this point. If 500 parts were received into location B the negative quantity would remain using the location method. However, if we were tracking only at the part number level, the negative would be removed.

	ON-HAND	NEGATIVE ON-HAND	NET ON-HAND
Part Number Level	100	100	-0-
Location A	-0-	100	(100)
Location B	100		100

To accomplish this, we checked deductions from a location to determine if the transaction would result in the on-hand quantity being less than zero. If it was, we added the negative amount to the current negative on-hand balance. When receiving an item into a location, the negative on-hand balance was brought to zero and the residual added to the on-hand quantity. All screens and reports were modified to reflect negative quantities. Additionally, a new report was developed and run daily that displays all locations for a part with a negative on-hand location.

Routing File Elimination

Bill of materials were restructured as demonstrated in Exhibit IV eliminating the need for a routing file. In addition, this change provided for accurate standard costs at each level and when combined with the backflushing technique, eliminated the need for work orders. Work-in-process traceability was significantly improved since information concerning on-hand quantities were available for each operation as well as raw parts and finished goods.

Backflushing

The backflush technique was designed to post-deduct components and credit production departments for standard hours earned for good production. This was accomplished by incorporating the following logic:

1. Daily production reports are prepared that recap the part numbers and quantities of items produced.
2. Part numbers produced are matched against the bill of material and a one level explosion occurs.
3. Quantities completed are extended times their respective bill of materials. Labor transactions are isolated for usage in the performance reporting system while material usage records are created for all other relationships and input into the perpetual inventory system.
4. The performance reporting system details the amount of hours earned by part number by foreman.
5. The WIP perpetual inventory is updated for the quantities produced and the quantities used.
6. A standard cost change report is generated that compares the frozen standard cost of the parent (item produced) with the standard cost of all items in the one level bill of material explosion.

Work Order Elimination

The elimination of the work order system necessitated the development of a new pick document that would be utilized in pulling and releasing inventory from the stockroom to work-in-process. Weekly, an inventory pull list request is submitted by the stockroom supervisor and each assembly exploded down one level, creating a pick record. A pre-numbered pull list is generated for each assembly with A and B items having daily pulls while C and D have weekly pulls. The items are pulled by the stock room attendant and quantities keyed into the computer via the inventory pull transaction. The inventory will automatically be transferred from the stockroom to work-in-process with this transaction. Items not completely pulled are regenerated reflecting quantity remaining to be pulled.

With elimination of the work order, the lot for lot planning method for MRP was replaced by the day's supply method. A, B, C classification was modified to include 5 variables: lead time, scrap factor, number of vendors, three month extended usage and unit price. This resulted in an improved priority planning and monitoring methodology.

New Transactions

Several new transactions had to be added to the system to facilitate accounting requirements and to meet repetitive manufacturing demands. Exhibit V is a listing of all transactions in the system along with a description of these transactions, and identification of new transactions. Also, it was determined that internal movement of inventory would not be tracked through the general ledger system eliminating the need for detailed journal entries for many of the transactions.

Outside Processing Tracking

The traceability of inventory at outside processing was eliminated by utilizing the same logic as the backflush routine. Parts were issued via the same pull system to the outside vendor warehouse with the location being assigned the vendor number.

Upon receipt of the finished item, a single level explosion was done and all components deducted from the outside processing location. In addition, a standard cost change variance is calculated for these transactions.

Accounts Payable, Purchasing and Receiving Integration

The MM-3000 software was modified to accommodate the Accounts Payable, Purchasing and Receiving integration requirements. Outlined below are the required modifications and enhancements:

1. Add a receiving number to the "Received P.O. Items" screen. This number used in the generation of a receiving log and is the unique identifier for the voucher process. The received P.O. screen also creates an accounts payable accrual record.
2. The accounts payable accrual file is a listing of all unvouchered liabilities. This file is created by utilizing data from the Open Purchase Order system. When an invoice is received from a vendor, the accounts payable clerk inquires into the file to determine if the items being invoiced were in fact received. Once the items are found, the normal verification occurs and the item is released for payment. At the time of release, a voucher document is generated which is attached to the invoice, packing slip and receiving document.
3. Released accounts payable items are entered into the Accounts Payable system when the vouchering process has been completed. No change to the Accounts Payable system is required.
4. Material price variance reports are generated for both the accrued and paid invoices for the period. The reports are presented in sequence by vendor, product line and controller.
5. At month-end, a listing of all unvouchered invoices is generated and used as a basis for making the month-end accounts payable accrual entry.

USER EDUCATION

Two types of user education were provided to CMI's employees. Generic (theory) education was provided to those on the task force and those who must understand the total integration of the repetitive manufacturing system. More importantly, the majority of the employees were provided specific education. In other words, the following questions were answered:

1. How does the system affect me?
2. What new or different things are expected of me?
3. Specifically, how will I use the new system?
4. What role will I play in the total implementation?

We felt it was vitally important to the success of the project that these questions be answered completely and in a timely manner. We tailored the program to CMI's business but did not over-educate their employees. We gave them the opportunity to get hands-on usage of the system before it was implemented. By doing this, they identified additional modifications and enhancements and were willing to make the commitment necessary for a successful implementation.

BENEFITS

CMI has recognized significant benefits from the implementation of their Manufacturing Resource Planning system. A few of these benefits are highlighted below.

- In the period March 31, 1984 to July 20, 1984, work-in-process was reduced by $1,072,437 or 60%.
- ESCORT raw parts inventory was reduced by $1,454,665 or 33% in the period July 6, 1984 to October 20, 1984.
- Overall excess inventory was reduced by 44% during the same period as mentioned above.
- Rework costs have been reduced significantly as a result of the work-in-process reduction program. Smaller queues have enabled more timely identification and elimination of quality problems.

- Perhaps the most important benefit of all is management's ability to request, at any time, up to date information that can aid them in making a decision. This "what if" capability has allowed management, for the first time, to determine the effect of changes in the master schedule before actually implementing a new or revised plan.

CMI management recognizes that the to date results have been very good and that even more significant gains will occur as their staff further refines the new system.

CONCLUSION

Below is a list of pitfalls that contributes to many companies' inability to implement their manufacturing system on the first try. By sharing these with you, perhaps you will avoid them. Don't assume ....

1. That record accuracy is no more important in this implementation than it was in your last.
2. That record accuracy will automatically improve with the new systems implementation.
3. That requirements definition and test plan are not needed since everyone knows what they want. Besides, it will take too long to prepare.
4. That you are so unique that no standard package will meet your needs.
5. That a detailed implementation plan is not needed because you have always gotten everything done in the past without one. Besides, it will take too long to prepare.
6. That top management does not really need to be involved since the system doesn't effect them and they have enough problems already.
7. That external assistance is not needed because in-house staff have read about MRP II, attended a MRP II class, and/or assisted in the implementation of the previous system.
8. That the implementation of a MRP II system can be done totally on a part-time basis.

9. That all software packages do what the salesman claims and that all the enhancements will be included in the package by next week.
10. That your current decision-making process and the way you've done business in the past will not change.
11. That the real measure of success is not whether you have met your goals or whether you are a Class A, B, C or D user based upon someone else's criteria.

We realize that numerous questions remain unanswered concerning repetitive manufacturing and its successful implementation. We could not address or answer all the questions in such a short presentation. However, we hope this presentation has provided you with thoughts, stimulating ideas and at least will provide some relief for fellow frustrated repetitive manufacturers.

Biography

Laymond Harrier is a senior manager with Price Waterhouse in Cincinnati, Ohio. His engagement experiences range from special studies to the design and installation of integrated manufacturing and financial planning controls systems. Prior to joining Price Waterhouse in 1983, Mr. Harrier was responsible for providing internal consulting to a $1 billion diversified multinational corporation. During his tenure with this company, he was responsible for the implementation of twelve cost accounting systems and ten MRP II systems.

Mr. Harrier is a member of the American Production and Inventory Control Society (APICS) and certified in the field of Production and Inventory Control. He received his formal education at Indiana State University where he earned a BS in accounting and at Indiana Central University School of Management MBA program.

Ralph Rinesmith is the Director of MIS for Cincinnati Microwave in Cincinnati, Ohio. He is responsible for all systems and developmental efforts of this $80 million electronics firm. Some of the unique applications software developed at CMI include on-line banking for credit card authorizations and a computerized building access system. Prior to his involvement at CMI, he has held management positions with Martin Marietta and Wang Laboratories. Mr. Rinesmith is on the board of directors for Control Data Corporation's "Fair Break" Center and a member of DPMA.

He received his formal education at Western Illinois University where he earned his Bachelor of Business degree.

EXHIBIT I

CINCINNATI MICROWAVE, INC.

DISCRETE VS. REPETITIVE MANUFACTURING

Discrete (work order)	Repetitive (mass)
o Machine oriented	o Assembly, line oriented
o Work order driven	o Blanket scheduling
o Short runs	o Long continuous runs
o Custom, or non standard products	o Standardized products
o Specific material issues to a job	o Non discrete issues
o Emphasis on detailed input/output controls and capacity planning	o Emphasis on controlling WIP inventory and process flow
o Requires significant reporting from the floor	o Requires minimal reporting from the floor
o Accounting emphasis on detailed accumulated of cost with questionable accuracy	o Accounting emphasis on simplicity and accuracy

EXHIBIT II

SYSTEMS REQUIREMENTS DEFINITION

PRODUCT STRUCTURES

A. The ability to maintain (add, change or delete) the following data elements within the master files.

	Currently Available Yes	Currently Available No	Desired Enhancement
- Product number	X		5
- Component product number	X		5
- Revision level per assembly		X	3
- Quantity per assembly	X		5
- Unit of measure per assembly		X	3
- Scrap factor per assembly	X		3
- Parent operation number		X	5
- Sequence number per assembly		X	5
- Serial number per assembly			0
- Status code per assembly	X		5
- Engineering change notice number		X	5
- Effectivity date control for assembly revision	X		5
- Effectivity quantity control for assembly revision	X		5

EXHIBIT III

CINCINNATI MICROWAVE, INC.

REQUIREMENTS DEFINITION APPLICATIONS

1. Product/Item Description
2. Product Structure
3. Permanent Tooling
4. Disposable Tooling
5. Work Station
6. Routings
7. Vendor Description
8. Customer Description
9. Quality Assurance
10. Master Scheduling
11. Material Requirement Planning
12. Capacity Planning
13. Production Scheduling
14. Sales Analysis
15. Product Costing
16. Vendor/Product Performance
17. Inventory Control
18. Customer Order Processing
19. Vendor Order Processing
20. Engineering Change Control
21. Accounts Receivable
22. Accounts Payable
23. Payroll
24. General Ledger
24. Fixed Assets

EXHIBIT IV

CINCINNATI MICROWAVE, INC.

BILL OF MATERIAL AND LABOR

DISCRETE METHOD

Bill of Material			Routing		
Level	Description	Quantity	Sequence	Operation	Run Time
0	PC Board Assembly	1	05	Stuff	.60
1	Raw Board	1	10	Wave Solder	.05
1	Resistor	2	15	Inspect	.10
1	Capacitor	3	20	Test	.08
1	I.C.	1			

REPETITIVE METHOD

Bill of Material and Labor

Level	Description	Quantity
0	P.C. Board Assembly	1.00
1	Test Labor	.08
1	Inspected Board	1.00
2	Inspect Labor	.10
2	Wave Soldered Board	1.00
3	Wave Solder Labor	.05
3	Stuffed Board	1.00
4	Stuff Labor	.60
4	Raw Board	1.00
4	Resistor	2.00
4	Capacitor	3.00
4	I.C.	1.00

EXHIBIT V

CINCINNATI MICROWAVE, INC.

INVENTORY TRACKING OVERVIEW

Transaction	Description	General Ledger Updated
Adjust Insp.	Same as Count screen except for Inspection area only. Must adjust to current level of + or	Yes
Adjust on Hand	Do not use. (Adjusts quantities at locations - Most adjustments should be performed on the Count screen).	Yes
Back Flush*	Explodes assembly into components and creates WIP Usage 01 and 02.	No
Count	Enter a physical count record of a part at a specific location, adjusting inventory levels up or down, or verifying present level. Cannot use in Inspection area.	Yes
Fill Backorder	Manually fill backorders that have been created for an order.	No
Inventory Pull*	Issue parts on a specific Pull List to WIP.	No
Issue Alloc.	Same as Issue an Order, except for one part at a time.	No
Issue Extra Usage	Issue Extra usage material on a work order.	No
Issue Order	Issue (give out) parts on a specific Work Order from Stockroom to Production or Sub-contractor.	No
Move	Move parts from warehouse to warehouse, within a warehouse, or backorders from Stockroom to WIP.	No
Pull Request*	Pull requests may be issued for the parent part to initiate the allocation of parts in the bill of material and generate the inventory pull list.	No
Receive Insp.	Receive goods or unusable parts from the Incoming Inspection area to the Stockroom.	No
Receive PO Item	The amount received against a Purchase Order line item.	No
Receive PO Part	Receive parts into Inspection or Stockroom from a vendor to a specific P.O. or Debit Memo number.	No
Receive Outside Processing*	Receive a specific set of subassemblies assigned to an order number from production or a sub-contractor.	No
Return	Recovery of parts previously written off.	Yes
Return to Insp.*	Return parts to an Inspection Location.	No

EXHIBIT V

CINCINNATI MICROWAVE, INC.

INVENTORY TRACKING OVERVIEW

Transaction	Description	General Ledger Updated
Return to Vendor*	Issue parts out of inventory to the Return to Vendor account.	No
Sales*	Reflect sales of finished products and accessories.	No
Sales Returns*	Return of a sale item from customer to Customer Service/Shipping.	No
Scrap*	Issue scrap parts out of inventory to the scrap account.	Yes
Sub Part*	Substitute one or more parts for a standard Bill of Material in a WIP Issue.	Yes
Supply Issue*	Issue supplies from the Stockroom to various departments.	Yes
Unplanned Issue	Issue extra usage and engineering parts from the Stockroom.	Yes
WIP Usage 01*	Receipt of Work-in-Process assemblies.	No
WIP Usage 02*	Deduction of component parts from Work-in-Process based upon assemblies produced.	No

* New transaction

INTEGRATING ORDER ENTRY WITH MANUFACTURING
HOW TO HANDLE OPTIONS?

Jim Lawshe
Arizona Industrial Management Systems, Inc.

Summary

The "planning bill of material" technique can be a useful tool in defining a complex product with many options in a manufacturing Materials Management System. Planning bills simplify the bill of materials; identify a product which is easier to master schedule and forecast; and provide a method to handle both long term forecasting and short term job scheduling, but they do have some drawbacks. When implemented in a stand alone Materials Management System, it is cumbersome to edit each order for the correct option mix and it is an effort to keep editing the master schedule to reflect changes in ship dates or to add new requirements. By integrating some of the functions into an Order Entry System and by providing an extended interface capability, many of these drawbacks disappear and the MRP system will function much smoother.

Paper

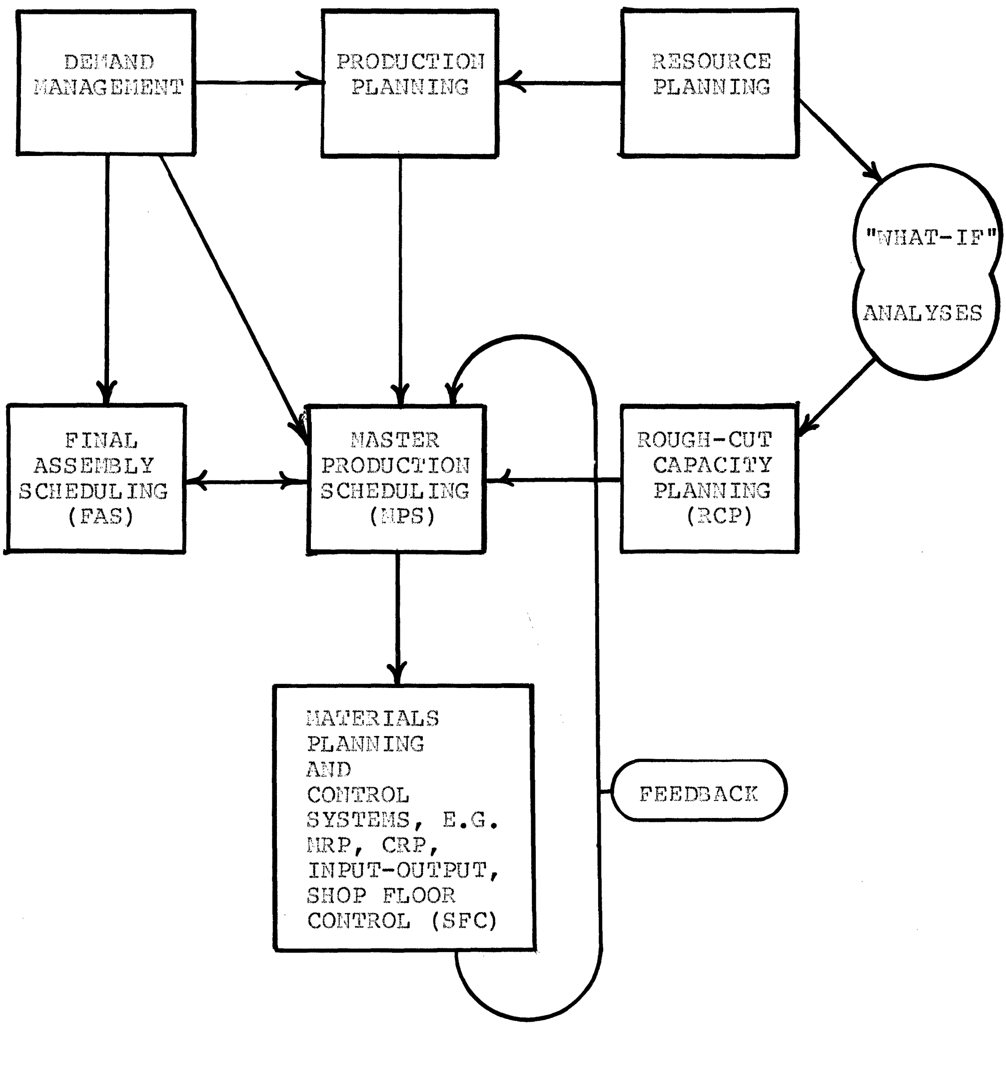
This paper investigates a software implementation of an order entry/materials management sytem interface using "planning bills of material" techniques to handle products with many options. The order entry system used for this paper was written by Arizona Industrial Management Systems (AIMS) for a division of Motorola and the manufacturing system was Hewlett Packard's Materials Management/3000 System. Before looking at the detail implementation of this interface and exploring the advantages of this technique, it is important to provide some background on implementation of MRP systems and a definition of Master Production Scheduling.

When MRP systems were first introduced there were many "trial and error" efforts to implement these systems. As APICS (American Production and Inventory Control Society) and companies began developing a better understanding of these systems, it became more and more obvious that Master Production Scheduling is the most critical management challenge in running an MRP system. Master schedules often times are unrealistic and out of step with available material and capacity. The results are that the shop cannot implement the master production schedule, service levels plummet, and inventories rise.

Figure 1.1 depicts the elements in manufacturing planning and control systems that relate directly to Master Production Scheduling. The arrows indicate the flows of information among these elements. As Figure 1.1 shows, there are six major areas to Master Production Scheduling. Below is a short definition of each of these areas and a set of principles which have been developed by companies who have successful Master Production Scheduling systems.

FIGURE 1.1

Relationship of Master Production Scheduling to Other Manufacturing Planning and Control Activities



DEMAND

Demand management encompasses the activities of forecasting, order entry, order promising and physical distribution. It also connotes the proper coordination of backlogged customer requirements, branch warehouse requirements, international requirements, and service part requirements.

Basic Principles:

1. A single, consistent forecast of demand is necessary.
2. Forecasts of aggregated product groupings improve forecast accuracy.
3. Forecasts of demand must include all business activities.
4. Physical distribution requirements and replenishment should be accounted for explicitly in the master production schedule.
5. The order promising activity must be tied directly to the Master Production Schedule.
6. When the MPS quantities are insufficient to satisfy a customer order promise, changes to the MPS should be evaluated against material (MRP) and capacity (CRP) availabilities.

PRODUCTION PLANNING

The production plan is a statement of the business stategy as reflected in the budgeted output for manufacturing. The output is generally specified in aggregate terms such as total dollar ouput, overall units of production, labor hours, or ouput of major product groupings. The production plan is management's control knob on the business.

Basic Principles:

1. The preparation of the production plan is a general management responsibility.
2. The sum of the parts must equal the whole.
3. The Master Scheduler should have the primary responsibility for disaggregating and monitoring the production plan.
4. Good production plans lead to reduced MPS execution problems.

MASTER PRODUCTION SCHEDULING

The master production schedule is a statement of the anticipated build schedule for selected products produced by a manufacturing facility.

Basic Principles:

1. The MPS must be in concert with the production plan.
2. The MPS drives the entire manufacturing system.
3. The MPS activity needs to be clearly defined organizationally.
4. All known Requirements should be used in preparing the Master Schedule.
5. Minimize the number of items that are needed to adequately express the master production schedule.
6. The MPS should be treated as a set of firm planned orders.
7. Stability in the master production schedule must be managed.
8. Closed loop MRP is important in insuring a realistic MPS.
9. Safety stock that is incorporated in the master schedule should be highly visible to the master scheduler.
10. The MPS should provide the basis for analyzing management "What If" questions.
11. The MPS system should be highly transparent.
12. The MPS should be evaluated with a formal performance reporting system.

RESOURCE PLANNING

Resource planning is the process of determining long-range capacity needs. The time horizon for resource planning usually ranges from one to ten years, depending on the lead time required to make capital investments in new production facilities.

Basic Principles:

1. A long range production plan or extended MPS should be used to plan new manufacturing facility requirements.

FINAL ASSEMBLY SCHEDULING

The final assembly schedule serves as the basis for planning and controlling final assembly and test operations in manufacturing firms. Once the production and procurement of component items have been completed, these materials are ready for release to assembly operations. The end-item product configurations for which these materials are destined to be used are determined by the final assembly schedule, which commits the company to the production of specific quantities of these products. The preparation of the FAS represents a critical time fence. When this time fence is crossed, and some products have not been sold to customers, they will be scheduled and may well become a part of the firm's finished goods inventory upon completion.

Basic Principles:

1. The final assembly schedule serves a purpose distinctly different from the MPS.
2. Avoid firming up the final Assembly schedule until the latest possible moment.

ROUGH CUT CAPACITY PLANNING

Rough cut capacity planning is an activity that involves an analysis of the master production schedule to determine the implied capacity requirements for critical manufacturing facilities. Such an analysis is often performed less frequently than capacity requirements planning (CRP), e.g. monthly or quarterly, often using less complex data, e.g. bills of labor rather than time-phased requirements and routing file data. Typically, this analysis serves as the basis for negotiations between marketing and production in making adjustments to the MPS.

Basic Principles:

1. The master production schedule should be translated into the capacity needs for key facilities to insure a realistic MPS.
2. The MPS should provide the basis for analyzing key resource requirements and trade-offs.

Each of the general MPS principles listed above could provide the topic for a lively debate, but the purpose of this paper is to delve into handling the demand from an order entry system to the master schedule for products that have many options. This is a small portion of the master schedule Demand Management problem, but it tends to be a major hurdle in beginning the Master Production Scheduling function. Since the Master Scheduling function is usually not well defined before the MRP system installation is started, the first Master Production Schedule tends to be a very rough forecast which is entered into the Master Schedule only to generate the MRP report. If the finished product has several options, the initial bills of material are usually defined with a unique structure/bill for each possible combination of options. After the initial MRP runs are completed and the effort is begun to formalize the master scheduling function, the initial selection of bills of material are usually expanded to complete the remainder of the option availability. As this effort is continued, it becomes very obvious that several problems exist. First, as the number of bills start to expand exponentially, the maintenance effort involved becomes more noticeable since any changes to the finished product must be included on all bills of materials which represent variations on the finished product. Second, the forecast accuracy becomes worse since marketing does not know the exact mix of options and/or will not take the time to generate a forecast for each possible combination.

A common technique to handle product options, which has been developed by APICS and progressive companies is included in the more sophisticated Material Management Systems. The technique is called planning bills of material. Figure 1.2 lists the general rules used to implement planning bills of material. These may vary slightly from system to system so a sample (Figure 1.3) of a planning bill of material and a detail description of how Hewlett Packard's Materials Management System implements planning bills of material in the Master Production Scheduling module is included.

Figure 1.2
Rules for Structuring a Planning Bill of Material

1. Define the top level part to a major product as a master scheduled part.

2. Structure all common parts and all options available to the master scheduled part number.

3. Define the structure quantity-per for the option to reflect the percentage of use for this option based on historical usage.

4. Safety stocks may be built into the planning bill by structuring an option mix with more than 100% usage. This allows for small month to month fluctuations in the option mix of actual orders.

5. The option mix must be maintained to reflect any major market changes since the planning bill is used to plan all long lead time item in the forecast. MRP generates the material plan from this forecast.

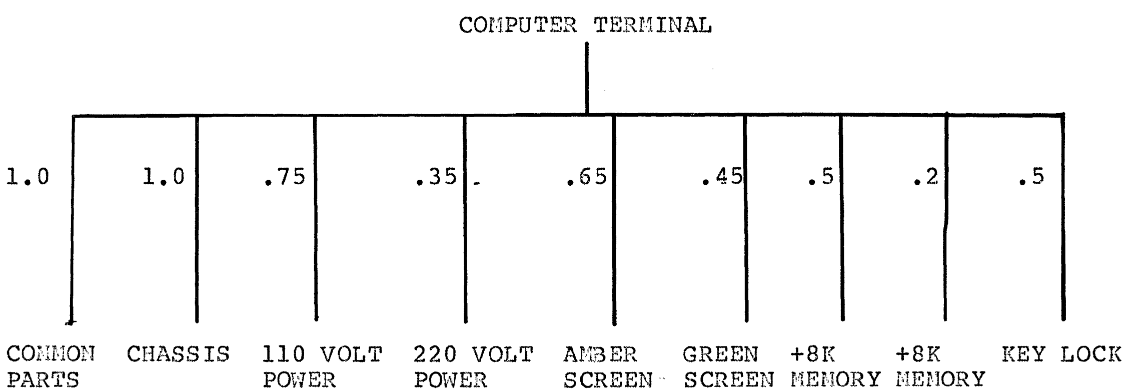
6. The master scheduling system must have the capability of defining the exact options ordered on all actual orders since pick lists must call out the exact parts required to produce the finished product.

Figure 1.3
Example of a Planning Bill

PRODUCT: Computer Terminal

OPTIONS AVAILABLE:

1. 110 Volt Power Supply
2. 220 Volt Power Supply
3. Amber Screen color
4. Green Screen color
5. First add-on module of 8k memory
6. Second add-on module of 8k memory
7. Keylock Option



The Hewlett Packard Materials Management/3000 System supports planning bills of material for Master Production Scheduling. Below is a list of the steps required to implement and use planning bills on MM3000.

1. To define a part as an option, the OPTION-FLAG on the item-data dataset must be set to "Y". The system will not allow a part to be defined as an option unless it is structured to a master scheduled part only (MASTER-SCHED = Y).

2. Any forecast orders set up in Master Production Scheduling will have a Run-Options record added for each option part. The quantity-per on the Run-Options dataset will be set to the quantity-per on the structure dataset times 100 which will convert the number to a historic usage percent.

3. If an actual order is entered into Master Production Scheduling it will also have the Run-Options records added for each option part. The options historic usage percent must be edited on the F_EDIT_OPTIONS screen. If an option is changed to zero, the option in effect is deleted from the order and will not be included on the pick list for the manufacturing order.

4. MRP uses the Manufacturing Orders for independent demand. If the Run-Options records have not been edited the demand used will be based on the historic percentage usage.

Below is a list of the problems encountered using the planning bill implementation in the MM3000 system.

- o It is time consuming and cumbersome to edit all of the options on each manufacturing order placed in the system.

- o If a forecast is entered into the master schedule, the forecast quantity for that period must be reduced as actual orders are entered into the schedule, otherwise demand gets included twice. This is sometimes refered to as the "shrinking MO" problem since orders in the master schedule are referred to as manufacturing orders (MO's) and they must be reduced as actual orders are entered into the MPS.

- o When costs are rolled up on the planning bill of materials all of the option costs are ignored at the master schedule part level, therefore Cost Accounting must manually add the option cost for each combination of options desired.

In an effort to address some of the short comings of the Materials Management implementation of planning bills and gain some addition benefits from the technique, the AIMS Order Entry System has been designed with some major interface capabilities to the Hewlett Packard Materials Management System. Figure 1.4 is a list of the design features included in the interface and a list of benefits associated with each feature.

Figure 1.4
Design Features and Benefits

	Feature		Benefit
I.	The top level of a product bill of material will be defined in the Order Entry System. This will include a definition of each valid combinations of options which may be ordered.	I.	With each valid combinition of options defined in the Order Entry System; o Orders can be validated as they are entered into the system. o Standard costs can be calculated for each order since the common parts costs are carried on the planning bill and each option ordered can have its costs accumulated.
II.	Order Entry part numbers must be able to carry a cross reference to the manufacturing part number	II.	Since often times the marketing organization in a company will use different part numbers, a cross reference capability allows each department to continue their current practices while relating both number to a common database. This cross reference is carried for the planning bill of material and each option. This enables marketing to define several products which are in fact just different options for the same product without affecting the product definition in the materials management system.
III.	An on-line screen (Order To Promise) is available to review the complete manufacturing schedule and the projected demand for each item to aid in setting the ship dates.	III.	This aids the master scheduler in assigning realistic ship dates based on the current manufacturing schedule.

IV.	The order requirements are added to the materials system automatically either by adding manufacturing orders with the exact option mix to the master schedule or by adding extra usage.	IV.	The cumbersome task of specifying the exact option mix for a manufacturing order is automated therefore master scheduling becomes easier. Detail reporting is available to enable the master scheduler to better balance the suppy and demand requirements within the master schedule. Since the orders are real customer orders and represent a working schedule, MRP can be run at any time.
V.	If manufacturing orders are added, the actual order quantity can be subtracted from any forecast orders during that period of time in the master schedule.	V.	By subtracting the new actual orders from the forecast, the forcast stays in balance. Reporting aids the master scheduler maintain a control of the total master schedule.
VI.	Any changes in the customer ship dates must be reflected in the materials system immediately.	VI.	Since all changes in the ship dates are made on line, the corresponding manufacturing orders can be checked to verify if production has started and if not the schedule can be changed. Therefore the master schedule is current. This eases the master scheduling task and enables MRP to be run at any time.

Since Order Entry System provides order data directly to the Materials Management System, the planning bill of material becomes a simple and easy way to handle products with many options and the master scheduling function is less tedious. By having the valid combinations defined in order entry, orders can be validated at order entry time, standard cost information can be accumulated for the exact mix of options ordered, and the requirements can be added to the requirements in the Materials Management System automatically. The simplification provided by using the planning bills offers many benefits to the manufacturing system users also. The planning bill of materials simplifies the definition of the product structure; ease the maintenance effort for the bills; and enables marketing to do a better job in providing forecasts. All of these factors smooth out the master scheduling function and has been shown to materially add to the success of an MRP installation project.

Biography

Jim Lawshe
is the President of Arizona Industrial Management Systems (AIMS). AIMS was incorporated 4 years ago to provide timesharing, consulting services, and additional software to users of HP's Material Management System. Before starting AIMS, Jim was a Systems Engineer and Industry Specialist for Hewlett Packard for four years. Prior to working for Hewlett Packard, Jim was the project leader on the Materials Management project at Arizona Public Service. Jim currently holds a Bachelors Degree and Masters degree in Electrical Engineering and is a certified Professional Engineer.

USING COMPUTERS TO CREATE BRAND NAMES

Roger Lawson

Proactive Systems Ltd, 110 New Bond St, London, England

SUMMARY:

This paper is concerned with the use of linguistics and lexicography in the creation of brand names with the assistance of HP3000 and HP150 computer systems. The development of such systems, which use computer software to assist in the creative phases of product name generation, is described. It is shown how a computer can help to generate ideas in a non-mechanistic way. Linguistic techniques are described that are used to ensure acceptability of created names in different languages. The problems of creating names for new computer software products are also discussed.

INTRODUCTION

It is natural to believe that the creation of brand names requires human imagination and inspiration. I will show how a computer can be endowed with some facets of these skills. A system is described that was developed specifically to assist in this activity for a company called the InterBrand/Novamark Group who specialise in the creation of brand names.

Let us first examine what makes a good new product name. It should have the following attributes:

- Originality so that it is both memorable and causes no confusion with competitors' products. The latter is also required if the name is to be acceptable in most countries for registration as a trade mark.

- It should have the right associations so that the product name gives the right message to the potential customer. However it should not be purely descriptive as such names are not protectable under trade mark law.

- If it is to be used internationally then it should be pronounceable to speakers of different languages and should not be rude or otherwise have the wrong associations in those languages.

Good examples that were created by InterBrand/Novamark are BRIGGAN (a beer), CARIBA (a soft drink), REPLAY (an erasable ball point pen), MICROJECTOR (for diesel fuel injectors), NOVANTRONE (a pharmaceutical), HOMEBASE (for DIY superstores), and VERSED (a pharmaceutical). Hewlett Packard has only recently chosen to start using product names instead of numbers but with no great skill so far eg. THINKJET for a dumb printer, TOUCHSCREEN which is purely descriptive.

How are most product names created? Somebody typically simply thinks them up in a semi-random manner. Certain organisations however use more controlled techniques including brainstorming groups, reference to dictionaries, etc. The aim of the computer system development that I shall describe was to assist in the creation process in such an organisation. The software tools that were produced run on an HP150 with supporting databases on an HP3000.

What is creativity? "To a great extent it appears to consist of noticing connections between things where no connection had been seen before" - see Reference 6. I will show how by analysing the subconcious processes that humans use and then synthesising them in a computer system it is possible to produce a creative system.

ABSTRACT NAMES

The type of name that is easiest to create (whether by humans or by a computer) is an abstract one. KODAK is an example where the word was a complete invention. It is simple to get a computer to generate names from random letter combinations but it will generate mostly unpronounceable names eg. KDA, KDB, KEA etc. The computer system that we have devised can also be instructed to use limited letter combinations eg. you can tell it that the name must contain a vowel in the second position. It also incorporates usage of vowels, consonants or consonant strings with the frequency with which they occur in natural words. Even with such limitations the computer can generate thousands of combinations in seconds - unfortunately although many will be good new names, many will still be unpronounceable!

SPELLING AND PRONUNCIATION CHECKS

To overcome this problem we have also incorporated algorithms to check that a created name can be both spelled and pronounced in one or more languages (presently rules for English, French and German are available). This can be done because there are certain conventions in each language for spelling and pronunciation. Even after using this technique to limit the number of names generated there can still be many hundreds of names generated that are all suitable for use as new product names. These are then presented to the computer user for review.

PHONETICALLY SIMILAR NAMES

Another way to create new names is by changing an existing word into a phonetically similar form (the initial word is chosen usually because it has the right associations). COMPAQ (a compact personal computer) or PHOOD (a restaurant in London) are examples of these. Again the computer has been programmed to do that. First the software takes an existing word and converts it into its phonemes - phonemes are the separate sounds that make up each word, to put it simply. The phonemes are then converted back into spelling - there are usually several spelling equivalents for each phoneme eg. PH and F are phonetically similar. By also randomly changing the phonemes slightly before reconversion one can produce many other possibilities (COMPAQ is an example where such a transposition was made). See references 1 and 2 for good introductions to linguistics - academic work in that field was used as the basis for the above systems.

WORD RECOMBINATION

Some product names are simply words or parts of words combined to form new words eg. INFOLINE, INFOLINK, DATALINK, DATELINE, INFODATA, TELELINK, TELECOM, DATACOM are all product or service names used in the UK. By submitting the words INFORMATION, LINK, LINE, DATA, TELEPHONE, COMMUNICATION etc to the computer system it can produce any number of similar recombinations. It can do this by recombining parts of words, or by looking for phonetic or letter similarities as the basis of a suitable point to break existing words. For example feed it INFORMATION and AUTOMATIC and it would produce INFORMATIC. By feeding in groups of words with relevent associations you get back newly invented names that still have a suitable "feel".

DICTIONARY USAGE

In addition to the above systems which run on HP150s, we have the equivalent of several different types of dictionary stored in an IMAGE database on an HP3000. It includes a normal dictionary, a synonym dictionary (synonyms are words of similar meaning), foreign language dictionaries etc. The entries are phonetically coded so that it is possible to retrieve any similar sounding word instantly. Such techniques using the Soundex algorithm have previously been widely used for telephone directory applications. However we have developed more advanced techniques so that a similar phonetic or spelling element anywhere in the word can be retrieved eg. if you type in the word CAT then it will retrieve CATKIN, CATCH, SCAT, SECATUR, SCATHE, KATION, CUT, etc. Likewise the synonym dictionary can be used to retrieve words of associated meaning eg. enter CAT and it might retrieve PUSSY, FELINE, TOMCAT, MOUSER, LION, etc.

You can see that both of these facilities can be used in an interactive dialogue to generate new ideas.

The former can also be useful if you want to retrieve a word that you have forgotten how to spell!

These techniques can also be of assistance in trade mark searching and other non-identical record retrieval problems (see later also).

If many new names have been generated by the processes described above then they can be automatically processed against the dictionary to see if there are any existing similar words in use.

Note that the dictionaries and associated retrieval mechanisms can be made available to other users if you have other applications for them.

INTEGRATED SYSTEM

By combining all of the above into a linked system with the addition of name list editing and printing/graphics it has been possible to produce a truely creative system. By putting the same words through the different processes one can produce names that are very distant from the original word or thought that was input. Moreover it does not replace human thought - the system often causes new ideas to arrive in the users' mind which he can feed back into the system. In the same way input from conventional brainstorming sessions can be integrated into the material. The user still has to perform selection from the mass of computer generated material - if anything the system is over creative.

It may be seen as partially deskilling the creative process but I prefer to view it as an aid to creative productivity.

NAME ACCEPTABILITY

By the use of the methods described above it is possible fairly readily to produce a list of 200 suitable names for a new product of which a high proportion will be acceptable to the client. From that point there are three further stages that need to be gone through before any name is used. These three stages are unfortunately often ignored by companies in the computer software field. They are as follows:

1 - Check it out for customer acceptability by doing some simple market research (this should not be overemphasised however, because very original names may not invoke much reaction at all or may seem incomprehensible at first sight).

2 - Consult a trade mark lawyer for his comments on its protectability - the fee can be a nominal amount.

3 - Look at existing trade marks to see if anybody is already using a similar mark.

TRADE MARK SEARCHING

As regards the last of the above three checks there are two procedures that should be followed. Firstly do a search of the trade mark registers in those countries in which you intend to sell the products. For example the US Trademark Register is available via the DIALOG network. With a $100 modem and an HP terminal you can get access from most countries in the world. Unfortunately in some countries (eg. USA, UK) you do not need to register a mark to establish rights in its use - many software companies do not therefore bother to do so because of the suspected cost, which is not very high in fact. Therefore you also need to do a search for usage. Again some on-line database services can help in this area, or refer to trade directories. We ourselves hold a number of registers and directories on our HP3000 so that we can automatically pass any list of created names against the existing trade marks. This is not as simple as it may appear because a non-identical phonetic search is necessary. For example, if one wanted to check out the name TOUCH then marks such as TUCH, TOUGH, OUCH, 2-CH, SCREENTOUCH might all be seen as conflicting if they were already in use on similar products.

NAME BANK

Once the above procedures are completed then the final choice of a name can be made from the list of about 200 that is typically created. Because the remainder of the material could be useful in future each list is stored away in a "name" bank (in an IMAGE database). It is categorised by type of product and by its associations (any English word, or number of words that are in the dictionary can be used as an association). Whenever any similar need arises in the future then the name bank can be used as one source of possible product names. This can be particularly useful if the product is near launch and no name has yet been fixed - this can often occur when a client has decided on one name himself and at the last moment discovers that someone else is already using it!

DISTRIBUTED SYSTEM

The system as it was designed provides a good example of a distributed processing system. Most of the processing concerned with name creation runs on the HP150s. Although the programs are quite large there is little need for large amounts of backing storage and it was required to be used at several locations worldwide. However the dictionaries, trade mark directories and name bank use a large amount of disc capacity and were required to be shared by all users. Hence an HP3000 is used for that purpose (based in London). DSN/LINK is used to transfer name lists from the HP150s to the HP3000 and vice versa using dial-up data comms. Remote offices are located in New York, Paris, Frankfurt and Tokyo.

TECHNICAL OUTLINE

The HP150 software was programmed in PASCAL using the ProPascal compiler from Prospero Software. The touchscreen functions were used wherever possible so that all commands were implemented as touch functions - the keyboard is only used when typing in a word or name. It was not found particularly easy to program the touchscreen functions so we wrote a standard piece of software to simplify it. The layers of software added by HP to MSDOS also make it difficult to program and use up a lot of memory. With MSDOS, PAM etc all in use then the HP150 operating system is now larger than was MPE on the HP3000 until recently. However the HP150 has a very unfriendly operating system with no security provisions and other features lacking.

The HP3000 system used is a Model 39. An IMAGE database is used for the file storage system and the software was written in COBOL - as usual with an HP3000, no problems or difficulties were encountered although it would be nice to have variable length fields in IMAGE for this kind of application.

FUTURE POSSIBLE DEVELOPMENTS

The next stage required in the development of the computer system described above is to provide it with better qualitative judgement on the merit of created names. This requires up-to-date knowledge of the environment in which the product is to be used and, as yet, there is no easy way of providing such knowledge in a computer system. The same problem occurs in the interpretation of natural language by a computer. Being able to parse a sentence using knowledge of English grammar is not enough for understanding of a sentence at other than a very elementary level eg. resolving ambiguities requires environmental background knowledge that humans acquire since birth but which computers do not have. However there is much academic research being done on this problem at present. See references 3, 4 and 5 if you are interested in natural language interpretation by a computer.

One way of representing knowledge is by a "semantic network". This is a network by which objects and attributes are linked in a complex inter-connected structure (see Ref. 7). For example, if one considers the word "car" then a conventional synonym dictionary will contain "automobile", "motor vehicle", etc. However car is simply one of the class "vehicles". Cars also have attributes of fast/slow, large/small, colour etc. Cars consist of engines, bodies, wheels etc. Cars normally contain people. Cars have a purpose of transport. Cars are restricted in function by law. All of these possible links represent information and are often used in the invention by humans of new brand names (the links can also represent the "associations" of a word). The task of building up such a knowledge network may be enormous - however there would be many applications for such a system.

CONCLUSION

A few years ago nobody would have envisaged that computer systems could be significantly useful in the business of creating names. I hope that I have shown that by breaking down the task into the various processes that a human uses when performing the task, it is possible to synthesise an apparently intelligent system. People only appear to be more creative because their performance of the tasks is unconscious and the processes used are not evident. The use of some of the techniques described in this paper (eg the use of a sysnonym dictionary to provide semantic associations) could assist in endowing many computer systems with apparent intelligence.

REFERENCES

1. Linguistics: An Introduction to Language & Communication; Akmajian, Demers and Harnish; MIT Press

2. Introduction to Theoretical Linguistics; J.Lyons; Cambridge University Press

3. Understanding Natural Language; T.Winograd; Academic Press

4. Natural Language Information Processing; N.Sager; Addison-Wesley

5. Computations from the English; R.F.Simmons; Prentice-Hall Inc.

6. The Creative Computer; D.Michie and R.Johnston; Viking

7. Associative Networks; Findler N.V; Academic Press

BIOGRAPHY

The author of this paper, Roger Lawson, is the Managing Director of Proactive Systems Ltd - an HP software company with offices in London and New York. He has a first degree in engineering and a masters degree in business administration. His computer experience spans 18 years of which the last 9 have involved usage of Hewlett Packard computer systems as programmer, systems analyst, d.p. manager and software supplier.

FUNDING PROJECTS MANAGEMENT SYSTEM IN THIRD WORLD COUNTRIES - AN APPLICATION OF HP3000

Vasant R Shroff and K Raghunathan

Blue Star Limited, Bombay, India.

I SUMMARY

Funding Agencies help Third World countries like India by organizing projects for knowledge transfer through training, seminars, visits of experts, etc. Other activities include supply of books and equipment, cultural exchanges, maintaining contacts with eminent individuals, etc.

The volume of activities and their diversity generate large amounts of data with complex inter-relationships and time criticalities. This requires systematic and timely data collection, organising of this data into an on-line data-base supported by tools for managing the same. The manual 'Card' system is quite inadequate in this environment.

The computerised Funding Projects Management System (FPMS) offers a solution by providing a mechanism for monitoring of projects, scheduling of activities, allocation of resources and developing/ maintaining contacts with individuals. It allows the Funding agencies to examine and manage projects by geography, source of funding, types of institution, discipline and officers-in-charge.

In this paper, application environment, data-base and system structure are discussed in the following sections, concluded by implementation review, in the context of a specific implementation experience.

II APPLICATION ENVIRONMENT

The significant points in the scenario are as follows:

a) The site does not have previous history of computerisation. A manual 'Card' system is currently used in conjunction with various reports, charts and tables. The growing load and information needs place strain on the operation, necessitating computerisation.

b) The end-users are functional managers and their staff. The system must be friendly. It should also prove its usefulness enough to make users rely on it and discard 'parallel' chits of paper.

c) The site keeps a close watch on hardware, software, maintenance, training and staffing costs. The costs are not unduly apportioned to future applications and benefits.

d) The data-base is quite rich in its content and relationships. The system activity is inquiry for most part. The update transactions are simpler in terms of their processing requirements while the inquiries could involve several data sets.

e) Some areas of the data-base show higher incidence of update transactions than other areas. Nevertheless the overall design is to optimise inquiry based on the user identified search items.

f) Given the structure of data-base, the time and the resources required to develop a comprehensive software system would be enormous, complicated further by evolving nature of user requirements from base level. In view of these factors, the software system structure is simplified which meets the requirements for the next 3/4 years adequately and which could be implemented with much less time and resources. For the same reason it is decided to use standard HP software wherever it serves the purpose.

III <u>DATA-BASE</u>

The pivotal points of the data-base are entities, which are broadly classified into main and secondary entities. Main entities constitute the core while secondary entities are used to define certain attributes. Major entities are identified with brief explanations in this section:

A) <u>Main entities</u>

Project : Defines various projects undertaken by organizations. Examples: Research & Development of Brick structures, Himalayan Geology, Performance of Boilers, Reusable fuels.

Activity: A project consists of various activities. Each activity is required to be monitored periodically. Examples: Consultancy on mechanical maintenance of turbines, Course in Hospital Administration, Visit of Specialist in Neurology of Leprosy.

Institution: Defines organisations associated with projects and activities. At any time, an institution may be participating in several projects and conversely a project may encompass several institutions. Similarly an institution may be hosting several activities at one time and conversely an activity may involve several institutions.

Department: An institution consists of various departments which specialise in specific areas of technology, science or any other discipline.

Individual: An individual is employed in a department within an institution. The individual may be engaged in several activities of one or more projects in a time span.

The Figure 1 shows the Main entities and their relationships which are 'one-to-many' or 'many-to-many'

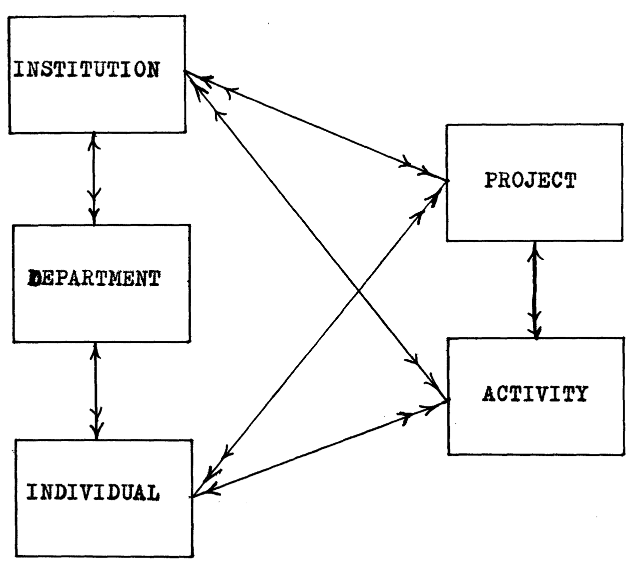


Figure 1: Natural Structure of Data-base

B) Secondary entities

Town:	Defines place of residence for an individual or location for a department or an institution .
State:	Defines state of the town.
Region:	Defines geographic region in which the state is included.
Subject:	Describes area of specialization for main entities. Examples: Ventilation Engineering, TB research, Transportation.
Sector:	Describes a broad-based classification of industrial sectors. Examples: Coal, Railways.
Activity scheme:	Defines funding agency's plan, program or scheme under which an activity is sponsored.
Activity class:	Types of activity such as visit, equipment, transfer, etc.
Institution type:	Types of institutions such as research, academic, etc.

C) IMAGE Schema

The natural structure exhibited by the data-base as shown in Figure 1 is mapped onto two-level network using 'one-to-many' relationships within IMAGE framework.

It is also required to maintain narrative information in the form of text about entities within the data-base ; for example, report on a visit, which is in the form of free text.

Many search items are long alphanumeric strings such as names of institutions and individuals. Spelling long names correctly and consistently poses problems. Hence names are assigned codes wherever required. The user however is not familiar with these new fangled codes. The software permits reference by name or code as a solution.

Codification also helps in conserving the disc space since search item is physically stored by IMAGE once in each data set concerned.

The data-base supports 33 search items and over 100 other information fields, organised into 25 detail data-sets and 33 master data-sets.

IV SYSTEM STRUCTURE

The major modules of the system provide for the following functions:

- Information Retrieval
- Information Maintenance
- Word-Processing Interface
- Archival

A) Information Retrieval

This module serves the most important function of providing the necessary reports to enable the user to effectively monitor activities and projects. It consists of the following sub-modules as shown in Figure 2.

- Statistical and Status reports
- Standard Queries
- Adhoc Queries

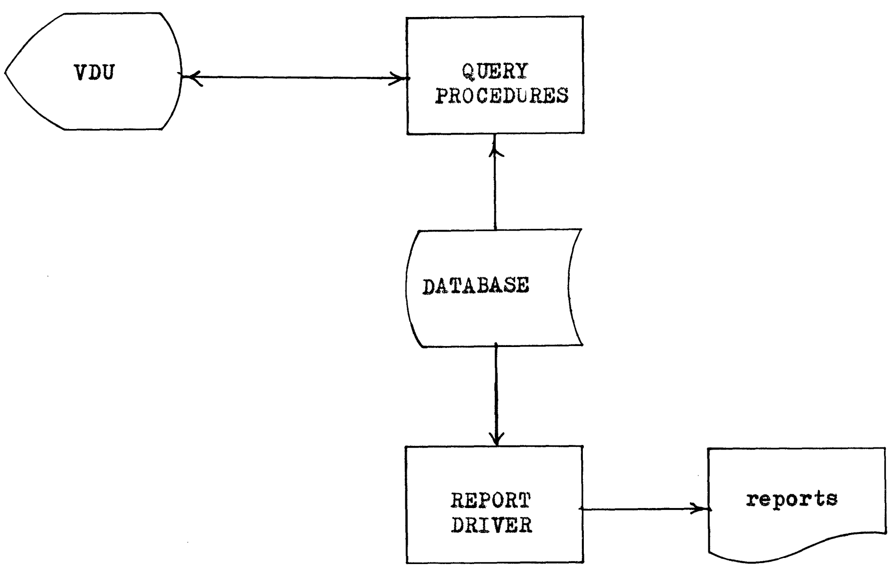


Figure 2: Information Retrieval

Statistical and Status Reports

A menu-driven report driver accepts user selection and triggers the relevant report generating program in background, freeing the terminal for further use. The various reports, useful for periodic monitoring of projects and activities, include the following:

Collaborative Activities Report: This report gives scheme-code wise activities within a specified time-frame listing out names of individuals and institutions participating in each activity along with funding code and activity narration.

Quarterly Visitors Report: This report gives sector-wise list of visitors inward or outward within a period along with institutions visited, status of the activity and duration of the visit.

Project Commitment Report: This report gives list of ongoing projects and information on the activities under these projects.

Project Status Report: This report gives financial status of each project with expenses under different heads shown against agreed inputs. Also shown are expenses for current quarter and next quarter.

Annual Scheme Report: This report gives scheme-wise progress of projects and within projects for each activity and the persons involved in it.

Standard Queries

HP's Query software is used to answer standard queries. A set of 50 standard queries made from time-to-time are identified and 'canned' query procedures for these queries are maintained in a library. If the performance of Query is found to be not acceptable, especially for the more frequently used queries, small programs could be developed later to handle some of these queries.

A directory of procedures is compiled and functionally grouped. A Query user could look at the directory and determine the procedure required to perform a specific function. When the selection query procedure is executed, embedded messages are displayed prompting the user for input values. These additional prompts are necessary because Query itself uses for prompting, data-item names which are not self-explanatory. The prompt messages are stored in a 'Help' data set and are customizable.

Some of the procedures are as follows:

- Given code, find name for institution, individual, department, project, activity, subject, town, etc.
- Find equipment supplied to an institution.
- Find all individuals by area of specialization.
- Find all projects within a financial year.
- Find all visits handled by a responsible officer of funding agency.
- Get summary of funding agency's involvement with an institution in terms of projects, activities, contacts, etc.
- List information about individuals such as birth date and dietary habits, useful for maintaining social contacts.

Ad-Hoc Queries

HP's Query is best equipped to handle ad-hoc queries. However, it presumes that the user is trained on using Query and is familiar with data-item names defined in the schema.

B) Information Maintenance

This module provides for basic processing of transactions, browsing through the data-base and performing other utility functions. It consists of the following sub-modules, as shown in Figure 3.

- Walk
- Delete
- Code
- Help

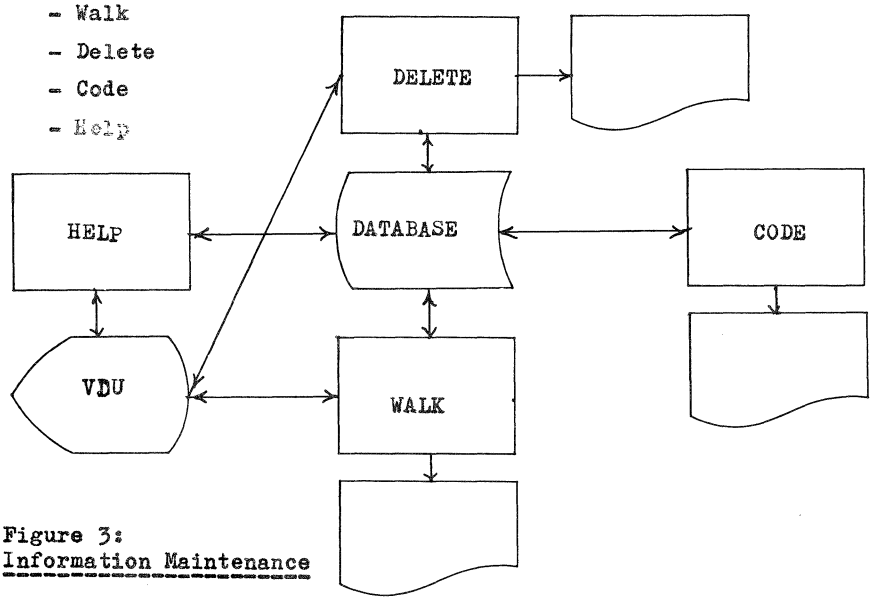


Figure 3:
Information Maintenance

Walk

Walk constitutes the heart of the entire system. It serves the dual purpose of processing transactions and of browsing through the database. Operating in 'Add', 'Browse', 'Change' or 'Delete' mode, the user is able to add new entities, recall on the basis of search items and optionally modify information. The interaction is through user-friendly screens combined with heavy use of soft-keys. A customizable 'Help' facility is available at every stage to assist the user.

The normal sequence of the screens and the possible transitions between screens require to be organized carefully in view of the dimension of the data-base. For the same reason, the user is required to commit on the mode of operation at the outset.

At a later stage, separate programs could be devised for processing specific transactions and also for maintaining the invariant information in various data sets.

Delete

This module enables the user to delete main entities with all the associated information scattered in different data sets.

Code

The codification of search items is introduced for reasons explained earlier in this paper. A certain amount of activity is expected to take place in regard to reassignment or rectification of codes. This module executes change of codes wherever they occur in the data-base, thus maintaining the integrity of the data-base.

Help

This module is used to maintain information in the 'Help' data set. The information is organised as one or more lines under a key. Any program wanting to provide 'Help' supplies the key to a procedure which retrieves lines of information stored under that key formats and displays on the screens. Thus the program is delinked from 'Help' messages. The information stored under the key itself is customizable using this module. Walk module uses it extensively. The 'canned' Query procedures described earlier also use this feature for issuing prompt messages.

c) <u>Word Processing Interface</u>

This interface allows use of HPWORD to produce mailing lists, combined with the powerful retrieving capabilities of Query as shown in Figure 4. In actual use, the Query procedure is invoked first and a file is created for selected entries. The interface program reformats the file into a form acceptable to HPWORD. HPWORD accepts input from this file and generates the mailing lists.

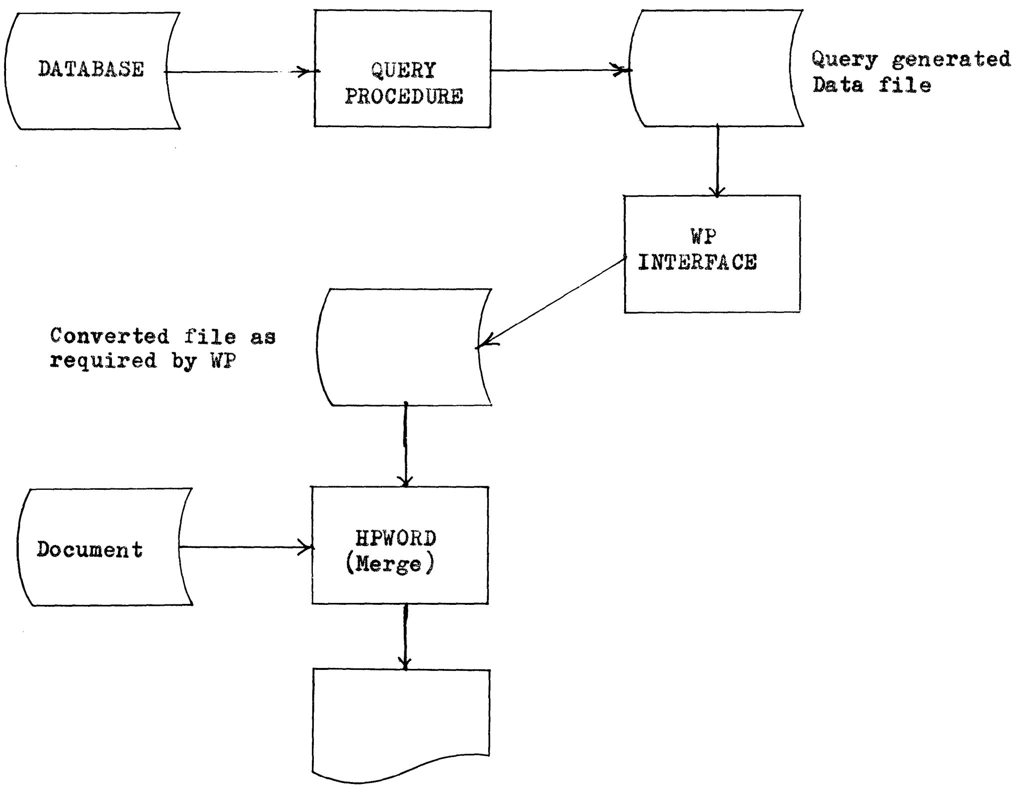


Figure 4: <u>Word Processing Interface</u>

With the availability of HPWORD intrinsics, now it may be possible to cut down some of the intermediate steps.

A similar interface to DSG (Decision Support Graphics) software is also conceived, but not implemented.

D) Archival

This module provides for selective backing and deletion of information from the data-base as shown in Figure 5. It is possible to back up entity-wise unlike the normal DBMS utilities. Thus, for example, information on projects may be archived by status and/or before/after an update date, or information on individuals may be archived on the basis of contact date.

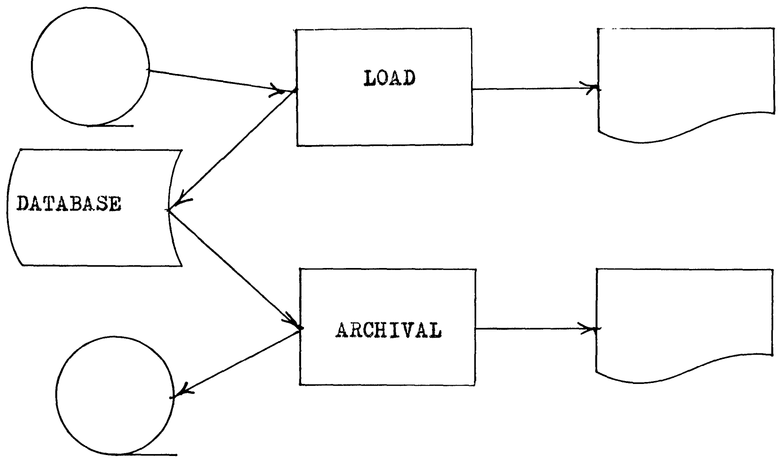


Figure 5: Archival

When the projects information is pulled out, all the related information such as on project related activities are also extracted for completeness.

A complementary utility is available to restore the information to the data-base.

V IMPLEMENTATION REVIEW

The specific implementation experience brings up a few significant points which are relevant in similar environments elsewhere in the Third World community, where computerisation is receiving increasing attention.

The classic approach of discussing and freezing the system design before commencing the development does not always work out. While the user is unable to contribute substantially in the beginning, interaction is much more as the project gets underway. Under these conditions, developing the system prototype first and then filling it out with details in steps, could be an alternative approach. Also, it could be planned out such that the implementation results in the first generating software for the site, fairly quickly, which performs certain basic functions. It then evolves into a more comprehensive system, as the requirements become more well-perceived and as the user awareness also increases. In this phase functionality may take precedence over system performance.

At the other end, it may also happen that the user expectations are very high. Setting them at the right level is necessary for successful implementation. A system appreciation programme in this context proves useful in preparing the user for his role.

It also needs to be emphasized sufficiently strongly that the system's usefulness is only as much as the accuracy of data going into it. The information content in the data-base must thoroughly be justified by the user for its inclusion and the resulting trade off/benefits should be evaluated jointly. Formal and informal techniques are available to the user and the designer to assist in this exercise. It should be borne in mind that just as the information content in the data-base increases, data collection and ensuring of its accuracy become more burdensome.

The standard software available from the vendor such as HPWORD and DSG greatly enhance the functional capabilities. Suitable interface programs may have to be developed and the operation may not be in a single step, but the capabilities that now become available to user far outweigh these factors.

The use of productivity tools such as Dictionary, Inform, etc should be seriously considered. The system development time would be significantly reduced by using these tools. The report generating software such as Inform and Report are friendlier to use than Query and provide more capabilities in handling reporting requirements. For reasons of performance or for non-standard requirements, specific programs could be developed later.

The use of macros in COBOL II enhances modularity in program design. In fact, a well-designed set of macros, could be set up as the application system intrinsics, insulating the main code from details. The macros also enhance the readability of the code.

The concept of capabilities as defined in MPE could be extended to application system, where a matrix of functions and user identity is set up to control user access.

Program structuring, processing soft-keys and consistent handling of error situations require careful consideration.

Some statistics in regard to the development effort involved are given in Figure 6.

Figure 6:Development Statistics

Program Size

Walk	10000 lines (COBOL)	17 Screens.
Reports	8500 lines (COBOL)	10 Screens.
Others	10500 lines (COBOL)	3 Screens.

Total Development Time

Systems Analyst	59 mandays.
Programmers (2)	324 mandays.

Elapsed Time 9/10 months.

The specific implementation of FPMS which is the subject of this paper is successful in terms of user acceptance, modular structuring and flexible design and documentation. The user has even introduced small changes within a month of its operation to enlarge the scope of such a large monolithic module as Walk (Refer to Figure 6). It is expected that the system would undergo a few more changes carried out without pain as more and different requirements are perceived.

Biography

Vasant R Shroff

is the Manager, Software Export, at Blue Star Limited, Software Development Centre in Bombay, India. Blue Star is developing software for HP3000 users in India and overseas. Vasant is a Master of Technology from the Indian Institute of Technology, Kanpur, India. He has over 14 years' experience in software development and project management in the USA, Holland, Middle East and India. Vasant is married and has two children.

K Raghunathan

employed with Blue Star Limited, Bombay, India for over 12 years, is presently Manager, Systems. He is in charge of the System Engineering activities for the support of HP systems in India and is also responsible for the development of software at the Software Application Centre of Blue Star.

DB - DATA BASE MANAGEMENT SYSTEMS

REMOTE DATA BASE ACCESS - TO NON-HP COMPUTERS

Rolf Frydenberg
Computer Toolware, Inc., Toledo, USA

Summary

This paper presents a new approach to accessing data on remote computer systems. The approach presented is designed for accessing data which is not accessible through standard MPE intrinsics (file-system, KSAM or IMAGE), i.e. data which is not on an HP3000 connected through HP's DS product.

The approach presented is valid for different computer systems, connected to the HP3000 in a variety of ways. Where we present details, they are mostly related to accessing data on IBM mainframes connected to the HP3000 through HP's IMF subsystem.

We have called our approach "Virtual Data Base" (VDB), to indicate that the data accessed is treated as if it were organized as a data base. The actual data structure on the remote computer system has no direct influence on the VDB approach; it can be flat files, index sequential files, or any kind of data base.

The VDB approach consists of two major components: A method for describing how to get at the data (data access paths), and a set of procedures that may be called by programs that desire access to the VDB. This set of procedures is independent of the connection method and the actual data structure on the remote computer system.

The Structure of a VDB System

Central to our approach, of course, is the concept of a "Virtual Data Base" (VDB). Let us therefore begin with a definition of a VDB, as well as its main entities - Virtual Data Sets (VDS) and Data Access Paths (DAP).

The definition of a VDB is:

> "A VDB is a structured collection of data, acessible to a user through the services of a computer or a terminal."

The definition of a VDS is:

> "A VDS is a grouping of data items accessible through one or more specific commands and in a specific format."

The definition of a DAP is:

> "A DAP is a sequence of commands and operations that may be excecuted by a computer, and which give access to a VDB or a VDS."

VDB's can - and should - be compared with accessing data on a computer system from a remote terminal. The VDB may be compared with an appliction or other major subdivision of data on the remote computer, for example a specific program (e.g. the inquiry program for the accounts payable system).

The VDS is an entity within the VDB, commonly a specific transaction within the application, a specific data entry/retrieval screen, or a specific set of commands in a query program.

The DAP is a computer executable description of how to perform functions within the VDB/VDS system: How to get to the VDB in the first place, how to select a specific VDS, and how to perform operations on a VDS (get and put data for example).

The fourth, and final, entity in a VDB system is the set of procedures that allow a program to access and manipulate the Virtual Data Base. We have identified and defined six such procedures: OPEN and CLOSE start/stop access to a VDB, SELECT defines which VDS to access, GET, PUT, DELETE access and modify data in the current VDS.

The Relationship Between the VDB and the VDS

The VDB-VDS combination is a two-level hierarchy. In many real-life applications, the line between the VDB and the VDS is not as clear-cut as it may seem from the definition given earlier on. Often, we may wonder exactly what parts of an application are elements of the VDB as such, and which exist at the VDS level.

The most frequent cause of this dilemma, is that most applications really consist of more than two levels, and we must decide at which level in the application the VDS begins. To solve such dilemmas, the best rule of thumb is to find the level to which we most often return when switching between transactions; this should be the level at which the VDB ends, and the VDS starts.

In some cases, it might be relevant to define one real data base or application as several Virtual Data Bases. This decision must be based not only on the structure of the application and its data, but perhaps even more on the way in which the users access the data.

Let us look at an example, using HP's QUERY program, running on an HP3000, as the application which functions as our VDB, and accesses an IMAGE data base. There are four levels involved in this VDB:

LEVEL 1:	HELLO USER.ACCT	(logon determines which data bases you are permitted to access).
LEVEL 2:	RUN QUERY.PUB.SYS	(other programs would give access to other data structures).
LEVEL 3:	DEFINE data base	(other data-bases would give access to other data).
LEVEL 4:	FIND ITEM=KEY	(data access operation).

The separation into the VDB and VDS portions is wholly user-dependent; the split can be made anywhere in this typical hierarchical "tree" of transactions and applications.

In the above example, we could make the split between levels 3 and 4, so that the OPEN call would perform the logon, RUN QUERY and DEFINE the desired data base. A SELECT would in this case have no function, since we are already at the level where we can directly GET, PUT and DELETE data.

Another possible level for the VDB-VDS split is between levels 2 and 3. If we did that, the level 3 operation (defining what data-base we want to access) would be performed by SELECT; and GET, PUT and DELETE would take it from there.

Data Access Paths

DAP's are an essential part of the VDB concept. The description of a DAP explains to the controlling parts of the VDB system how to get to the data in the VDB, and how to perform operations on this data.

Path description methods are not universal. The path description method depends to a large extent on the type of connection that we have to the remote computer system. The essence of a path description is to provide a description of how things would be done manually, so that the VDB system can follow this description, and "emulate" the user.

For an IBM3270 type of connection, which is the connection used by VTS/VDBS, the path description uses concepts such as field numbers, cursor addressing, special function keys (PF-keys, PA-keys) and other specifics of emulating a user sitting at an IBM3270-type terminal. Input can be in any location on the screen (except in protected fields), and output data may also be located "anywhere" on the screen.

But these same concepts, which are essential to the IBM3270-connection, would be void in many other cases. One typical example is a connection which emulates a teletype terminal: Reference to where data is located would be "x lines back on the listing, in column y"; all input would be without addressing, since it takes places wherever the carriage happens to be; and the number of special function keys is limited to a few control characters and the carriage return.

If the connection emulated an HP-terminal, then the data access path might need to provide for both the teletype addressing method, as well as for "format mode" which is closer to the IBM3270-type connection. A program like QUERY uses essentially a teletype-structured data communications approach, whereas applications that use VPLUS are in format-mode, and therefore use a full-screen approach more like the IBM3270-type connection.

Even though applications on an IBM mainframe can be "almost teletype", there is one significant difference: The terminal itself does not support the "scrolling" of data, so that the user - or the application - must clear the screen when it has been completely filled. An IBM3270 connection can therefore always be assumed to be full-screen.

DAP Description

We have already looked at some of the essentials of path descriptions. Let us now look at some of the details.

A path description facility must be able to handle the following:

TEXT INPUT - this is for inputting commands and data; constants for specifying application "ABC" or transaction "XYZ", as well as variables that are passed from the program that calls the procedures.

TEXT OUTPUT - this is for returning data to the calling program; mostly data retrieved from the VDB, but also (possibly) constants such as messages that tell the end-user application something (e.g. error messages).

SCREEN COMPARES - the comparison of a part of the screen with a parameter or constant; of particular importance is the ability to check for error messages and other "special" output from the remote computer system.

SCREEN ADDRESSING - specify a location in the screen; for text input and output, as well as screen compares. The actual type of addressing depends on the type of connection through which we access the VDB.

FUNCTION KEYS - specify that a function key should be pressed; particularly in full-screen terminal emulation, but also to some extent for emulating control-keys on a teletype-connection.

BRANCHING - the path description must be able to execute differently depending on some condition; typical examples are: branch if an error occurred, branch if location (x,y) contains an error message, etc.

The VTS/VDBS system uses a simplified "programming language" to describe paths. This language is derived from the "automatic dialogue faclity" of another Computer Toolware product, VTS/IMAS. This language contains commands such as:

```
INPUT    - to input a text constant in a specific location.
INTEXT   - to input text passed from the calling program.
OUTTEXT  - to pass text back to the calling program.
CHECK    - to compare a text string with a location in the screen.
CURSOR   - to move the "cursor" to a specific location.
TRANSMIT - to transmit the screen to the mainframe, specifying which
           function key was "pressed".
GO       - to branch to a specific location based on a condition.
```

There are approximately 20 other commands available, which handle special functions.

Screen addressing in VTS/VDBS is by cursor position as well as by field number, so that the programmer who sets up the DAP's can choose the approach most valid for a specific VDB. In some cases, a combination of the two methods may be the best one to use.

For performance reasons, VTS/VDBS uses an "assembler" to translate the user's DAP's into a compact, internal format. This assembler uses an input structure consisting of named "transactions" which contain up to five DAP's or "operations". The five DAP's for each transaction are:

ENTER - which is executed by the SELECT procedure.

EXIT - which is executed by SELECT or CLOSE to exit the VDS.

GET, PUT and DELETE - which are executed by the corresponding procedures.

Additionally, there are the "global" ENTER and EXIT procedures (executed by OPEN and CLOSE, respectively), and certain global variables needed for opening the communications connection to the mainframe.

The VDB Procedures

The user's program, which wants to access the VDB, does so through calling a set of procedures. These procedures are independent of the connection used, and independent of the actual VDB structure. They are as important a part of the VDB concept as the data access paths.

Any access to a data base can be described by one or more of the following data access operations:

GET - retrieve data from the data base.
PUT - add data to the data base.
DELETE - remove data from the data base.

These three operations are sufficient to do "anything" with the data in a VDB, and are therefore the three central VDB procedures. More complex operations an be performed by combining several of these "primitive" operations into a user-level transaction.

But, in addition to performing data access operations, we need to access the VDB itself, as well as the VDS's that it consists of. For these purposes, we need three more procedures:

OPEN - specify the desired VDB and initialize access to it.
CLOSE - terminate access top a VDB.
SELECT - specify a VDS and initialize access to it.

It is in integral part of the SELECT and CLOSE procedures that they terminate access to the current VDS (if one has been selected). Therefore, there is no need for a special "DESELECT" procedure.

Using a VDB consists of the following two "loops" of procedure calls within an application program:

```
OPEN the VDB.

    REPEAT
           SELECT a VDS.

           REPEAT
                  GET or PUT or DELETE data.
           UNTIL  no more need for this VDS.

    UNTIL  no more need for this VDB.

CLOSE the VDB.
```

Let us now look a little closer at these procedures, to see how they work. We will illustrate the functions of the procedures with details of how they work in VTS/VDBS.

The OPEN Procedure

The purpose of the OPEN procedure is to initialize the VDB system on the HP3000, and establish communications with the VDB. The OPEN procedure should:

- Initialize the VDB global variables.
- Establish the connection to the remote system.
- Execute the initial DAP.

In VTS/VDBS, a "VDB Global Area" is used to keep track of global variables, including the status of the current VDB. The OPEN procedure initializes this data area. The data area exists as an array that must be passed to all the VDB procedures.

The CLOSE Procedure

The purpose of the CLOSE procedure is to terminate the connection to the VDB in an orderly manner. The CLOSE procedure should:

- If currently in a VDS, exit it.
- Execute the terminating DAP for the VDB.
- Terminate the communications subsystem used.

In VTS/VDBS, the global data area is inspected to determine the whether the first step needs to be executed, and which DAP to use to exit from the VDS. Then the terminating DAP is fetched from the DAP file, and executed. Finally, the connection to the mainframe is terminated, and CLOSE terminates.

The SELECT Procedure

The purpose of the SELECT procedure is to execute a DAP that allows access to a specific VDS in the current VDB. The SELECT procedure should:

- If currently in a VDS, exit it.
- Execute the DAP to enter the specified VDS.

The first step is the same as in the CLOSE procedure. The second step is to search a "transaction table" for the VDS name passed to SELECT, read the required DAP from the file, and execute this DAP.

The PUT, GET and DELETE Procedures

The PUT, GET and DELETE procedures all perform essentially the same operations:

- Execute the DAP for the relevant operation, using the data passed in the procedure call as input, and return data from the VDS as specified in the DAP.

VTS/VDBS first determines whether the user is actually in a VDS where the operation requested can be performed. The relevant DAP is selected based on the "transaction table", read from the file, and executed. The input data is passed to the DAP executing procedure (an internal VTS/VDBS procedure), and output data generated because of the DAP statements are passed back to the calling program when the DAP terminates.

Conclusions

In this paper we have looked briefly at a new approach to remote data access called Virtual Data Base (VDB). The presentation has concentrated on concepts rather than details, though with some practical examples on how the VDB concepts have been utilized in a specific VDB-type product for accessing data on IBM mainframes.

The major advantage of a VDB approach, is that the same structure can be used to access data over widely varying connection types. Much of this presentation has concentrated on IBM3270-type connections from the HP3000 to IBM mainframes, but many other connections will be of great importance in the future. Some examples are:

- X.25 network connections.
- Asynchrounous teletype connections.
- Local Area Networks.

In many cases where we need to access data on remote computer systems, we will have to have a different lower level, i.e. different ways of executing the Data Access Paths. But the VDB structure can help us develop systems for accessing data in a manner that makes it unimportant to the application program whether the data is on a remote IBM mainframe on an IBM3270-type connection, on a Local Area Network file server, on a data base service accessible through a public packet-switched network, or through any other connection type available to us.

Biography

Rolf Frydenberg is the Manager of Technology at Computer Toolware, Inc., in Toledo, Ohio, USA; a company which specializes in software for HP3000 to IBM mainframe communications, and which he helped found in 1983. Rolf Frydenberg has contributed papers to the HP3000 IUG conferences in Copenhagen (1982), Montreal (1983), and Edinburgh (1983). He has also contributed papers at several national Users' Group conferences. Many of his contributions have been in the data communications area, with special emphasis on HP3000 to IBM mainframe communications.

Rolf Frydenberg has a Masters degree in Computer Science from the Norwegian Institute of Technology in Trondheim, Norway, from 1980. He has 5 years' experience with the HP3000.

4th GENERATION DBMS AS THE HEART OF FUTURE SYSTEMS OF THE AUTOMATED OFFICE

Dr. Sreekaanth S. Isloor
Cognos Incorporated
275 Slater St., 10th Floor
OTTAWA, Ontario, Canada
K1P 5H9

SUMMARY

As the electronic office of the future becomes the reality of today, tasks formerly performed by office workers are delegated to computer driven technologies. Most of these tasks revolve around the creation, storage, retrieval, presentation and communication of information. Transaction oriented 4th generation database management systems (DBMS) are extremely important as a common link between the diverse types of personal and organizational office data. However, adaptation of the DBMS technology to the merging office, needs to be better understood. To be responsive to the special requirements of the distributed environment of the office, a modern DBMS must significantly differ from its traditional data processing ancestors. An analysis of the office issues is performed in this paper with particular emphasis on the role of the DBMS in their implementation. Consequently, a partial list of requirements for the new generation of DBMS, so as to facilitate integration of office functions effectively, is identified.

1. INTRODUCTION

Processing data to create information is essential for effective business management in today's world of exploding sets of facts and figures. The Computer powered office provides a means for conducting these modern business activities. The value of a sophisticated, fast, and reliable information system is constantly increasing as it provides, at acceptable costs, accurate and timely information for major strategic planning and tactical decision making. As the electronic office of the future becomes the reality of today, tasks formerly performed by office workers are delegated to computer driven technologies. Most of these tasks revolve around the creation, storage, retrieval, presentation and communication of information.

Computer systems increasingly depend on distributed components to increase performance while providing autonomous computing. This evolution taking place in the office of today, implies that corresponding advances must take place in software that integrates the functions of an organization. A database management system (DBMS) is a software that facilitates storage and retrieval of text messages, voice messages, facsmile, records and data as people exchange information.

Transaction oriented 4th generation DBMS are extremely important as a common link between personal and organizational office data.

Section 2 of this paper identifies the diverse types of data that play a role in the office of today. The "unstructured" nature of these data (in the sense that its structure is not predefined for purposes of electronic storage) plays an important part in determining the characteristic features of a DBMS that can support it. The role of such a DBMS of the automated office as the centre of activity is characterized.

Section 3 considers office issues and their requirements vis-a-vis the data base management system functions. In particular three office issues: Activity Management, Word Processing and Electronic Mail/Messaging are analyzed. The analysis of these is performed to clearly define the subfeatures in each of the office issues and to map these subfeatures to DBMS functions to facilitate their implementation.

In Section 4, the functional features of a DBMS are mapped against electronic office issues. Also appropriate system data flows and interrelationships are identified. A partial list of requirements for the new generation of DBMS are determined to facilitate integration of office functions effectively. A brief report on database research efforts applicable in offices of the future is provided.

Section 5 summarizes the impact of DBMS in the electronic office.

2. 4th GENERATION DBMS OF THE AUTOMATED OFFICE

An office information system is "a distributed operating system with a highly refined user interface and database facility" [1]. DBMS which controls databases provides: independence of file definitions from programs, data manipulation capability to perform all input/output operations, and simplified application programming against the database. Database management systems offer many attractive features including: multiple views of data, database flexibility, non-redundancy of data, security controls on data, consistency and integrity of data, self-description of databases, concurrency controls, access optimization, failure protection, checkpoint/restart, auditing facilities, query processing and an interface with programming languages and high productivity development tools. Therefore a DBMS makes it feasible to access integrated data that crosses operational, functional and organizational boundaries within an enterprise. Although, in the past, the DBMS have been reserved for the mainframe and minicomputers, microprocessor-based systems are now ready to cash in on these DBMS benefits [2]. <u>To be responsive to the special requirements of the distributed environment a modern DBMS must be significantly different from its traditional data processing ancestors</u>.

Data in an electronic office consists of reports, memoranda, letters, contracts, forms, charts, graphs, pictures, telephone messages, financial data etc. These diverse types of data are "unstructured" in the sense that their structure is not predefined for purposes of electronic storage and retrieval. A typical limitation of many data processing systems is their inability to handle "unstructured" data. To be useful in supporting office work it is essential that a DBMS provide capability to store and manipulate the diverse types of data in an integrated manner through a uniform user interface. A single, uniform mode of interaction to handle diverse types of data results in higher productivity due to reduced interruption in work flow.

In an office environment the users' will interact with their work stations to do word processing, electronic mail, electronic filing, data entry, graphics output, activity management, security and integrity assertions, traditional application development and data processing, or Analytic Support. A 4th generation DBMS that supports these automated office features is essential, but must be hidden from view (Figure 1).

End users can get access to a database, if they know what data exists and how to access the data via an artificial language. Although, users have an idea of what data exists, but may have difficulty in posing queries that can extract the required data. By using restricted forms of English it is easier to generate natural language querying systems with reasonably good performance as demonstrated by relational-like DBMS.

However, direct manipulation of data by using two-dimensional programming technology provides a novel approach to database creation and retrieval. The higher level language interface to the DBMS (in Figure 1) must

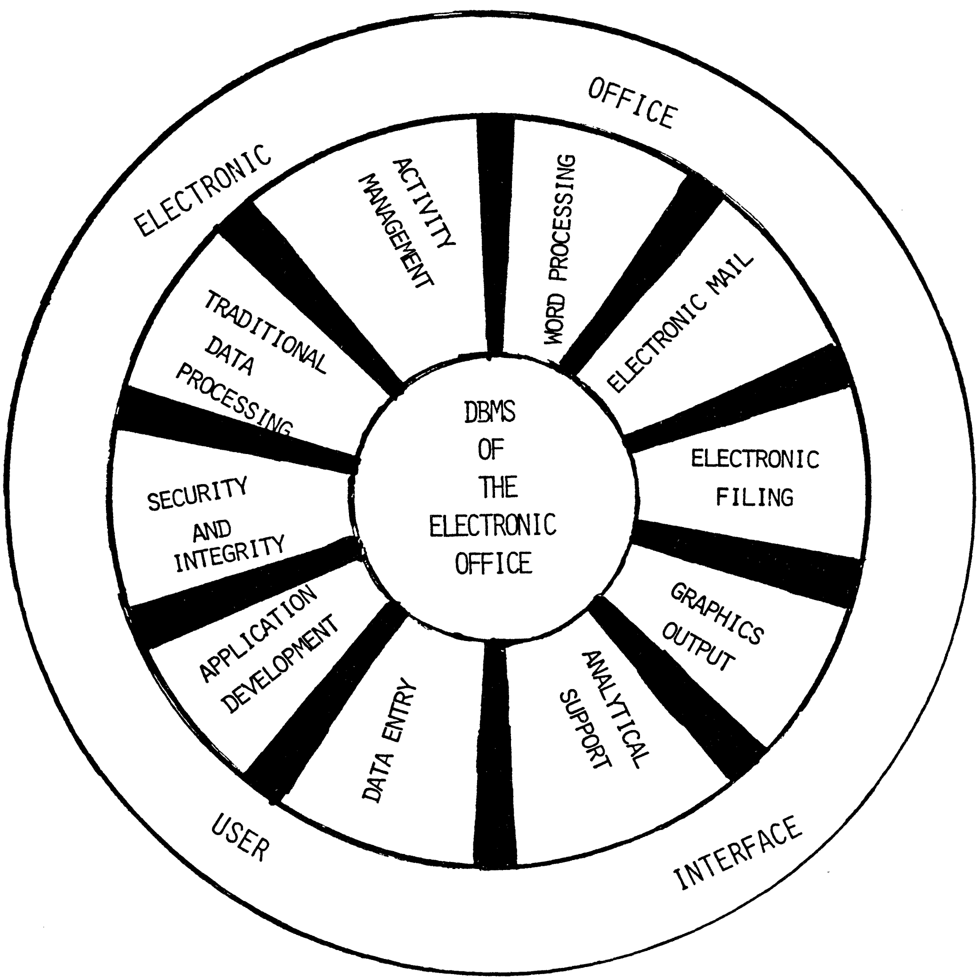


Figure 1:

A single unified office functions interface to the DBMS.

bring new flexibility to nonprogramming computer users (the typical user in the electronic office). One way of providing this is with skeletal programs which can be filled in with appropriate parameters. For instance, Office Procedures by Example (OBE) [3] with its two-dimensional programming approach can be used to compose, edit, format, and distribute letters, reports and graphs. OBE is an extended version of the Query-by-Example (QBE) data base language [4], implemented at IBM Thomas J. Watson Research Center.

Users use their office work stations to do word processing, data entry, electronic filing, electronic mail etc. Since a DBMS of the Electronic office has to manage all of the diverse types of office data, it is natural to use the DBMS as an integrator. A DBMS that supports these functions as an integrator is absolutely vital and must be hidden from view. A strong dose of human engineering is necessary to transform the current DBMS into a new generation of DBMS that will be responsive to the special requirements of office issues as detailed in the next Section.

Thus it is critical for an electronic corporation of the future to choose a 4th generation DBMS [5] that most suitably fulfils its office and database needs, and shows potential to evolve into a new generation DBMS. Such a selection of DBMS should not only meet the current needs but also facilitate all future data integration [6]. The ultimate challenge in integrated office systems is the capability to achieve data integration of the diverse office data.

3. OFFICE ISSUES VIS-A-VIS DBMS FUNCTIONS

Office issues are shown in Figure 1. In this section, we will investigate the implementation of these office requirements using the functional characteristics of the DBMS. In this discussion, each of the office issues considered is clearly defined; all features of the office issue are identified; and functional characteristics of the DBMS are mapped to implement the office issue appropriately.

For the sake of brevity, only three functions in an office system are considered. They are Activity Management, Electronic Messaging and mail system and Word or Document Processing.

3.1 ACTIVITY MANAGEMENT

Activity Management consists of the features required to provide each user with the ability to manage his/her functional responsibilities. Consequently, it must include an electronic desk calendar to schedule meetings, appointments and conferences. It incorporates both a "tickler" feature for creating lists of things to do and an automated bring forward facility to remind the user of important dates. Indeed, by the joint use of the tickler and bring forward features in conjunction, users will be able to create activity lists and completion dates to enable them to perform simple project monitoring and control functions. Project management facility in Activity Management, however, is not intended to represent a facility for sophisticated project management or to provide system generated impact analysis of slippages against project milestones or replanning facilities. Such sophisticated features should be incorporated in analytic support software part of the automated office.

The activity management function can be broken down to consist of following features:

> Electronic Calendar feature provides each user with a calendar extending user specified number of months into the future and retaining days specified in the past (to enable review of past activities);

- Activity Scheduling feature enables users to schedule both business and personal activities, with other users not be able to determine the type of activity scheduled unless specifically authorized. Thus, personal activities would simply show up as times when the user is unavailable upon inspection of the calendar;

- Calendar retrievals for Review permits users to review other users' calendar for "available" and "unavailable" times, at the option of the user inspecting the calendar;

- Multi-access support feature permits simultaneous access to a group of users' calendars for the purpose of scheduling conferences;

- Multiple updates to enable users to make tentative appointments in other user's calendars during available blocks of time. It is necessary to automatically notify the calendar's owner of all such tentative appointments on request and enable him/her to confirm or reschedule them automatically with the requesting party;

- Activity Coding feature permits activities to be coded and automatically logged for time monitoring and control purposes. Thus, automatic generation of time utilization reports for all such coded activities is possible;

- Project Monitoring and control feature permits a group or series of scheduled activities to be nested under a single user defined code. In effect, this permits creation of a "project activity list" and series of milestones for simple project control;

- Reporting feature provides automatically generated bring forward activity lists showing all activities due on a particular day. It is necessary to permit users to electronically check off completed activities or to add/delete activities at will;

- Scheduling Meetings feature automatically determines the earliest available time slot for two or more users of the system, and schedules tentative meetings at that time. It is a part of this feature to permit users to accept, reject or ignore other users' attempts to schedule meetings on their calendars;

- To-do lists feature provides an "electronic note pad" that can be used to record lists of things that must be done. It also enables dates to be logged against the activities listed on the note pad and automatically brings these activities forward to the user's attention; and

- Activity Diary to record unscheduled but completed activities.

Implementing activity management using a DBMS involves interactions between several components of the DBMS. In particular, query processing, retrieval, storage, concurrent access, security and reporting components play a major role in the development of all features considered. For instance, to schedule a meeting electronically, using a DBMS, results in following interactions.

- Query Processing issues the command to retrieve the earliest available time slot for the meeting of a set of people. If calendars for personnel are logically organized and stored by the name or other identifiers of personnel, this command is equivalent to issuing retrieval requests for calendars of each of the personnel involved;

- Searching of all calendars (of personnel involved) for a common earliest available time;

- Retrieval of the available time slots from all relevant files;

- Integrity & Consistency Checks performed during retrieval to ensure that no two meetings for a given person clash, no one involved is on vacation or away on business, no one invited has tendered his/her resignation or is about to be transferred;

- Security Features are enforced to maintain overall file security, control access and grant authorization to update the calendar file;

- Storage of a record in multiple calendars, to indicate that a meeting has been scheduled. Based on the logical organization considered this command would result in multiple storage of the same record in different calendars. The storage organization for this system must also include a method of indicating multiple locations of the same record, to facilitate faster access should such a record need to be deleted or updated;

- Reporting of a daily, weekly, or monthly calendar for the personnel involved. A tickler file and reminder system capability to prompt the user based on the system date, time and the presence of the record in the calendar will be desirable;

- Multiple views of the same calendar, where for instance, the manager may see the full calendar and the secretary will have the purpose of the meeting blanked out because it is "confidential".

3.2 WORD PROCESSING

Current State-of-the-art document processing involves using text editors. These text editors have their own facilities for storing and manipulating data. If documents can be organized like normal structured data, then all the facilities offered by a DBMS can be automatically used for document storing and manipulation. There are added advantages such as:

- It is easier to merge database data such as graphs, pictures, statistical data, tables and figures, into the documents within a single system;

- Sophisticated security features of access control, access authorization, encryption, system crash recovery are available to documents;

- Powerful capabilities to reorganize the text in comparison to those offered by word processors. For instance, at the revision stage, adding many new ideas, deleting many old ones, or drastically changing the logic of the presentation is achieved better with DBMS features.

Stonebraker et al. [7] have a proposal to enhance the relational database system INGRES to incorporate the features required for document processing. These features aid a relational database system to be substantially more useful in text processing applications by supporting ordered documents. The features include supporting variable length strings and ordered single and multidimensional relations. Ability to decompose and compose text fields is aided by new substring operators, a new break operator, and a generalized concatenate operator.

Xerox's Star [8] system for non computer knowledge workers has a user interface of "what-you-see-is-what-you-get" philosophy. Within this user model the Star system integrates traditional data processing using standard Star documents in relatively unstructured situations. Star's system provides inclusion of documents, database data, graphics, figures and tables in a single system. Star also provides the process to establish the correspondence between data in a document and records of a record file.

Star's query language facility called filtering is similar to Query-by-Example [4]. We have so far identified the new features required to allow a DBMS to perform document processing effectively.

Brent [9] has shown that a DBMS is more useful for technical writing that involves complex arguments covering a number of issues. Brent's approach involves identifying the stages of the writing process where a DBMS can be more effective. He identifies 3 stages in technical writing. The first stage is a rather long "Note Taking" stage in which basically ideas and references as they occur are recorded for future retrieval. A DBMS is very effective in this stage in saving time and improving quality.

In the second stage of outlining, keeping track of massive amounts of information on various topics and managing it quickly and efficiently while writing is achieved by using a DBMS. DBMS allows organization and reorganization of all notes with minimal effort, thus permitting many different approaches to the topic.

In the final stage of revising, to fit in a new theme or address new issues a major reorganization of the paper becomes necessary. This may involve deletion of entire sections or addition of others. To perform major reorganization of the text, a DBMS is more powerful than a word processing program. Brent identifies strategies for using a DBMS in the three stages outlined above.

In conclusion Brent [9] summarizes that "It would be particularly helpful to develop DBMS and word processing programs that are compatible with each other. In that way, the database created in the earlier stages by a DBMS could form the beginning of a text that would be modified and revised by a word processing program."

3.3 ELECTRONIC MAIL AND MESSAGING

The electronic mail/messaging in an automated office must permit individuals with workstations to create, send and receive mail or messages to one another within a network of specified users. In addition to permitting users to send or receive short, informal messages and memos, the electronic mail facility should allow for the transmittal of lengthier and more formal correspondence composed with the aid of the text processing feature. Finally, the network should support transmittal of completed copies of electronic screens that have been filled out using the Data Entry facility.

A typical electronic mail and messaging system has a central database which stores messages, and assigns a mailbox to each user accessible from any terminal that can log on.

To fulfill these functional requirements of electronic mail and messaging, the system generally consists of following features:

- Message Creation feature enables the user to create, send and receive mail and messages to any user or set of users on the network;

- Mailbox feature provides an electronic "in basket" where mail/messages can accumulate;

- Message Log provides a user-reviewable automated log of all messages sent by a user showing date, time, destination, priority and, if appropriate, topic. This feature also provides a user-reviewable log/listing of all accumulated messages showing date, time, sender, priority and, if appropriate topic;

- Confidential and Urgent mail feature provides for classifying mail as confidential or urgent. If the message is confidential, enforces security controls.

If the mail is urgent, sends appropriate notification to the recipient and enables the sender to determine whether or not the recipient has reviewed it;

- Distribution Lists creation feature enables the sender to transmit multiple copies. These lists can be permanent or temporary. Also provides system-established distribution lists that can be used by users. The feature permits use of aliases for senders/recipients;

- Message Manipulation feature enables messages in the in-basket to be reviewed in user determined order, printed, forwarded, deleted or remailed. It permits the recipient to reply either by appending text to the current message or composing a new message. The system automatically logs the source of the original message so that a reply can be made without reentering this information;

- User Status feature provides a directory of individuals on the network and a capability of indicating which of those individuals are currently "logged on";

- Text processing interface provides access to administrative support facilities.

For instance, implementing a message log feature for accumulated messages using a DBMS involves the following DBMS functions:

- Query processing issues a command to retrieve all accumulated messages for a given user;

- Retrieval of all relevant fields (date, time, sender, priority, topic) for accumulated messages for the given user from the database;

- Security Features are enforced to blank out the topic if classified confidential;

- Reporting of the automated log of messages showing appropriate fields in chronological order of arrival.

Most Office System vendors have designed their electronic mail filing system metaphorically similar to cabinets, drawers, folders and documents. Although, users are familiar with such a filing system, they are also aware of the time it takes for retrieval. Integrating electronic mail with a DBMS offers several advantages. They are:

- Any document or data can be retrieved in a consistent manner;

- DBMS provides the ability to manage vast numbers of messages and users;

- It can handle a variety of field sizes and numbers without a painful updating burden;

- DBMS provides better security for sensitive materials communicated through Electronic Mail;

- Messaging utilities such as the electronic janitor, system utility to generate accurate message system name (for multiple users with same last name) are better addressed through DBMS capabilities;

- An application built using the DBMS can be designed to deal with a wide variety of messages, flexibly, by providing a simple and uniform interface to the user. It can easily manage a large volume of messages.

Lowenthal [10] has assessed the impact of a database machine, the Intel-Database Processor (iDBP), on local area networks. Lowenthal has pointed out the importance of a concise and a high level interface to the DBMS to minimize network traffic. The relevance of a higher level of cooperation between a DBMS and an advanced communications systems is evident in the implementation of NDX-100 [11]. NDX-100 is a microprocessor-based filing system, that is also a part of the advanced communication system of the office of the future. A voice store and forward message system [12], can use a telephone or terminal and a voice store database for its implementation.

4. DISCUSSION AND STATE-OF-THE-ART

4.1 DISCUSSION

Figure 2 depicts a table of mapping of the functional features of a DBMS software, that play a role in the implementation of the electronic office issues. This mapping matrix is derivable through the analysis of all office issues as discussed in Sections 3.1, 3.2 and 3.3.

In order to take full advantage of the technology, it is essential to identify the interrelationships among the office issues. Such an analysis aids to implement an integrated office system, based on the premise that office automation is a process not a device or a product. To implement an integrated office system with a DBMS, it is necessary to understand the System data flows that show linkages between system functions. Thus, it is difficult to design a fully integrated service, without understanding data flows and process interrelationships. Figure 3 shows data flows between various subfunctions of the electronic office.

Based on the discussion in Sections 3.1, 3.2, 3.3 and an analysis of Figures 2 and 3 a partial list of requirements for the new generation of DBMS can be defined. The list is neither complete nor exhaustive but provides an insight into what is required in a 4th generation DBMS so that it can integrate office functions effectively. The requirements are:

OFFICE ISSUES ↓ / DBMS FUNCTIONS →	SEARCHING	SORTING	STORAGE	RETRIEVAL	REPORTING AND FORMATTING	MULTIPLE VIEWS	QUERY PROCESSING	SECURITY FEATURES	INTEGRITY AND CONSISTENCY CHECKS	DATA ENTRY
ACTIVITY MANAGEMENT	×		×	×	×	×	×	×	×	
ELECTRONIC MAIL	×	×	×	×	×		×	×		×
WORD PROCESSING	×	×	×	×	×			×		×
ANALYTIC SUPPORT	×	×		×			×			
GRAPHICAL SUPPORT	×	×	×	×	×	×			×	
ELECTRONIC FILING	×	×	×	×	×	×	×	×	×	×
TRADITIONAL DP	×	×	×	×	×	×	×	×	×	×
APPLICATION DEVELOPMENT			×	×	×		×	×	×	
TELEPHONY SUPPORT	×	×	×	×				×	×	×

Figure 2:

Functional features of DBMS software vis-a-vis Electronic Office Issues.

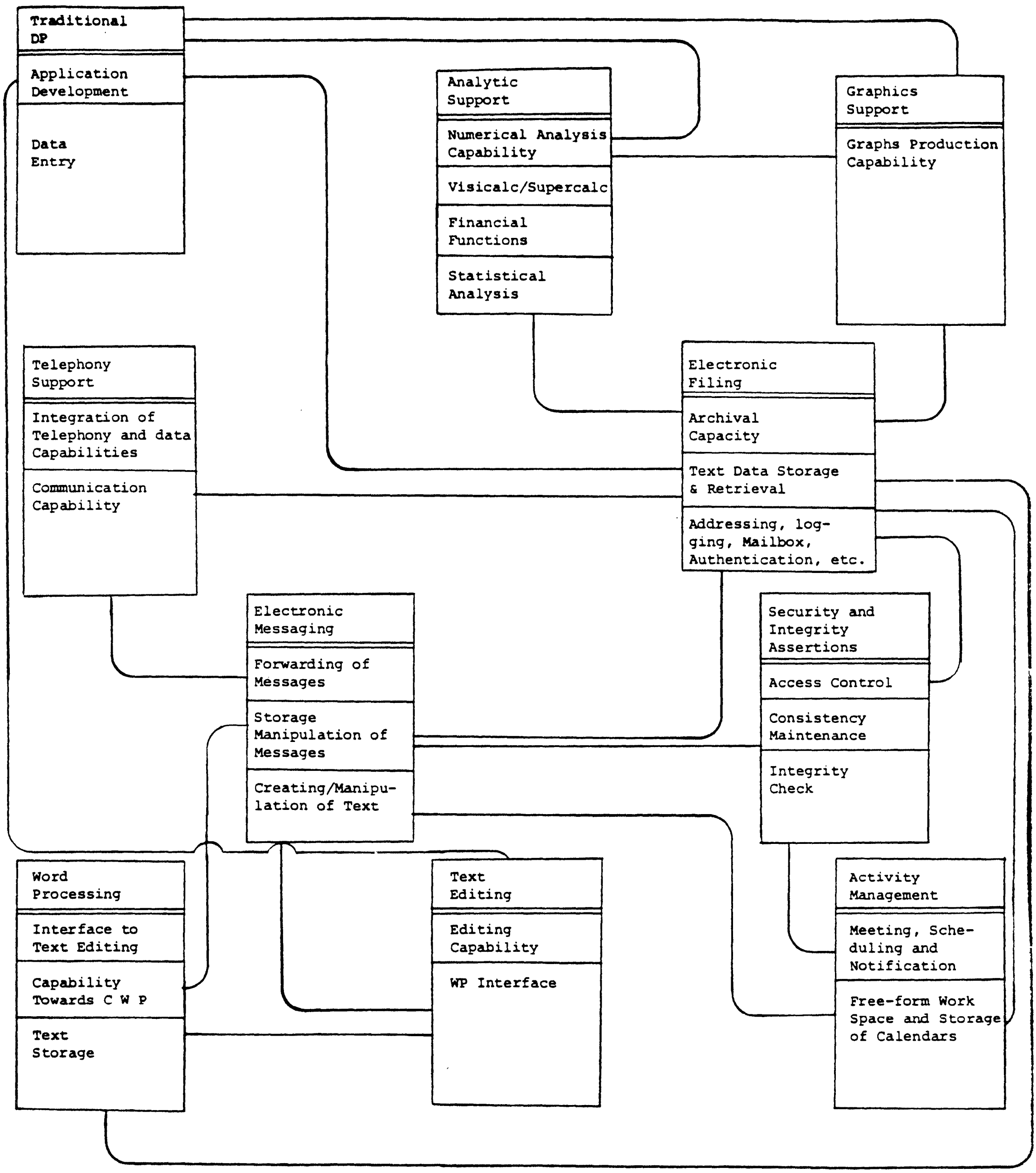


Figure 3:

System Data Flows and Interactions

- Powerful reading capability to browse through longer documents and simple databases is necessary.

- A DBMS allows structured files to be created, updated and deleted easily. The capability to file and retrieve commonly used information such as notes, drafts, to-do lists, addresses, telephone numbers and appointments is vital.

- Easy access to large corporate and public databases through a single, unified access language and user interface is essential.

- Easy interface to electronic message systems to permit exchange of notes, documents, business forms, and information is required.

- Word Processing interface to allow text entry and cataloguing by the user to enter and manage documents of any size is necessary.

- Interface to common calculation tools such as spread sheet calculator, together with decision support system software for project management, financial analysis, statistical analysis and job estimation is useful.

- Tickler file and reminder system capability to prompt the user is desirable.

4.2 OFFICE SYSTEMS CONSISTENCY AND INTEGRITY

The very nature of office systems, indicates that the redundant storage of data at multiple sites is a definite possibility for large corporations of the future. The problem of minimization of the cost of updating this information at all relevant nodes, is as yet unsolved. Updating from several sites concurrently can result in inconsistent copies. A thorough understanding of consistency problems involved are discussed in [13]. A project [14] for the ARPANET community has implemented a duplicate database facility that utilizes different update protocols for different classes of update transactions. When multiple processes need to synchronize, the problems of deadlock [15], and cyclic restart result in violation of interprocess consistency. Research on other aspects such as Integrity constraints on office forms [16] and numerous areas of forms, message files/schemes/systems, etc. [17] is underway. Almost all research efforts have been based on known principles of the database systems and have originated from the database community.

5 CONCLUSIONS

A major objective of office automation is to improve the decision-making process by producing accurate and timely information. With the diverse nature of the office data and its interrelationships it is a formidable task to produce timely information. The ultimate challenge in building integrated office systems is to achieve data integration of the diverse office data. A 4th generation DBMS makes it feasible to access integrated data that crosses operational, functional and organizational boundaries within an enterprise. To be responsive to the special requirements of the office environment a modern DBMS must significantly differ from its traditional data processing ancestors. In this paper, a partial and non exhaustive list of special requirements for modern DBMS that can integrate office functions is identified.

As the automated office of the future becomes a reality of today, the need for powerful underlying database tools and techniques will become increasingly obvious. However, adaptation of the DBMS technology to the merging office, needs to be better understood. Database technologies [18, 19] have matured; they can now form the building blocks of the electronic office. The emergence of the information centre concept [20] and the near future availability of expert systems, further enhances the concept of the 4th generation DBMS as the integrator of office systems.

REFERENCES:

[1] C.A. Ellis and G.J. Nutt, "Office Information Systems and Computer Science", ACM Computing Surveys, Vol. 12, No. 1, March 1980, pp. 27-60.

[2] K.S. Barley and J.R. Driscoll, "A Survey of Data-Base Management Systems for Microcomputers", Byte, November 1981, pp. 208-234.

[3] M.M. Zloof, "QBE/OBE: A Language for Office and Business Automation", IEEE Computer, Vol. 14, No. 5, May 1981, pp. 13-22.

[4] M.M. Zloof, "Query-by-Example: A Data Base Language", IBM Systems Journal, Vol. 10, No. 4, 1977, pp., 324-343.

[5] E. Miklovich and S.S. Isloor, "Selecting the Right DBMS for Your Environment", Proc. CIPS Session '82, Saskatoon, Canada, May 1982, pp. 87-94.

[6] S.S. Isloor and J.E. Drawbridge, "Managing Applications Development through Data Integration", Proc. CIPS Session '83, Ottawa, Canada, May 1983, pp. 13-23.

[7] M.R. Stonebraker et al. "Document Processing in a Relational Database System", ACM TOOIS, Vol. 1, No. 2, April 1983, pp. 143-158.

[8] R. Purvy et al. "The Design of Star's Records Processing: Data Processing for the Noncomputer Professional", ACM TOOIS, Vol. 1, No. 1, Jan. 1983, pp. 3-24.

[9] E.E. Brent, "Writing with a Data-Base Management System", Byte, November 1981, pp. 18-33.

[10] E. Lowenthal, "Database Systems for Local Nets", Datamation, August 1982, pp. 96-106.

[11] J. Slonim, et al., "NDX-100: An Electronic Filing Machine for the Office of the Future", IEEE Computer, Vol. 14, No. 5, May 1981, pp. 24-36.

[12] J.D. Gould and S.J. Boies, "Human Factors Challenges in Creating a Principal Support Office System - The Speech Filing System Approach", ACM TOOIS, Vol. 1, Nol. 4, October 1983, pp. 273-298.

[13] S.S. Isloor, "Consistency Aspects of Distributed Databases", Ph.D. Thesis, The University of Alberta, Edmonton, August 1979.

[14] P.A. Bernstein, et al., "Concurrency Control in a System for Distributed Databases (SDD-1)", ACM TODS, Vol. 5, No. 1, March 1980, pp. 18-51.

[15] S.S. Isloor and T.A. Marsland, "The Deadlock Problem: An Overview", IEEE Computer, Vol. 13, No. 9, Sept, 1980, pp. 58-78.

[16] J.C. Ferrans, "SEDL-A Language for Specifying Integrity Constraints on Office Forms", Proc. SIGOA Conf. on Office Information Systems, Philadelphia, U.S.A., June 1982, pp. 123-130.

[17] D. Tsichritzis, (Ed.) Beta Gramma, Technical Report CSRG-150, Computer Systems Research Group, University of Toronto, Toronto, Ontario, June 1983. (A collection of eighteen office automation research papers).

[18] D.K. Hsiao, (Ed), "Special Issue on Database Machines", IEEE Computer, Vol. 12, No. 3, March 1979, pp. 7-79.

[19] F.J. Maryanski, "Backend Database Systems", ACM Computing Surveys, Vol. 12, No. 1, March 1980, pp. 1-25.

[20] A. Benjamin, "Information Centre: Problem Healer Or Symptom Remover", Computer Data, October 1982.

Biography

Dr. Sreekaanth S. Isloor
is a Senior Consultant with the Consulting Services Division of Cognos Incorporated. He has provided management consulting to Canadian federal government departments, Crown Corporations and private industries in the areas of Databases and 4th generation languages, Information Resource Management, Computer Communications and Office Automation.

Prior to joining Cognos Inc., he worked for Bell-Northern Research (a division of Northern Telecom), Systemhouse Ltd. and the University of Alberta, Edmonton, Canada.

Dr. Isloor holds a Doctor of Philosophy in Computer Science from the University of Alberta, Canada. Dr. Isloor has given university-level instruction in Canada and India. In addition to a number of presentations to Senior government officials in Canada, Dr. Isloor has been an invited speaker at workshops/symposia on industry trends and emerging information technologies. Dr. Isloor has authored a number of technical publications and presented several conference papers in Canada, United States, India and Europe.

SYNCHRONIZATION AND RECOVERY OF A DISTRIBUTED DATA-BASE
AN IMPLEMENTED EXAMPLE ON HP/3000 COMPUTER SYSTEMS

Ewald Maria Mund
Mund-Software

Buchholzstr. 13
3066 Stettlen/Bern
Switzerland

Summary
In this paper we show that with the help of software design methods like the Entity-Relationship Model and Petri-Nets togather with programming- and communication tools, offered by a HP/3000 computer systems, it is possible to implement true, but in a certain way limited, distributed data-bases today.

I. The Application

For DU PONT de Nemours Europe in Geneva I got the challenging task to develop a computer supported Price Administration System. Without going to much into details, the system maintains price-lists which are created in several European subsidiaries. The price-lists get copied to Geneva for acknowledgement, sometimes modified there and afterwards sent back to one ore more subsidiaries. In general, there is a copy of every price-list in at least one subsidiary and in the central in Geneva. That fact makes the Price-Administration system to a true distributed data-base.

I.1 Simplification and Generalization of the Application

a) As already mentioned there is always a total copy of a price-list in the central as well as in at least one European subsidiary.

b) Parallel global acces to the same price-list(s) do not happen and are not implemented. That represents a important simplification of the application, since we do not have to implement a global, fullsynchronized access to distributed data and so we do not have to solve the problem of global locking and global dead-lock recognition and prevention. In general we have a distributed data-base system with recovery but with following limitations:

   1. Every entity (price-list) of the information system has a total copy in the central-system as well as in at least one slave-system in the subsidiary. Slave- and central systems are connected via DS/3000.
   2. Transactions on one or more entities of the distributed information system can only be done at one, but per entity different computer system. Parallel access to the same entity stored at more than one computer system in the network are not needed for the applications. Examples: All information systems that do not belong to the class of reservation (booking) systems like distributed electronic mail, distributed inquiry-systems, interactive videotext systems?.
   3. The synchronization, that means the process of distributing those objects modified by one transaction, have not to be done online (< 5 sec). A timeperiod of 30 minutes or even more is acceptable.

II. Desirable Solution.

For generalized applications, described above, I developed a methology for the synchronization which is independant of the application itself. That methology makes a clear distinction between the application software and synchronization software. That distinction allows us to run the application centralized on only one computer or distributed on two or more computer-systems, without any modification of the application software.
The result is that the application programmer pays less attention for the synchronization software and v.v. the synchronization software is absolutely application independant.

III. Hardware and Software

IMAGE/3000, DS/3000 and RAPID/3000 are the software tools to implement our distributed data base running (now) under MPE V. Because TRANSACT/3000 works on top of DS/3000, the synchronization software has been very fast implemented. To make it clear again: a synchronised informations system in our terms is a distributed information system, where every modification, done to local entities, is also done to every copy of those entities on every computer system in the network within a certain time period (< 30 minutes or more).

III.1 Topology of the Data-Transport-Network

DU PONT's European price administration system is running on eight different HP/3000 computers, installed in seven different European countries. The HP/3000 systems are connected togather via DU-NET, DU PONT's multiple protocol supporting network. One of those supported protocols is X.25 and therefore all HP/3000 systems use DU-NET as a private X.25 data-transporting network.

IV. Developing Models and Methods

Static Model.

The Entity-Relationship Model /1/ was used to construct the price information system. Every entity (in our case a price-list is one of the entities), used in our information system gets a unique entity-number when it is created. The creation of that unique entity-number is the job of the application software. The structure of the entity-number contains two indications, the type of the entity and the subsidiary, where the entity has been created. The entity-number is not always visible for the user. He sometimes accesses the entity via its logical key. In that case the entity-number is a "Alternative Key" for that entity. A entity itself consists of one or more records in one or more IMAGE/3000 data-sets. Every data-set is normaly in Third-Normal-Form according to the Relational-Data-Base design methods. Every transaction itself is a entity; that makes it is easy to distribute and synchronize transactions. Every transaction gets a unique transaction-number. The creation of the entity "Transaction" is job of the application software. The contents of that entity are mainly the entity-numbers of those entities, that are modified by that transaction. In addition the entity "Transaction" contains a so called "Destination Table". The destination table shows in which subsidiary the transaction was originally executed.

Dynamic Model

Petri-Nets /2/ represent a ideal method to make dynamic and parallel processes visible and understandable. In our case we have to copy all the entities, modified by one transaction, to the according systems. That is a a dynamic and parallel process. A transaction is totally synchonized, when all entities, modified by that transaction, are copied to every node including the transaction itself as a entity.

V. Synchronization

Whenever entities have been modified by a transaction, the entities togather with the transaction as an additional entity have to be copied to other systems. The creation of a entity itself is a transaction. The synchronization of the transaction, that means the distribution of the modified objects to the central- and the slave-systems, can be controlled by four "Transaction-States".

V.1 Transaction States

State "Local"

A transaction is in that state, if it has not (yet) to be synchronized. An example is the creation of a price-list.

State "Pending"

A transaction and its modified entities is in that state, when it waits for the transport to the central-system. The destination table shows to which subsidiaries the transaction and its associated entities have to be copied in addition to the central-system. The application software is responsible for the contents of that table. It also sets the transaction-state to "Pending" when the transaction is finished. That is the only interface between the application- and the synchronization software!
Whenever a transaction is set to the state "Pending", a so called "Transporter" process is activated, which copies the according entities togather with the transaction to the central-system. When that transport is finished, the transaction is set to the transaction-state "Central". The transport represents a normal, multiple-entity-transaction for the data-base in the central-system. Because the standard IMAGE/3000 Logging and Recovery facility (for single, non distributed data-bases) is operator-dependant, every transaction is first done on a Shadow-data-base. After the transaction is totally finished, the entities involved in one transaction are copied to the original data base. The Shadow-data-base concepts works like a "Backout-facility". It quaranties that every data-base on every system is always in a logical consistant state. (ILR quaranties physical consistancy). So we do not need the IMAGE/3000 Logging and Recovery facility, but we still keep the logical consistancy of every data-base on every computer in the network without any human intervention by an operator or other person.
When a transaction switches from the state "Pending" to the state "Central", the transaction gets a unique timestamp. The timestamp is the time of the central-system when the transport of the transaction to the central-systems has started. The use of that timestamp will be explained later togather with the recovery process. Transactions which have been executed in the central-system, are also set to the transaction-state "Pending". A so called "Mini-Transporter" process switches the state of that transaction directly in the state "Central" without any copying activity. But the "Mini-Transporter" also sets a timestamp.

State "Central"

A transaction is in the state "Central" when it has been copied to the central-system, but it is still not synchronized. Not synchronized means there are still subsidiaries, which do not have a (total) copy of the modified entities of a transaction. The destination table indicates to what subsidiaries the entities have to be copied. When a transaction is in the state "Central" , "Synchronizer" processes are started, one per a slave-system . The "Synchronizer" looks in the destination-table of the transaction to find out if and what entity it has to copy to the slave-system. After having finished the copying process, every "Synchronizer" sets a stateflag in the destination table to indicate that the expected work is done. When all "Synchronizers" have finished their task, the transaction switches to the final transaction-state "Distributed". Especially in that case Petri-Nets are a phantastic model to explain and implement dynamic and parallel processes. (See figure 1)

State "Distributed"

A transaction is in that state, when the transaction as a entity and all the entities, modified by that transaction, are copied to all subsidiaries according to the destination-table. Not only in the subsidiaries but also in the central-system the copies of the transaction are in the state "Distributed". During that process it is possible that the copy of the transaction in a subsidiary is already in the state "Distributed", but in the central-system the copy of the transaction is still in the state "Central". That causes no problem as we see later.

V.2 State Transitions

A state transition or state switch of a transaction is caused by the application software (Local to Pending) or by the "Transporter" process (Pending to Central) or the "Synchronizer" process (Central to Distributed). The activation of the processes (Transporter, Synchronizer) can be achieved in two different ways:

1. active: The processes themselves ask after a certain timeperiod, if a transaction has switched into a specific state.
   Advantage: Simple to implement, no special recovery procedure needed
   Disadvantage: A smaller timeperiod increases system resources; activation of processes without orders is possible.

2. passive: Application software and the above processes activate other processes via Message Files.
   Advantage: Processes are immediately activated. The result is a short period for transaction distribution.
   Disadvantage: After a system- or program abort a special written recovery-software has to recreate and refill the Message-files.

Remark 1:
Whenever a transaction is copied from the subsidiary to the central-system or from the central-system to the subsidiaries, the state transition is first done in the receiving data-base and afterwards in the sending data-base. If you hurt that rule, a correct recovery process is not possible.

Remark 2:
The transporter- and synchronzer software are MPE processes, that execute remote data-base access to the subsidiaries. That allows an easy analysis and trigger ing of the recovery software.

Remark 3:
Whenever a entity, modified by one transaction, has to be copied to the central- or to a slave-system, the whole entity with all its data-sets is copied, even if only one single item of the entity was modified by the transaction. That allows us to write the synchronization/recovery-software independant of the application transactions, but on account of performance.

VI. Recovery

To manage the recovery process after an abort of one or more computer-systems, including the central one, we need three different data-base-states. To make it again more clear, the shadow-data-base concept (or the IMAGE/3000 Logging and Recovery feature) in addition of ILR (Intrinsic Level Recovery) quaranties, that every data-base is in a logical state after the restart of the system. But it is possible that for a lot of reasons (DISC CACHING) the data-bases are not synchronized.

VI. States of the Data-Bases.

State "Not Synchron"

Whenever a computer-system in the central in Geneva or in a subsidiary is started, a special "Starter" process is activated (e.g. by Operator UDC) to set the data-base into the state "Not Synchron". That means we do not know at the moment, in which state is our distributed data-base. No user is allowed to modify the data-base when the data-base is in that state.

State "Semi Synchron"

Will be explained later togather with the different recovery cases.

State "Synchron"

A data-base in the central- or in every slave-system is in that state if we can assume that every other data-base is also in that state. That is the normal state of the data-base, and application users are allowed to modify the database only in that state.

VI.2 Analysis of Recovery Cases

Case 1: Line Down

Transporter and synchronizer recognize line downs (IMAGE/3000 Remote Data Base Access error 101 to 110). Because the shadow data-base concept works like a Back-out no incomplete transaction and entity are neither in the central-system in Geneva nor in the slave-systems in the subsidiaries. Every data-base is in a logical consistant state, but perhaps not synchronized. That can be ignored, because normally the line is up after a short time-period. If the time is not up within a non critical timeperiod, the systems can be stopped until the lines are up again.

Case 2: One Slave-System Down

That case is recognized and correctly analyzed by the "Transporter-" or "Synchronizer" processes. They start than a so called "Mini-Recoverer" process. The "Mini-Recoverer" process waits in the central-system until a connection is made via Remote Data-Base-Access to the data-base in the subsidiary. The data-base in the slave-system is now in the data-base-state "Not Synchronized". The "Mini-Recoverer" searches with the help of transaction-timestamps for the youngest transactions in the central-system and in the previously down slave-system. Perhaps one or more transactions have now to be recopied to the slave-systems. When that task is finished, the "Mini-Recoverer" resets the data-base in the subsidiary to the state "Synchronized" and also reactivates the "Synchronizer/Transporter" process in the central-system. During the time period the "Mini-Recoverer" is working, the users in Geneva as well as in all other up susbsidiaries can work without any problem.

Case 3: Central-System Down

When the central-system is down, all the application users in the other subsidiaries can still work, because the state of their data-base is still "Synchronized".
When the central-system is up again, the previously mentioned "Starter" process sets the central data-base into the state "Not Synchronized" and activates per subsidiary a "Retransporter" Process. The "Retransporter" processes compare per subsidiary the transaction-timestamps and copy, if needed, missing transactions from the subsidiaries to the central-system. Only transactions in the state "Distributed" are used for comparisons. When a "Retransporter" has finished its activity,it sets a statemark in the destination table. When all "Retransporter" processes have set their statemarks, the state of the data-base in the central is switched into the state "Semi Synchronized". All transactions are now again back in the central-system and in the state "Synchronized". Because it is possible that at the same time one or more slave-systems could also have a system abort, the central data-base is not immediatley free for the application users. Now let's analyse the most complex recovery case.

Case 4: Central-System and one ore mor Slave-Systems Down

When all "Retransporter" processes have finished, so called "Resynchronizer" processes are started, one per slave-system. Every "Resynchronizer" process compares the youngest transaction-timestamps in the central-system with the youngest transaction-timestamp of the slave-system and copies missing transactions and the entities, modified by that transaction, from the central-system to the slave-system. Having finished, the "Resynchronizer" process sets the data-base of the slave-system to the state "Synchronized", but only a statemark in the central data-base. When all "Resynchronizer" processes have set their statemarks, the central data-base switches from the state "Semi Synchronized" to the state "Synchronized". Now the central-system is free for user access also.

VII. Future Aspects

To minimize the software for similar applications in future, a standardized procedure is possible. Under that procedure the structure of transactions and entities are stored in a dictionary. Application- and synchronization-software use a standardized distribution protocol. That protocol follows the methods described above. A next step would be to integrate such a protocol into the IMAGE/3000 data-base administration system!?

As a side effect, you can use above methods to mirrow a single data-base on a second system. After a system-failure the user can switch immediatley to the mirrow data-base. Some last transactions may not be copied to the mirrow data-base, but it is at least in a logical consistant state.

VIII. Conclusions

1. Without conceptual models and methods you can not implement complex applications like a synchronized distributed-data-base.

2. Without a (homogen) computer-network software like DS/3000 you can not implement a synchronized distributed-data-base under reasonable economical conditions.

3. A FLEXIBLE! programming language of the 4. generation (TRANSACT/3000) can drastically reduce software costs.
   The synchronisation software for ten different entities of the price-administration system was done in SIX man-month!)

4. Rome was not built in one day!. Although the described method does not allow global access and global locking of distributed data, the method shows that a step-by-step implementation is possible and pays for itself.

References.

/1/ Flavin, M; Fundamental Concepts of Information Modelling.
A Yourdon Press Monograph

/2/ Goos, G; (In German) Programmkonstruktionen
Fakultaet fuer Informatik, Uni Karlsruhe
Vorlesungsskript 1975-76
Kapitel 7.6 Petri-Netze

About the Author

Ewald Maria Mund has studied computer science at the Technical University of Karlsruhe and finished with a diploma. After 5 years working as a systems-engineer on the HP/3000 in Germany and Switzerland and two years as chief-engineer for a swiss-software-house, he is know a independant consultant. His main working areas are information-systems, communications and decision-supporting end-user-systems.

FIG. 1.
SUB-PETRI-NET FOR SYNCHRONIZATION

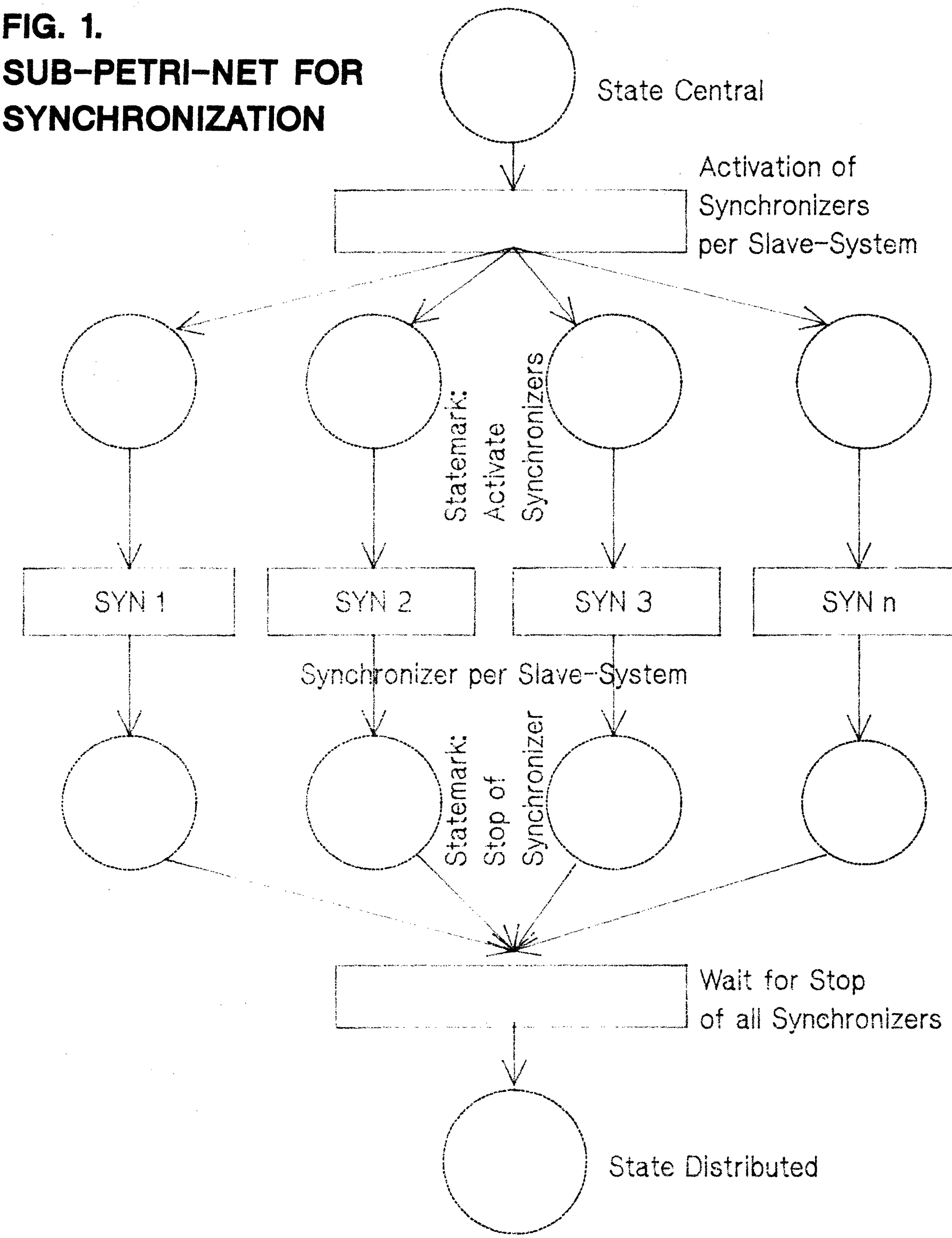


IMAGE Reflections

F. Alfredo Rego

Adager

Apartado 248
Antigua
Guatemala

A creative photographer takes an aspect of reality and, through a clever combination of shadows and depth of field, focuses our attention on relevant features and minimizes clutter. The result is a prize-winning image which reflects reality in an exciting way.

A creative database designer goes a step further: The result is a prize-winning IMAGE/3000 database which reflects reality in an exciting way, in image as well as in performance.

In this essay we reflect upon those qualities that distinguish prize-winning IMAGE/3000 databases and we give you practical advice which you can use, immediately, to create your own.

QUICK REVIEW

An IMAGE/3000 database consists of two complementary parts: a USER INTERFACE and a STRUCTURE.

The user interface helps authorized human beings perform two kinds of functions:

- addition, modification, deletion and reporting of INFORMATION stored by means of the database structure;

- general maintenance of the STRUCTURE itself.

The structure of a database is not strictly necessary, but it has a dramatic effect on the performance of certain functions, such as STORING, FINDING, RELATING, and REPORTING. Storing and finding are two sides of the same coin. First of all, we must store something before we can even hope to find it. Then, we still have to find that something among all the other stored things. Typically, the amount of effort required to find something is inversely proportional to the thought, structure, and effort expended in storing it. (To drive this point home, please close your eyes for a minute and think about your bedroom and your office!)

If we are dealing with trivial computer files that contain a few hundred records, we can afford to search through everything every time. If we have intelligent peripherals that can do these serial searches blazingly fast, we can get away with a few million records. But there is a limit to how far we can push the limit!

The whole idea of online database APPLICATIONS is to be able to relate appropriate data entries while somebody waits over the counter or over the telephone. The whole idea of online database DESIGN is to provide the fastest possible response time for the most important transactions and queries.

We may decide to relate when we enter the information, or when we report the information, or at any time in between, or during each of these stages. We may want to experiment with various combinations of resources, costs, and benefits. The choice is ours and, with IMAGE/3000, the question is one of CONVENIENCE: When and how are we willing to pay the price? If alternative ways suddenly become more attractive, Adager is always there to reconfigure things around.

IMAGE Checklist

Database management systems are just frail human attempts to organize unruly pieces of information.

For the sake of honor, though, database management systems should put up a fierce battle against entropy. Let us examine some tricks of the trade that we consider standard for such battles, as implemented by Hewlett-Packard in IMAGE/3000.

Approachability

IMAGE gives us contents-addressing methods (hashed and chained) as well as location-addressing methods (serial and directed). We can cleverly combine IMAGE's methods to build our own indexing methods.

IMAGE and the HP3000 operating system orchestrate concurrent approaches to one or more databases:

- By different processes within the same time frame. Some processes may be updating, some adding, some deleting, some reporting, some relating, some doing everything.

- By the same process throughout various time frames, even as the structures of the databases evolve, thanks to data-independence.

- By a mix of online and batch processes.

- By a mix of local and remote processes.

In all these modes, accessors are restricted by a security layout specified by the database administrator. For instance, in a payroll database some people may not be allowed to modify or browse the salary fields.

Flexibility

As circumstances change, we have to transform our database to keep up with our reality. IMAGE has powerful utilities supported by Hewlett-Packard, such as DBUTIL and Adager, that offer easy access to a wide range of database maintenance and tuning functions.

Performance and Cost-Effectiveness

"Performance" is a relative term. For instance, performance in the addition of new entries is totally meaningless if the performance of finding, relating and reporting those entries is pitiful. And a system with good performance in storage and retrieval may still be unacceptable if it requires human sacrifices just to keep it going!

Hewlett-Packard measures performance by results. We encourage you to measure the overall performance of IMAGE and to compare it to the overall performances of other database management systems. Your results will confirm this historical fact: IMAGE's price/performance ratio has held the world's record for over a decade!

Privileged Protection

In the good old days, every part of the computer was accessible to everybody. With the advent of operating systems and databases, some locations in main memory as well as in auxiliary storage were reserved for specialists and baptized as "privileged".

Since the internal tables of an operating system or a DBMS are overwhelmingly complex and sensitive, standard users should not access them directly. You would not let any old person operate on your brain or your heart, would you? Ditto for your operating systems and databases. Delicate stuff, indeed!

In the HP3000, software and hardware mechanisms make sure that users access privileged entities only by means of intermediaries approved by management. For IMAGE, the intermediaries supported by Hewlett-Packard are the IMAGE intrinsics (such as DBOPEN and DBPUT) and the IMAGE utilities (such as DBUTIL and Adager).

Reliability

Bells and whistles are fun but not sufficient. If a fancy database management system is not reliable, it is not worth using at all. Reliability means AVAILABILITY and, in the event of temporary unavailability, effective RECOVERABILITY.

The world's reliability record belongs to the HP3000 computer with IMAGE/3000, complemented by Adager's online recovery functions.

Ease of use

IMAGE's data structures are conceptually simple, well-defined and easy to understand. We can build all kinds of powerful and complex structures using IMAGE's components. We can easily access its intrinsics from BASIC, COBOL, FORTRAN, PASCAL, RPG, and SPL. We can build all kinds of friendly user interfaces by calling upon these fundamental procedures.

And if you are in a rush, you do not have to build anything yourself! There are hundreds of user interfaces to IMAGE available through the HP3000 contributed software library and through the worldwide network of applications suppliers. Just ask your Hewlett-Packard engineer for references or attend any of the many Users Group meetings. IMAGE's network of dedicated people is easy to use too!

Practical Advice

All right! So far, so good. But how do you go about modeling YOUR PARTICULAR reality? Let's see, using IMAGE/3000 as our frame of reference.

Define your Operational Entities and Relationships

Identify those aspects of reality that you want to model with the database. Have a party. Don't be bashful. Include everything that turns you on. Send all your consultants on a holiday. Dream and make notes of your dreams. Everything and anything is ok. Since you are REALLY INVOLVED in your business, you will naturally tend to focus on the OPERATIONAL aspects of your reality, since they are the most fun. The few CONTROL aspects that will manage to sneak in will be those that you consider absolutely necessary. ("Operational" aspects are FUNDAMENTAL. "Control" aspects are a pain in the neck, NECESSARY EVILS that would go away if everybody could just become 100% honest!)

Then, after the party is over, remember that a database can only reflect a small portion of reality in an effective way. Sober up and separate your big list into two lists: (1) a small list of those things that you MUST model and (2) a still-big list of those things that, though desirable, can wait until the next pass. Listen carefully to your consultants now (they are better at thinning down your wish list than at CREATING it). File away your big list of "desirables" for future reference and polish up your small list of "musts" for immediate use.

This list contains aspects of YOUR reality which you may classify in two convenient categories: ENTITIES and RELATIONSHIPS. You know exactly which are which. For instance, if you are a marriage counselor, you may think of MARRIAGES AS ENTITIES, since marriages are the focus of your professional life. But if you are a movie director, you may think of MARRIAGES AS RELATIONSHIPS, since people are the focus of your attention and the fact that some people happen to be married to other people is incidental.

Keep in Mind the Equivalence of Entities and Relationships

Entities and relationships are just two sides of the same coin. IMAGE/3000 uses the same concept of "data entry" to model either entities or relationships. For convenience (and performance), you may want to use MASTER datasets to model ENTITIES and DETAIL datasets to model RELATIONSHIPS. After all, the designers of IMAGE tried very hard to help you out and they spent many sleepless nigths optimizing these two kinds of datasets. But you can always change your mind to suit YOUR convenience! That is what creativity is all about!

Classify your Entities and your Relationships

Your entities and your relationships will conveniently fall into classes which are obvious to you. For instance, if you are a marriage counselor in an "average" town, you may find that you need two classes of entities ("males" and "females") and one class of one-to-one relationship ("marriage"). However, if you practice in a "rotten" city, you will probably ignore the gender of your entities and lump them into one class ("people") with many classes of complex many-to-many relationships among themselves!

Notice that an entity may be related to zero, one, or more entities of its own class or of different classes. Simultaneously or sequentially through time. For example, a Mormon in the 19th century could have many wives simultaneously, or one wife after another in the case of a widower whose wives just kept dying, or zero wives in the case of a bachelor. And the same Mormon could be related to many other entities that did not belong to the "people" class. For instance, in addition to being a husband, a father, a son, a brother, and so on, he could also own land, or write books, or plant trees, or hunt deer, or mine gold, or work full time in one job and part time in a dozen other jobs, or be the president of a large company. Or all of the above!

These apparently complex relationships are a delight to reflect with IMAGE/3000, thanks to its conceptual clarity and modeling power. Ironically, if you look at a standard database book, you will see these kinds of relationships used as examples to show what is wrong with standard database management systems. IMAGE users of the world: Count your blessings!

Model your Entities and Relationships

For convenience, use MASTER datasets as repositories of entities, DETAIL datasets as repositories of relationships, and PATHS as performance boosters to improve response time.

For example, if you are interested in keeping track of various relationships among people, you may want to define the following datasets.

MASTER dataset "PEOPLE", with this data entry:

- ID (key)
- name
- sex
- birthdate
- deathdate.

Since "ID" is probably some kind of UNIQUE identifier, you may want to choose it as a convenient key.

If you are interested in accessing people by more "human" attributes (such as their names or nicknames), you can certainly use "ID" as your "internal" key and many other fields (or parts of fields) as your "external" keys. The internal key just happens to be convenient for the sake of IMAGE's hashing algorithm (which converts a data value to an address). But you can design ANY mathematical mapping of your choosing that will convert any data value into a reference to whatever key you defined for IMAGE.

For instance, I like to use an indexing scheme that I developed with Ross Scroggs back in early 1981 during one of his visits to Guatemala. We used soundex-like algorithms to build powerful structures which allow us very quick answers to questions like "Give me all the Fred's who live in New York City and have a pre-release version of Adager's ItemChng". The fun part of our indexing is that we can get the same answer even if we pose a question which uses the equivalent attributes: "Give me all the Freddies who live in The Big Apple and have product JC810312".

So much for indexing! Back to the "People" data entry. If you are interested in other attributes, you may certainly add them: profession, salary, height, weight, religion, and so on. Whatever makes you happy, since it is YOUR database!

MASTER dataset "PROFESSIONS", with this data entry:

- profession code (key)
- name of the profession
- name of professional association
- address of professional association
- is certification required to practice this profession?

The comments regarding indexing and additional attributes are as relevant to the "professions" master dataset as they were to the "people" master dataset. And you can keep on going, adding master datasets as repositories of entities such as "companies", "clubs", "universities", "teams", or whatever.

Now that you have your entities, you may define all kinds of relationships by means of detail datasets like the following.

DETAIL dataset "MARRIAGE", with this data entry:

- husband's ID (path to people)
- wife's ID (path to people)
- marriage date
- dissolution date
- reason for dissolution
- number of children
- best man
- maiden of honor.

You may want to link datasets "marriage" and "people" with two paths to improve the performance of your queries: one path through "husband's ID" and one path through "wife's ID". If you have millions of data entries in your people and marriage datasets, these paths will allow you to find (in milliseconds) the marriage history and current status of the two individuals sitting across your desk at this moment!

You may want to define other paths for the following detail datasets if you are interested in seeing the whole spectrum of relationships in a snap.

DETAIL dataset "ENGAGEMENT", with this data entry:

- fiance(e)#1 ID (path to people)
- fiancee(#)2 ID (path to people)
- engagement date
- proposed marriage date.

DETAIL dataset "APPRENTICESHIP", with this data entry:

- teacher ID (path to people)
- student ID (path to people)
- profession code (path to professions)
- beginning date
- end date
- grade.

And so on and so forth!

As you specify your masters, your details, and your paths, keep in mind that the important question is: "Can you define, redefine or cancel these entities and their relationships at any time during the life of the database?" For performance reasons, you may want to wire some OBVIOUS relationships "hot" in the database's structure by means of PATHS, such as the relationships involving the people ID field in these examples. But you do not want to be stuck for life, since some hot relationships may cool off and some sleepers may wake up unexpectedly!

Orchestrate your Transactions

Specify the transactions that will allow you to add, modify, delete, and report these entities and their relationships. Decide whether or not some of these transactions need to be undisturbed by other concurrent transactions. Take advantage of IMAGE's locking to make sure that you achieve a fair compromise between "privacy" and "sharing".

Perform your Transactions

At your convenience, add, delete, find, modify, relate, and report entries. Do it solo or invite all your friends and fellow workers. IMAGE/3000 is a multitasking system, after all!

Tune up your Performance

Fine tune things in such a way that you reach a reasonable compromise between the RESPONSE TIME for any of these functions and the global THROUGHPUT for the whole transaction load. You may want to get some help from performance specialists. And you may want to implement indexing techniques to improve the performance of queries which are unique to your application.

What are you going to do for an encore?

Whenever things are going well, go back to the wish list that you squirreled away! Review the list. Throw away those entities and relationships that are no longer relevant. Add new aspects of your reality that are exciting to you at this point in time. Use IMAGE and Adager to adapt your database. Bravo!

What now: are you a Photographer or a Performer?

The answer to this question does not really matter (since, in reality, you are a database designer!) Best of luck in whatever you do. Have fun!

Biography

F. Alfredo Rego is Adager's Research & Development Manager. He has been working with Hewlett-Packard instruments since 1966, when he was a Physics research assistant in the Center for Nuclear Studies at The University of Texas. In the 1970's he worked as a university professor in Guatemala, teaching courses in theoretical mathematics, physics and computer science. He has worked exclusively with IMAGE and Adager since 1978. The HP3000 International Users Group honored him with the 1980 Hall of Fame Award, which reads: "Outstanding Contributor, for exemplary service to the Group and its membership".

THE DESIGN AND IMPLEMENTATION OF CHINESE DATA BASE MANAGEMENT SYSTEM

C.C. Yang, Shyh-Ming Chien and Jen-Shyh Lai
National Taiwan Institute of Technology,
Telecommunication Training Institute, Taiwan, ROC

Summary

In the advanced information society, more convenient and more diversified information management system should be provided to the user at a reasonable cost. The reletional data base management system, especially QBE(Query By Example)-like system may be one of the best choices. Since it can be used easily.

The aim of this paper is to design and implement a Chinese Data Base Management System(CDBMS) for Chinese non-professional computer users. It can provide multiple functions, including storing, retrieving, generation and management of data. The CDBMS based on QBE-like which is established on the HP-3000 computer systems. The system adopts HP's KSAM (Keyed Sequential Access Method) as the internal file management tool and FORTRAN as implementing program language.

Biography

Dr. C.C. Yang
is a associate professor with Department of Electronic Engineering and Technology, National Taiwan Institute of Technology, ROC. He received his Ph. D. in Computer Science from the Northwestern University, ILL., USA, 1980. His specilities include Software Engineering, Data bases, Computer Network, and recently, Software Test Automation.

Shyh-Ming Chien
is a lecturer of Computer Science Department, at the Telecommunication Training Institute, Taiwan, ROC. He received his Master of Science degree from National Taiwan Institute of Technology, 1983. Experience of Information Systems for several years. His current interests include computer network, Data Base Systems, and Digital Switching Systems.

Jen-Shyh Lai
is a instructor of Computer Science Department, at the Telecommunication Training Institute, Taiwan, ROC. He received his M.S. degree from State University of New York at Binghamton in 1981. His current interests includes Data Base Systems, Programming Language, Algorithms.

THE DESIGN AND IMPLEMENTATION FOR CHINESE DATA BASE MANAGEMENT SYSTEM

C.C. YANG

Department of Electronic Engineering
and Technology
National Taiwan Institute of Technology
Taipei, Taiwan, R.O.C.

SHYH-MING CHIEN and JEN-SHYH LAI

Department of Computer Science
Telecommunication Training
Institute
Panchiao, Taiwan, R.O.C.

ABSTRACT

In the advanced information society, more convenient and more diversified information management system should be provided to the user at a reasonable cost. The relational data base management system, especially QBE(Query By Example)-like system may be one of the best choices. Since it can be used easily.

The aim of this paper is to design and implement a Chinese Data Base Management System(CDBMS) for Chinese non-professional computer users. It can provide multiple functions, including storing, retrieving, generation and management of data. The CDBMS which based on QBE-like is established on the HP-3000 computer systems. The system adopts HP's KSAM (Keyed Sequential Access Method) as the internal file management tool and FORTRAN as implementing program language.

1. Introduction

In the Chinese data processing society we have to provide information management systems to serve as a tool for the management systems mainly covers two major categories, namely the file management system and data base management system. Recently some studies on Chinese file processing have been done [14], however existing Chinese file handling systems have certain defects, such as only handling one file per time, redundance of information, waste of space, poor information independence, and too much manpower involved. In the data base management aspect, although there are many English data base management systems available, they are not popular in our society. Firstly, because an absolute majority of the users of the Chinese information management system are Chinese people, and due to the language barrier existed, they feel extremely inconvient to use the English data base management systems, it is of necessity to provide a set of complete and easy to learn and use, the data base softwares with the Chinese data, due to the unique characteristics of characters, are quite different from the English data. For example, Chinese characters are all coded in fixed length codes and there is no word division problem as in English text. Consequently, it is necessary to design an easily applicable Chinese data base for our applications.

In this paper, we shall discuss the design of a easily applicable table-type Chinese Data Base Management System (CDBMS), based on the relation data model, which provides users with a variety of functions including the storing, retrieval, generation and management of Chinese information [15].

2. System Structure

2.1 Design considerations

All the common data base user belong to the non-computer professionals. Therefore, to design the language of the data base, the main object is the easy of use and memorize of the query language. In comparing the data base with the related human factors, many people have studied, on a trial basis, to use various measuring methods to evaluate the friendliness degree of different languages. For example, the average time required to completely write query, the proportion of correct query, and the confidence rating and kinds of errors, etc. [6,7,9]. Thomas and Gould pointed out QBE is easier to learn and use, the time spent on user traning is only one third of that for SEQUEL [8]. Later, Greenblatt and Waxman also concluded that in the easiness of learning and application, QBE is better than SEQUEL and the relational algebra [3]. So CDBMS adopted the concept and structure similar to that of QBE [10,11].

2.2. A general picture of the system structure

CDBMS adopts the relational data model to be established on HP3000 series III minicomputer system and executed by MPE-IV operating system. CDBMS is written in FORTRAN. The hardware structure of CDBMS is shown in Fig. 1. The Dragon Chinese terminal [13] provides the primary hardware interface between the users and the system. It is a multi-function Chinese and English intelligent terminal, and it provides the Chang-Je character input method and the vector output method. This terminal is connected to the main computer via a standard RS-232C. The screen display formats are 14 lines with 30 characters in each line, each character forms are structured by 15X14 dot matrix, and the Chinese internal codes are 4 bytes in fixed length.

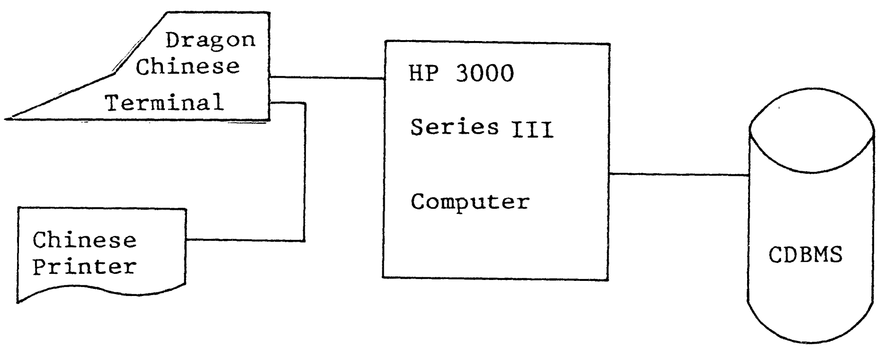


Fig.1 CDBMS Hardware Structure

As shown in Fig. 2, the software of CDBMS consists of four main modules, namely definition module, query processing module, I/O processing module and storage module. Their respective functions are described as follows:

(1) Definition module: To process the establishment and change of the data base schema and also to conduct the management work of the data base directory. This module includes the schema processor and directory manager.

(2) Query processing module: As the kernel portion of the system, this module is responsible for analyzing the overall query syntax, also processing various query operations, establishing various internal tables required for query processing, and through analysis, generating the sequence of query operations before execution of queries. This module comprises the query analyzer, query operation generator, query optimizer and query executor.

(3) I/O processsing module: This module is responsible for interfacing the man-machine interactions. An user talks to the system via man-machine dialogue and table display of CDBMS. Meanwhile, this module is also responsible for the chinese input/output work. This module is mainly composed of I/O format processor.

(4) Storage module: This module is exclusively responsible for the storage and management of the data files and key files in the data base. The Keyed Sequential Access Method (KSAM) [5] serves as the CDBMS data storage and access method to establish various data files and key files.

2.3. Storage structure and access method

The query processing speed on the data base is determined by the storage structure and access methods inside the data base. The data base often stores a massive amount of data in direct access storage device like the disc. By the current hardware technology, the time needed to retrieve the data stored in the secondary storge is a main factor to determine the query processing speed. Therefore, how to promptly and effectively retrieve the data from the secondary storage is largely determined by the file and index structure used. Although there are serveral methods [4] available to establish the file and index structure, unfortunately none of them can satisfy all the desired applications. Of which the B-tree structure has been widely used and henceforth become a popular structure [2] of the prevailing indexed files of the data base indeed. It mainly has five significant advantages [1] :

(1) Storage utilization is at least 50% at any time and should be considerably better in the average.

(2) Storage is requested and released as the file grows and shrinks. There is no congestion problem or degradation of

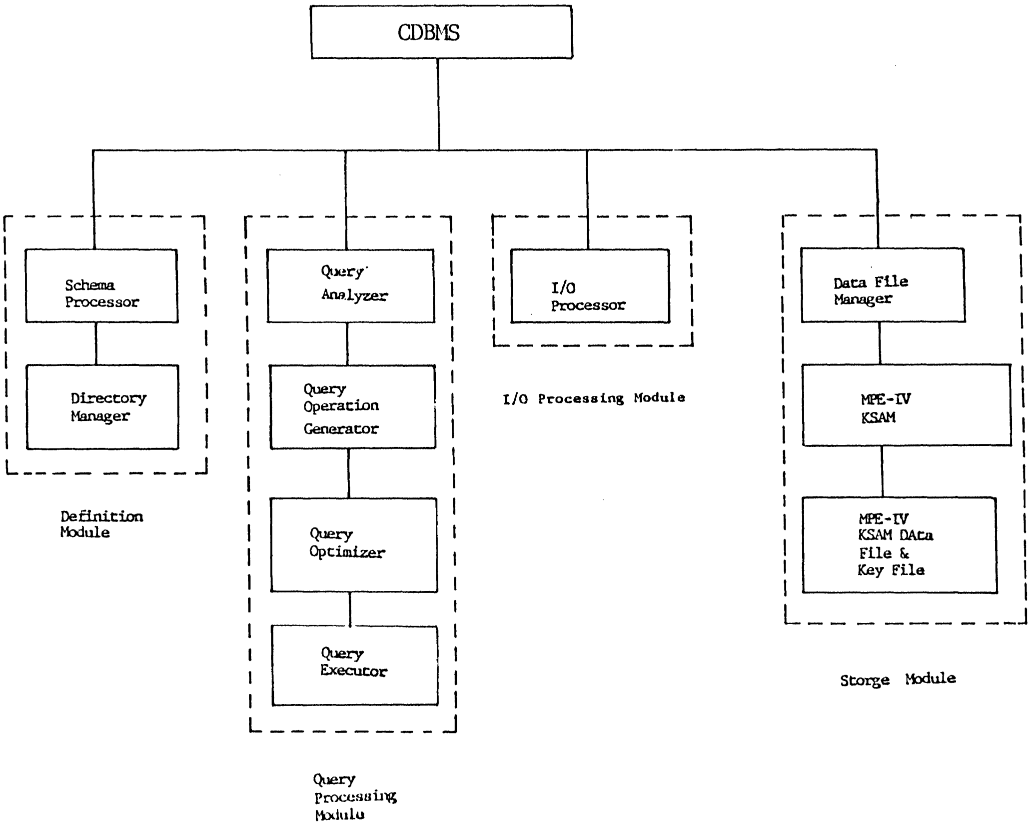


Fig.2 System Software Structure of CDBMS

performance if the storage occupancy is very high.

(3) The natural order of the keys is maintained and allows processing, such as find predecessors and successors, search the file sequentially to answer queries and skip, delete, or retrieve a number of records starting from a given key based on this order.

(4) If retrival, insertions, and deletions come in batch, very efficient processing, essentially sequential, of the index is possible by presorting the transactions on their keys and by using a simple prepaging algorithm.

(5) It does not need any periodic reorganization for all the files [2].

HP3000 system provides KSAM, which has been developed from the B-tree structure. KSAM not only has a perfect utility such as establishing, copying and supporting function but also provides KSAM directives for various languages to retrieve, update the data in KSAM. Therefore, it is decided to use the KSAM as the CDBMS storage structure, hereby save much time and energy on designing the storage structure and access methods in this system.

2.4. The syntax of CDBMS

In designing the syntax of CDBMS, we try to fit the need of easy usage on the one hand and match the display characteristics of the Chinese terminal on the other hand. Consequently, the syntax of CDBMS is divided into two kinds: one is the sub-language for data base definitions to be used for establishing and utilizing the new schema of the data base; and the other is the sub-language for the data base query to be used for indexing, query and modification of the data base. The menu selection and tables are used in combination to serve as the input purpose.

I. The definition sub-language (see Appendix A)

The entire definiton sub-language is composed of three expressions: "schema", "domain", and "relation" and Chinese character "End". A semi-colon (;) identifies each two adjacent expressions and a comma (,) separates each two sentences in various expressions. Because the key-in of the Chinese characters is extremely troublesome and time-consuming, to save the user's operating and thinking time, this sub-language uses, by all possible means, automatic guidance to lead the user data entry operation. The system will automatically display all the key words such as "schema", "relation" and "domain" to guide the user keying in, while there is no need for the user to type these Chinese characters in. At the same time, this definition sub-language can also provide various functions such as "deletion", "change", "addition" and "printing" to aid the user establishing the data base schema.

II. The query sub-language (see Appendix B)

This sub-language uses QBE-like structure. Since CDBMS uses the Chinese terminal apparatus, the following three different aspects exist between this sub-language system and the English QBE system:

(1) The query inputs of CDBMS use Chinese vertical table formats instead of English horizontal format as in QBE. To the Chinese input display method is easier to understand and operate, and all the domain names belong to a relation are vertically displayed on the screen, the user only needs to input the query directive to the line corresponding to the related domain.

(2) Except that the example element uses English words, all the remaining use the Chinese character inputs, so CDBMS can omit the identifying symbols of the example element in QBE such as the horizontal line inputs, thus simplifying the user's input burden.

(3) The functional key words, include printing, not, change, deletion , insertion, count, sum, average, max, and min, are represented by a single Chinese character, respectively.
Therefore, this can take full advantage of the characteristics of the Chinese characters that each of them is in an equal length, hereby simplifying the work of the query analyzer, and quite unlike the English QBE system, all the directives do not need a period (.) for separation.

3. System design and implementation

As discussed in previous section, CDBMS consists of four major module and query module. In this section we shall give an in-depth discussion on the internal design and implementation of these modules.

3.1. Definition module

This module is responsible for the establishment and modification of the new data base schema and the management of the data base directory.
To store the schema information of various data bases, the data base directory contains three system tables: data base table, relation table and definition table.

(1) Schema processor
A user can use this processor to define, via the I/O processing module, the data schema. Basically there are schema, domain and relation expressions in the definition sub-language. The schema expression defines the Chinese data base name. The domain expression defines the name, type and length of a domain, The relation expression defines a relation name, and the domain and key of this relation (in CDBMS, there is only one primary key for for each relation). Accordingly, the schema processor establishes above-mentioned three system tables and delivers them to directtory management for establishing the data base directory.

(2) Directory manager
All the newly defined schema after processed by the schema processor, and the queries issued by a user must be processed by the directory manager in advance for establishing a data base directory as required.
This directory manager also provide function to modify the current schema, to delete or rename the data base, to add or delete a relation or domain, or to rename or delete, or add a domain in a relation.
In addition to being responsible for managing the system tables, the directory manager also establishes the corresponding KSAM file according to the relation structure table via the data file structure table and via the data file manager of the storage module.

3.2. I/O processing module

This module exclusively processes the man-machine interaction portion between the user and the data base system and is mainly composed of the I/O foramt processor, which is basically in charge of the Chinese inpu/output work.

Via the the Chinese terminal, an user gives directives to select one of the three works: the establishment of new data base. Based upon the syntax of the languages for the selected work, the I/O format processor analyzes the Chinese sentences that the user has keyed in, and if it is syntactically correct, it will compose various related information into a schema or query buffer and then transfer the information to the schema processor or query analyzer for processing.

In the output aspect, the results from the query executor and the error messages,if there is any, are sent to the I/O format processor for being displayed on the Chinese terminal.

3.3. Storage module

All the information of the data base is stored in the discs. Via the keyed sequential access method, the data file manager of storage module establishes KSAM file retrieves the data in KSAM files. Functions of the data file manager includes:
(1) Based on the relation table to create new KSAM files.
(2) According to data base relation format, put the external information into KSAM files.
(3) To insert records into the designated KSAM file.
(4) To retrieve the stored data from the designated KSAM files.
(5) To delete records from the KSAM file.
(6) To modify records in the KSAM file.

3.4. Query processing module

This module is the kernel part of CDBMS. It is responsible for the analysis of the query syntax, and the generation of various tables as required, such as query level table, example table, query table and relation algebric calculation table. This module composes query analyzer, query optimizer and query executor, these are discussed separately as follows:

(1) Query analyzer
After the I/O processing module has made the syntax analysis on the Chinese query block, the related information such as relation name, domain name, example element, function code count, conditions are sent, via the query buffer, to this query analyzer. The main task of the query analyzer is to transform the related information into three intermediate tables: query level table, query table and example table.

(2) Query operation generator
The main function of the query operation generator is to utilize various tables established by the query analyzer for establishing a relation alegric calculation table which is used as a basis for the relation algebric processing. Each row in the relation algebric calculation-table represents a relation algebric calculation equation.

(3) Query optimizer
Basically, this optimizer contains seven procedures, as show in Fig. 3.
It is composed of the procedures such as Check 1, Check 2, Check 3, Check 4, single relation processing, two-relation processing and objective domain setting.
The main function of the optimizer is to analyze each row in the relations algebric calculation table. Thus we can identify various rows that can be processed simultaneously to reduce the total number of rows in this table so as to cut the number of the necessary calculations accordingly. At the same time, to give proper relation algebric calculation according to the characteristics of each query.

(4) Query executor
Based on the relation algebric calculation table build by the query optimizer, the query executor sequentially executes and processes the corresponding computations according to their respective operation code. At present, CDBMS only processes the basic operations: select, project, select and project , join and tabulation printing, while other operation functions such as substraction will be added later on.
This executor is also responsible for the execution of the built-in functions, include counting (all and unique), summation (all and unique), average (all and unique), minimum value, maximum value, and grouping.

4. Conclusion

The CDBMS system combines the essences of the table language and menu-select method to provide a simple syntax and easy to learn and use Chinese query language, which is convenient for the non-professional users in their dialogue with this system. Further, this system adopts the automatic guidance to reduce, by all possible

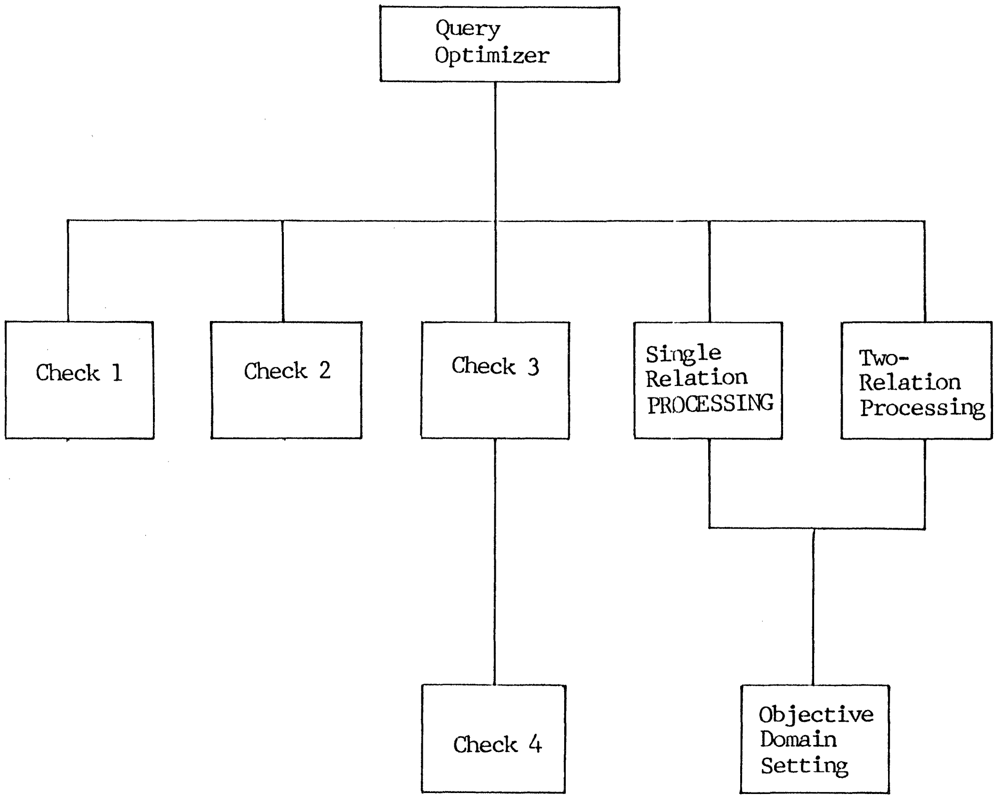


Fig.3 Internal Structure of Query Optimizer

means, the effort of entering Chinese characters by the users, so a user only needs to be trained in a simple manner without understanding the internal structure of the data base in depth, he can use this system well. Besides, since all the error messages in this system can be displayed in Chinese, the users will feel this system more friendly.

This system utilizes the principles of top-down design and modulization to construct the modules. FORTRAN language is adopted for the implementation of this system, so it is very convenient in its maintenance. If a Chinese terminal of other type is fitted on this system as a replacement, except that minor modification must be done on the Chinese error messages of the I/O processing module, all the other part remain unchanged.

Subjected to limited time and man power, the design and implementation of CDBMS now has only achieved basic functions. This system may become more complete, if the functions of this system can be added with such functions as multi-users characteristics, privacy controls, backup and recovery capabilities.

Just like the common Chinese data processing systems, the functions of CDBMS are subjected to many limitations of the Chinese terminal. For instance, the Chinese display of Dragon Chinese only has 14 lines with only 30 Chinese characters in each line, so the total number of characters displayed is too small indeed. On the other hand, the Chinese characters input method is still rather troublesome to the non-professional users, while the characters selection is susceptible to difficulties. Therefore, to improve all these and develop a multi-functional intelligent Chinese terminal by making it suitable for the Chinese data processing and the Chinese characters processing is of the great importance in the development of the Chinese computers.

If this sytem can get more powerful and intelligent Chinese terminal in the future, it maybe further enlarged to serve as a basis for studying the Chinese characters office automation in the business. For instance, it may use the specially designed OBE (Office-By-Example) system similar to Dr. Zloof of IBM, since OBE system is based on QBE language by combining data processing, word processing and electronic mail in the OBE language for the office automation in the business [12].

References:

1. Bayer, R. and McGreight, E. "Organization and Maintenance of Large Order Indexed, "Acta Informatica, Vol. 1, 1972, pp. 173-189.

2. Comer, D. "The Ubiguitous B-tree", computing Surveys, Vol. 11, No. 2, June 1979, pp. 121-137.

3. Greenblätt, D. and Waxman, J. "A Study of Three Database Query Language", In Database: Improving Usability and Responsiveness, Ben Shneiderman (Ed.), New York, Academic Press, 1978, pp. 77-79.

4. Knuth, D.E. The Arts of Computer Programming, Vol 3: Sorting and Searching, Addison-Wesley Publ. Co., Reading, Mass,. 1973.

5. KSAM/3000 Reference Manual, Part No. 30000-90097 HP Santa Calif.

6. Reisner, P. "Use of Psychological Experimentation as an Aid to Development of a Query Language, "IEEE Transactions on Software Engineering, Vol. SE-3, No. 3, May 1977, pp. 218-229.

7. Reisner, P. "Human Factors Studies of Database Query Languages: A Surveys and Assessment", Vol. 13, No. 1, March 1981, pp. 13-31.

8. Thomas, J.C. and Gould, J.D. "A Psychological Study of Query By Example", Proc. National Computer Conference, AFIPS Press. Arlington, Va., 1975, pp. 439-445.

9. Welty, C. and Stemple, D.W. "Human Factors Comparison of a Procedural and a Noprocedural Query Language", ACM Transactions on Database systems, Vol. 6, No. 4, Dec. 1981, pp. 626-649.

10. Zloof, M.M. "Query-By-Example: A Data Base Language", IBM System Journal, Vol. 16, No. 4, 1977, pp. 324-343.

11. Zloof, M.M. "Design Aspects of the Query-By-Example Data Base management Language", In Database: Improving Usability and Responsiveness, Ben Shneiderman (Ed.), New York, Academic Press, 1978, pp. 29-55.

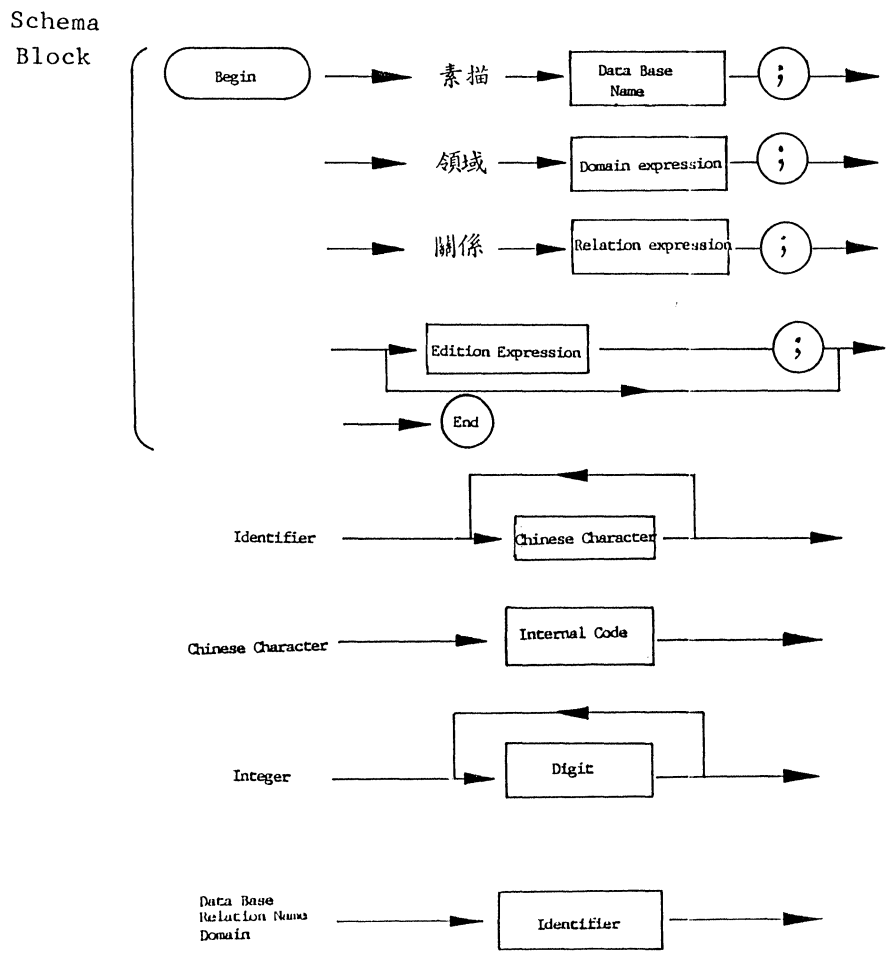
12. Zloof, M.M. "Office-By-Example: A Business Language that Unifies Data and Word Processing and Electronic Mail", IBM System Journal, Vol. 21, No. 3, pp. 272-304.

13. Dragon Chinese Terminal Operation Manual, Multitech Co., Taipei, R.O.C.

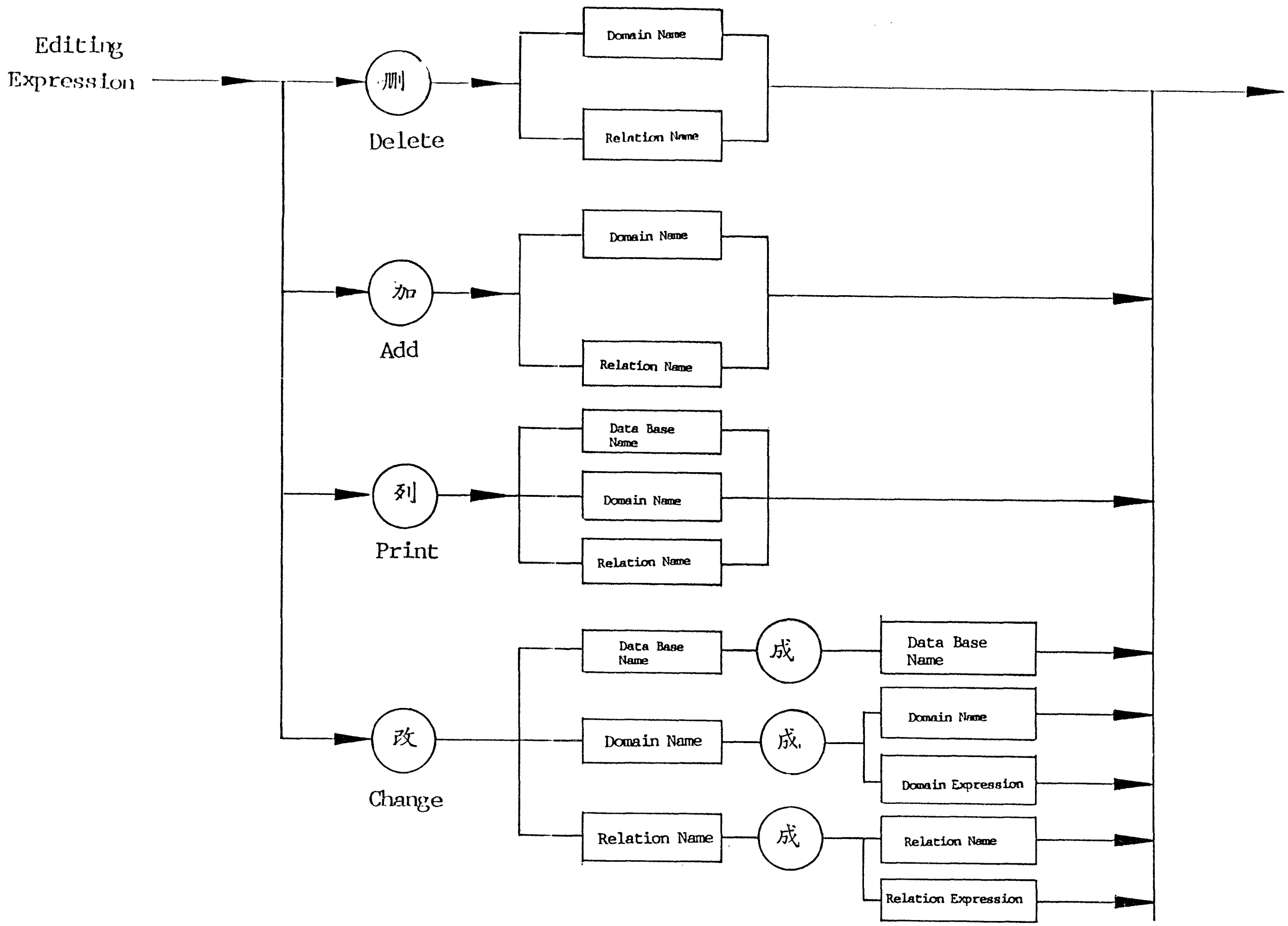
14. Su, S.C. Chinese Automatic file processing system, Master Thesis, Tankang University, Taipei, R.O.C., 1982.

15. Yang, C.C. and Chien, S.M. "The Design and Implementation of Chinese Data Base Language", National Computer Symposium, Sin-Tsu, Taiwan, R.O.C., 1983, pp. 136-146.

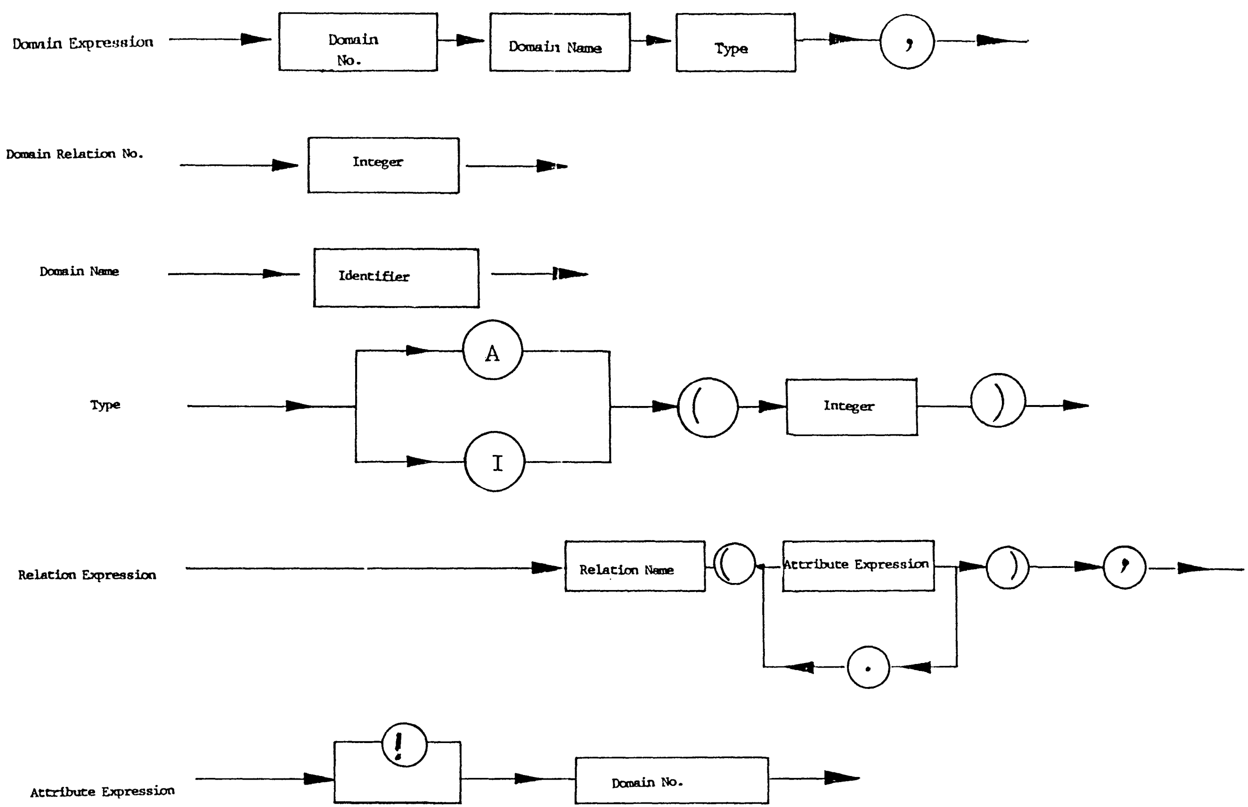
Appendix A The Syntax of CDBMS Definition Sub-Language



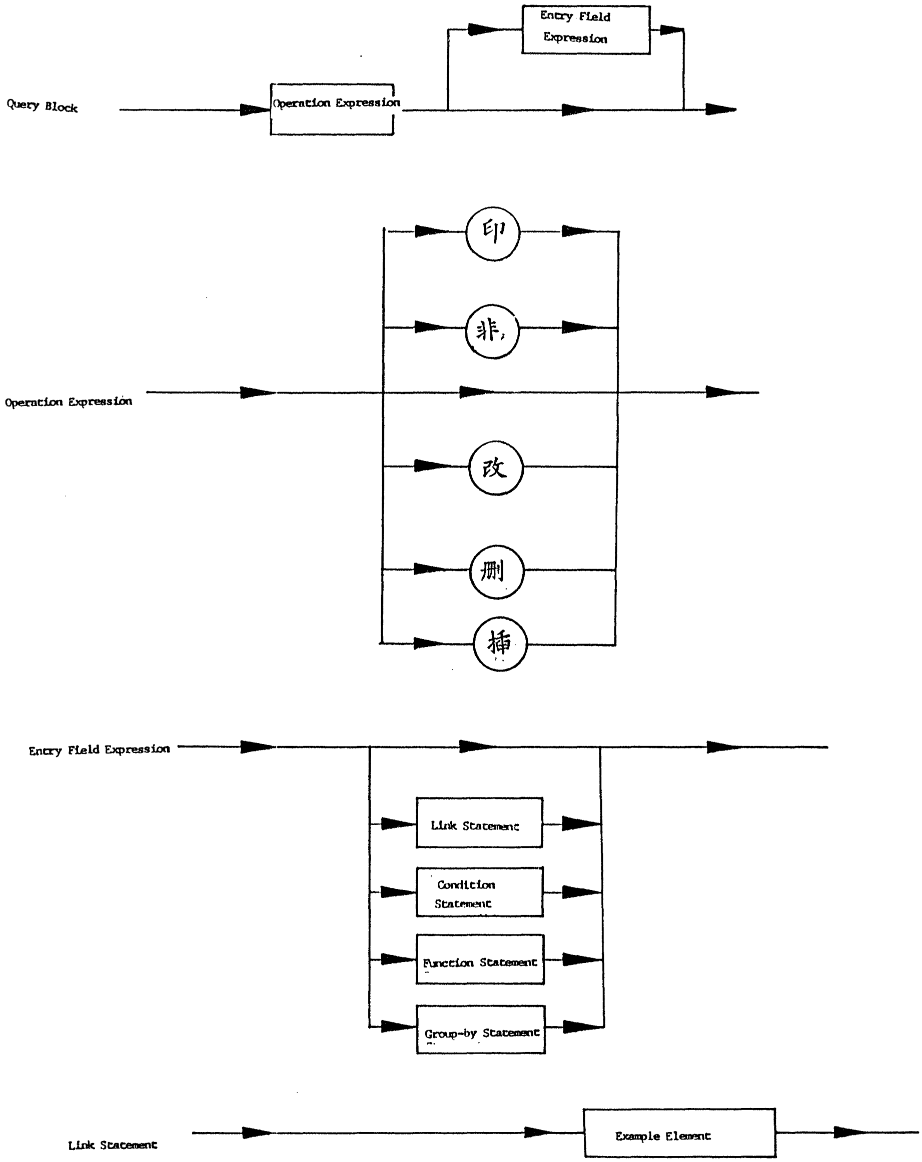
Appendix A The Syntax of CDBMS Definition Sub-Language(Continue)



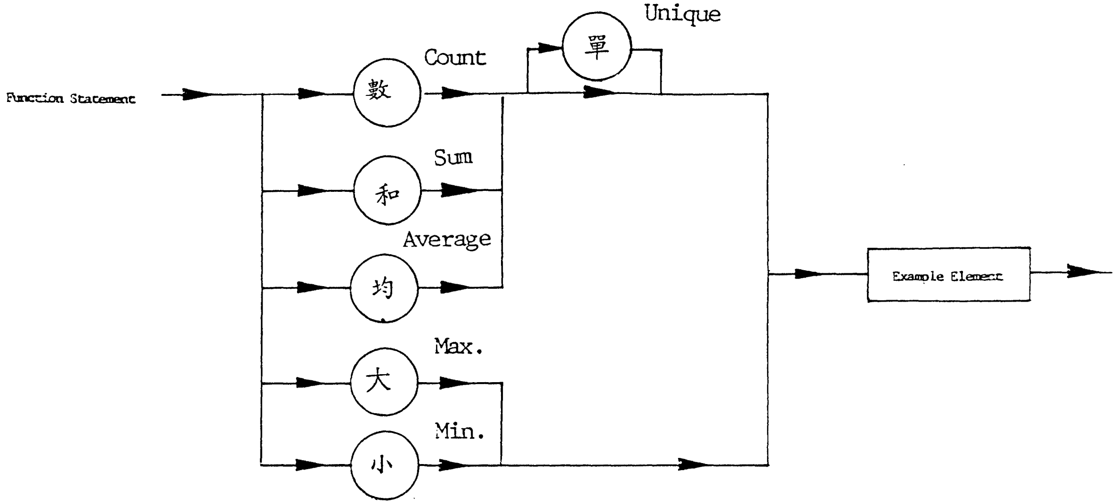
Appendix A The syntax of CDBMS Definition Sub-Language(Continue)

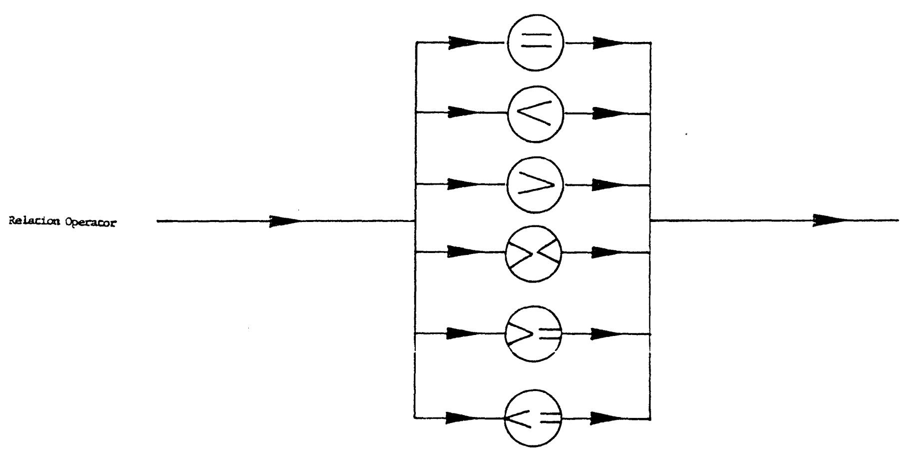


Appendix B The Syntax Diagram of CDBMS

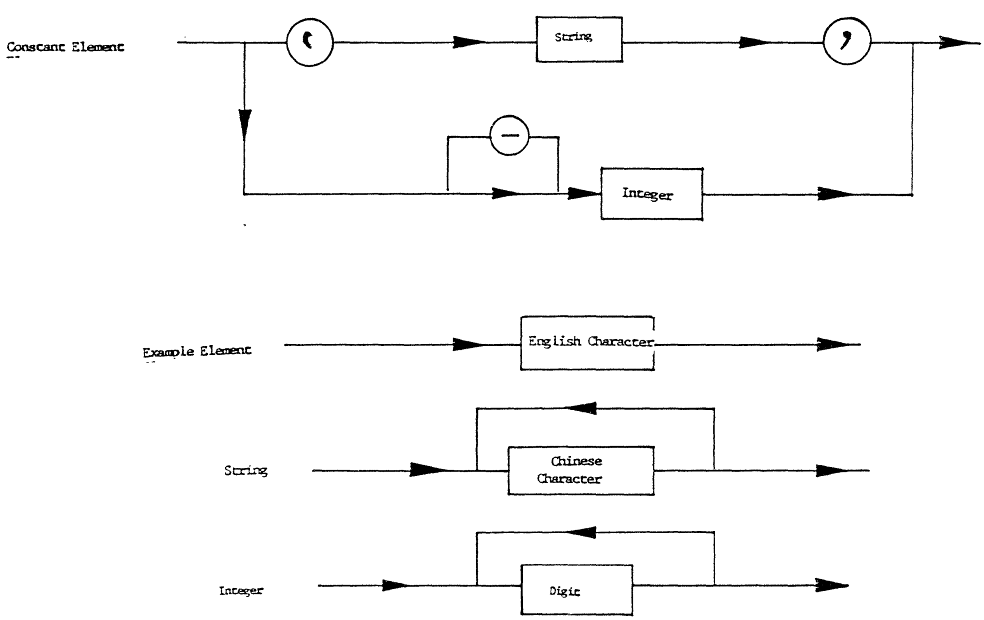


Appendix B The Syntax Diagram of CDBMS(Continue)





Appendix B The Syntax Diagram of CDBMS(Continue)



DC - DATA COMMUNICATIONS AND NETWORKS

DESIGNING A NETWORK TO MATCH YOUR OPERATING ENVIRONMENT

Philip Aramoonie and Arie Scope
Hewlett-Packard, Cupertino, California, USA

INTRODUCTION

In today's business environment, the professional needs transparent information access across a wide variety of application environments such as finance/administration, marketing/sales, manufacturing and engineering. One also needs access to these information services and office functions such as electronic mail, document (both text and graphics) creation and management without worrying about link topology and technology.

This paper will discuss techniques which will allow such access such as system and network directories for transparent access of data, people, and peripheral devices. These network directories/dictionaries should have access to a variety of generic network services such as Network File Transfer, Inter-Process Communication, Virtual Terminal Access, Remote File and Data Base Access and Remote Peripheral Access with Network Spooling. These services should have access to a variety of Transport / Link services such as Point-to-Point, LANs, X.25 PDNs, satellite or gateways where appropriate for the system, application and geography.

NETWORK ENVIRONMENTS

A network is a multi-vendor environment with many vendors supplying a variety of systems, workstations, links, modems, data switches, multiplexers, etc. Basing your network and products on an architecture which conforms to industry and de facto standards is the only way to ensure lasting value of the network for future product compatibility. Such a network should be based on the International Standards Organization's Open Systems Interconnection (ISO/OSI) reference model. This reference model segments data communications into seven (7) layers. These layers are grouped into two basic functional areas.

The first four layers are the Transport Services which allow a device or system to connect to another and pass data between them. Layer 1 is the specification for the electrical connection to the communication media. Wide area or point-to-point connection may employ standards such as EIA RS-232C or CCITT V.35 while local, high-speed connections may use IEEE 802.3, 802.4 or 802.5 Local Area Networks. The second layer is the first level of software and defines how the information is to be formated and sent. Common protocols for this level include HDLC (High-level Data Link Control) for wide area communications or CSMA/CD (Carrier Sense Multiple Access with Collision Detection) and Token Passing for higher speed Local Area Networks (LANS). A workstation or data terminal device may only need a simple "hand-shake" or acknowledgement that the device is ready to accept information with a computer if a dedicated line per device is used. Common handshakes are reciever controlled (X-on, X-off) or transmitter controlled (Enk,Ack). Other protocols not defined as industry standards (but are certainly de facto standards) are IBM's older Bisync and newer SDLC (synchronous data link control, a subset of HDLC). Layer 3 defines the method for addressing or identifying where the information is coming from and where it is supposed to go. A standard at this level implemented in varying degrees is the ISO recommendation of X.25. Another definition which is used is the Internet protocol for local area networks and gateway services. The forth layer is the Transport Protocol which is software at each end which manages the flow of information to make sure the packets of information arrive in the correct sequence, are error-free, and that the systems can effectively handle the arriving information. There is no ISO accepted Transport standard today but a recommendation (which is similar to the TCP used in ARPANET within the U.S.) similar to those defined by the U.S. National Bureau of Standards and the European Computer Manufacturers Association may be adopted within the next few years.

The upper three (3) levels comprise the application services which synchronize the two or more nodes with the appropriate network service and then identify and interpret the data so that the application process can use it. Layer 5 is the Session Level which provides user interface network services such as Process-to-Process communication, Network File Transfer, Remote File Access or Virtual Terminal connections between the system nodes. Once information is sent and received, it must be interpreted as to the kind or type of data and then formated for presentation to the appropriate application. Layer 6 or the Presentation Layer performs this function. The Application process is the Seventh Layer. In extended architectures, Data, Device and People Dictionaries are on each system node with directories for other network nodes as indices. These provide the user with fully "transparent access" to programs, data, devices and other users within the network.

One may also require the capabilitiy to manage the network from as many or as few nodes as required by the network layout which should be based on the business environment. Many factors such as the functional, geographic and political distribution of information services need to be taken into account in determining how the network is structured and managed. The two major areas of of Network Management are first, Data Standards or specifications contained in the organizational applications maintained by data bases or files and data dictionaries with network directories to identify which applications and their respective data elements are on what systems. The second area is operations management for proper configuration, monitoring and maintenance of the network.

A network which provides solutions that are as easy to use as single system solutions, have a broad range of price/perfromance alternatives and is based on an open systems interconnection transport and application services will offer the most flexible, cost-effective way to allow any user from a PC to the largest multi-user system to draw freely on files, programs an computing resources of the network as often and as easily as using the telephone.

Networks may be segmented into three (3) primary operational levels each with several application areas. These are workgroup networks, department/divisional networks and Company-wide Networks.

Workgroup networks are primarily defined by a group of closely related individuals or machines (in manufacturing workcenters) sharing a related functional set of applications in their respective application environments. They may have specialized or intelligent workstations connected to a higher performance, multi-user system to provide organizational data sharing, personal file serving and peripheral sharing.

Department/divisional Networks are those comprised of the functional or geographic collections of workgroup systems as well as departmental information data bases and dictionaries.

A company-wide network is the collective interconnection of the department/division networks to provide such services as company-wide electronic mail, application information access and summary reporting from the divisions' functional or geographic networks to corporate data bases such as centralized financial, personnel, payroll, distribution or manufacturing systems.

OFFICE AUTOMATION / DISTRIBUTED DATA PROCESSING NETWORKS

Office automation is the combination of office functions, personal computing for decision support and filing with access to data from other workgroup or related department/division application systems in a network to increase the effectiveness of the "knowledge workers" in the company. The heart of the office for decision support systems will be Intelligent Personal Computers or specialized workstations tied to a multi-user workgroup computer via point-to-point connections or over existing telephone cables through a digital switch or PBX or multi-dropped over dedicated high-speed Local Area Networks (LAN).

Office automation users may need to transparently, through applications, access data to add, update or delete information directly to databases, spread sheets, electronic mail messages, documents with text, data and graphics, graphs and slides. The information may reside on the workgroup's local network or system or the interdepartmental local area network as well as the division or corporate systems or mainframes. Interconnection to public data networks, public data bases such as Dow Jones or The Source with the ability to easily incorporate their information into decision support systems for analysis or reporting is also a requirement.

Transparent Information Access

In order for the user to transparently access this information in the network, the system nodes must have dynamic, up-to-date directories for people, application data and devices. Directories for people are those commonly found in intersystem elctronic mail services. The sender need not know where or on which system the receiver resides; only the person's name. Directories for application data access are indexes to system or application data dictionaries so that an application user need not know where or on what system(s) the information physically resides, but relies on the directory to establish the connection to the appropriate system or systems. Peripheral sharing might also be accomplished by using a directory to establish appropriate links and initiate network services to perform serving or spooling to devices which, because of functionality or cost, may not be easily duplicated at each node. The Directories are not enough, however. The applications may use a variety of data files formated as documents (both text and graphics) as well as different data bases or message files across a variety of multi-vendor or different system architectures. Appropriate file converters or translators may have to be employed within each application areas to assure proper interpretation of the data formats and values by the application process.

Formats for various file types such as text, graphics, and data tend to be unique according to the application program and vendor, and therefore, need to be standardized or conform to an open architecture so that file converters can be easily designed and employed. Some file formats for text, graphics and data files are becoming de facto standards such as MS-DOS DIF files or DIA/DCA text files under IBM's SNA.

Network and Transport Services

Because of the increasing demand for "knowledge workers" to increase their effectiveness, the intelligent workstation or Personal Computer is the predominant interface into the information network. It allows the user to file and process data used on the job which may be personal in nature or unique to the individual without encombering the network resources for storage or access. The workstation is tied to the workgroup network in a way which is determined by a number of factors. The connection should be low in cost since the workstation is relatively low in cost, flexible because people and job functions change and provide access to local and global information sources within and outside the company.

In a local environment where information needs are ad hoc or transaction-based in nature, point-to-point connections either directly attached, through a digital switch, or digital PBX is the prefered way. It offers low-cost connections especially if existing twisted-pair telephone circuits can be utilized. It also offers flexible attachment and modification to the network in the same way that voice phones offer.

Where longer distance connections are required, workstations can be clustered through multiplexers; Time Division Multiplexing/TDM, where 2-4 workstations are needed, or Statistical Multiplexing, where 4 - 16 to 32 workstations (depending on the application transaction size) are needed. The multiplexers are then attached to dial-up or wide area public data networks (X.25), dedicated leased lines or higher-speed digital circuits to a workgroup computer or application engine. Twisted pair, coaxial cable, or microwave links may be employed depending on the volume of data, number of devices and cost. Where more than 16 to 32 workstations are needed and if the application can be distributed, one should consider the use of a remote workgroup computer to lower the communication expense.

Twisted-pair in "clean" environments can easily handle data rates of 64 kbits/sec. for distances up to 2 KM. With appropriate transceivers and data protocols such as those employed by T1-DMI or IEEE 802.3 - CSMA/CD Local Networks, PC's can reliably employ 1 megabits/sec. or faster data transfer rates over twisted pair. These higher speeds allow for higher data volumes and make such network services as Inter-Process Communication, Network File Transfer, Remote File and Data Base Access and Remote Peripheral Sharing more practical.

Where very high data volumes are used in local dedicated application environments such as dedicated document (text and graphics) creation and management, personal or application file serving, or peripheral sharing, a dedicated LAN using baseband (50 ohm coax) or a single channel on a wider area broadband LAN (75 ohm coax) should be considered although the higher cost per connection must be justified. Baseband coax links exist today such as Ethernet or IEEE 802.3 employing protocols optimal for bursty communication requirements such as Carrier Sense Multiple Access/with Collision Detection (CSMA/CD). Some local area networks exist today that employ this protocol but use a dedicated 75 ohm broadband link to overcome the distance restrictions of digital baseband.

The workgroup computers also need to be attached to other functional (application) or geographic workgroup systems to form the department or divisional network. If the systems are geographically close and share moderate to large volumes of data or where the systems are functionally distributed application engines in the same computer room, the prefered link may be a Baseband LAN using the IEEE 802.3 or Ethernet Rev.2 protocols over a 50 ohm coax. This link offers a 10 megabit/sec. bandwidth at a relatively low cost per system. One or more systems in this configuration may operate as a electronic mail hub or gateway to company or outside wide area services such as SNA, X.25 or PDN, or Telex. This type of department/division network also allows for better data management at the local level to automatically back-up local files and data bases as well as keep transaction logs of locally maintained corporate information.

In summary, general distributed data processing and office automation application environments require Transparent Infromation Access with the ability to move information between different applications residing on multiple systems which are organized in different file and data base formats. Meeting this need requires the following: network directories/dictionaries which maintaines indices for the location of systems, application programs and data, peripherals and people; common interface formats for specific applications with the ability to interpret/convert file and data base types to and from it; and the ability to automatically connect a workstation or system to any other systems which contain the needed data among a variety of local and remote locations.

Flexible, multi-vendor communications based on the ISO Open Systems Interconnection reference model to interconnect to any of the company's computers worldwide with options to use a variety of communicaton links between systems and workstations to match the different costs, speed and flexibility needs of each application environment is a way to assure lasting value of the network.

Network management with the ability to have remote systems managed from a central site at the operations level include remote back-up and systems management, the capabiltiy to configure links remotely without bringing that node or network down, keeping accounting and statistics on line loads, access, disc space used, system/network usage and connect times, etc. and the ability to easily trace and diagnose network problems to the replaceable unit. Network Directories using available network access services also allow for applications and data standards to be developed and maintained centrally.

MANUFACTURING NETWORKS

The factory floor of many manufacturing companies tend to be large, dirty, and subject to temperature extremes. Computers singly or in small groups dedicated to specific manufaturing functions make up automation islands on the factory floor. Each island is usually dominated by computers or controllers from a single vendor. Corporate , division, and plant-wide information systems tend to be mainframes or other general distributed data processing systems used for transaction-based materials, production or maintenance administration/management. These systems, in many cases, need to interface to real-time supervisory control systems in the various production workgroups or centers. The supervisory level systems may manage, monitor or control the first level controller which, depending on the task, may be a multi-tasking real time computer, programmable logic controller, or numerical control machine. By interconnecting these islands of automation to the supervisory level systems and operations management systems into one large, transparent, muti-vendor network, productivity increases will result by using a common set of data bases which have the latest planning, process and control information.

The resulting network structure may take the form of a hierarchy of system levels with the planning and operations control transaction-based systems at the top, interfaced to order processing, financial systems and engineering networks at the divisional level in a peer fashion or to centralized sales or financial systems one level above. Below the manufacturing management systems are supervisory real-time systems which may be independent if the workcenters are, or synchronized in a hierarchical or peer-to-peer fashion if workcenters are time phased or dependent on each other. All of these systems and controllers may be multi-dropped off of the same network media (broadband LAN) with independent channels for dedicated access or have dedicated point-to-point links depending on plant geography and functionality.

In many manufacturing firms, the large-scale, harsh environment of the factory floor has led to the adoption of networks designed to broadband CATV industry standards as the desired cabling system. Communication among machines may require predictable response times for monitoring and control. A communications protocol such as token-passing on a broadband bus which offers a deterministic response time is required. The IEEE 802.4 Specification encompasses these features. This type of network configuration is justified where many (>100) first level controllers need to be interfaced to a large number of supervisory level systems which need to be synchronized for a production line or process over distances which exceed the 2500 meter limitations of baseband schemes.

Smaller manufacturing environments or workcenter clusters which are independent of each other with 40 or fewer devices (intelligent programmable controllers) may be adequately served by IEEE 802.3, CSMA/CD, 10 megabits/sec. baseband links tied to a real-time supervisory level system. The physical link could be 50 ohm coaxial cable (thick Ethernet or thin RG-58U) limited to distances of 300 to 500 meters or a single channel of 10-12 Mbits/sec. on a larger broadband factory link.

If the environment suffers from electrical noise, the supervisory systems and first level controllers may require dedicated fiber optic links which are immune to electrical interference. Point-to-point attached fiber optic links can extend in many cases up to 2 km.

ENGINEERING NETWORKS

The engineering market has been characterized by multi-user mainframes and super-minicomputers. Recent trends are towards high performance, individual engineering workstations with UNIX as the prefered operating system. Two key application areas are Computer Aided Design/Engineering (CAD/CAE) and Computer Aided Test (CAT). Typical applications involve finite element analysis, modeling, simulation, 2 or 3 dimension layout for wiring/piping/printed circuit board design, drafting and product testing. Target users are electrical, mechanical and process engineers.

Because of the massive volumes of data required in this environment, high-speed local area networks featuring transparent file and peripheral sharing and serving are needed. The high performance workstations must also interface to the manufacturing management systems for materials specifications and costing as well as provide the engineer with some of the typical office functions such as electronic mail and document preparation and management through the application data and people directories. The appropriate network services required to interface to the other areas of manufacturing and financial/administration are network file transfer or virtual terminal access. In addition to these network services, remote file and peripheral access, and remote process management are required within the workgroup local area network. Designated systems within the workgroup also need to act as a resource manager in order to leverage the cost of large, expensive disc drives and plotters or other unique peripheral devices. Automatic, centralized archiving and back-up of data bases is also a requirement.

SUMMARY

In many cases today, business professionals are asked to make important decisions without accurate, timely information. Interactive personal workstations tied together into workgroup and department information networks that are linked to corporate information systems may be a way to provide this information. In order for the network to be used to its full potential, the user must have easy, transparent access to application programs and related data files, documents and graphics as well as other network users.

Providing Network Directories/Dictionaries and layering these on top of system and application dictionaries with appropriate file interpreters/converters which can use generic network access services where appropriate for the application is a way to achieve this transparent access. The appropriate physical links and transport protocols are then used for the funtional and geographic connection of systems-to-systems, systems-to-workstations, and workstations-to-workstations.

Basing these various levels of network software and hardware on an Open Systems Interconnection reference model provides the following benefits:

- The ability to connect into and pass data within a multi-vendor environment with standard transport and link services;

- A flexible network topology which can change as the company grows or technology changes without having to change any application software;

- A common, easy-to-use interface to directory and network services so users need only understand one operating environment for the workstation and network;

- Management of the network from as few or many locations as required for Data and Document Management Standards and Operations Management for Configuration and Maintenance of the network components.

<u>Biography</u>

Phil Aramoonie
is the Network Sales Center manager with responsibility for presales support of Networking Products on HP 3000, HP 1000 computer systems for the USA and InterCon Field Sales Force.

Arie Scope
is the Manager of the Product Marketing Council for the four Network Products Divisions at HP with primary responsibility for coordinating the products marketing strategy and development among the various divisions.

ADVANCES IN THE WORLD OF DATA COMMUNICATIONS

KEN BAYNTON

MICOM-BORER, READING, ENGLAND.

SUMMARY

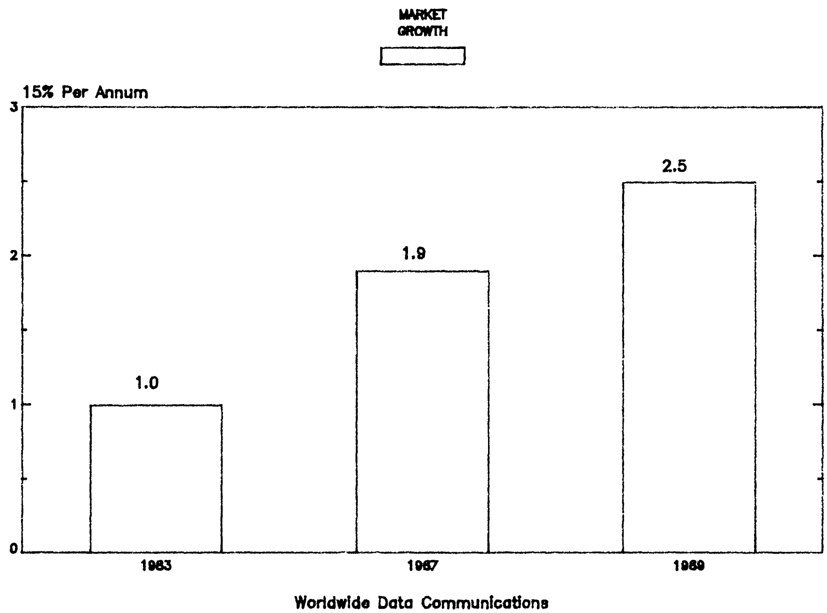
Data Communications technology is responding to the fast growing needs of the world of computers and represents one of the most dynamic sectors of electronics technology. Newer techniques are reducing costs and extending the availability for the transmission of even larger volumes of data between computers and users. Market trends, standards and techniques are of vital importance to organisations planning computer networks to fulfil distributed processing needs. This paper outlines some of these facets which are relevant to HP 3000 users.

MARKET TRENDS FOR DATA COMMUNICATIONS

Data Communications no longer confines itself to the establishment of mainframe centred networks for significant corporate computing requirements. It covers the need for mainframe and micro computer users alike. With the proliferation of personal and home computing the thirst for accessing, disseminating and distributed processing has grown fast and is unlikely to diminish in the foreseeable future.

Communication over distances of more than a few miles has historically revolved around the utilisation of the telephone network. Devices called data modems convert binary digital information into analogue waveforms that can be transferred over dedicated or dial up circuits. Despite predictions that this outdated approach would diminish with the advent of new methods, growth is still high.

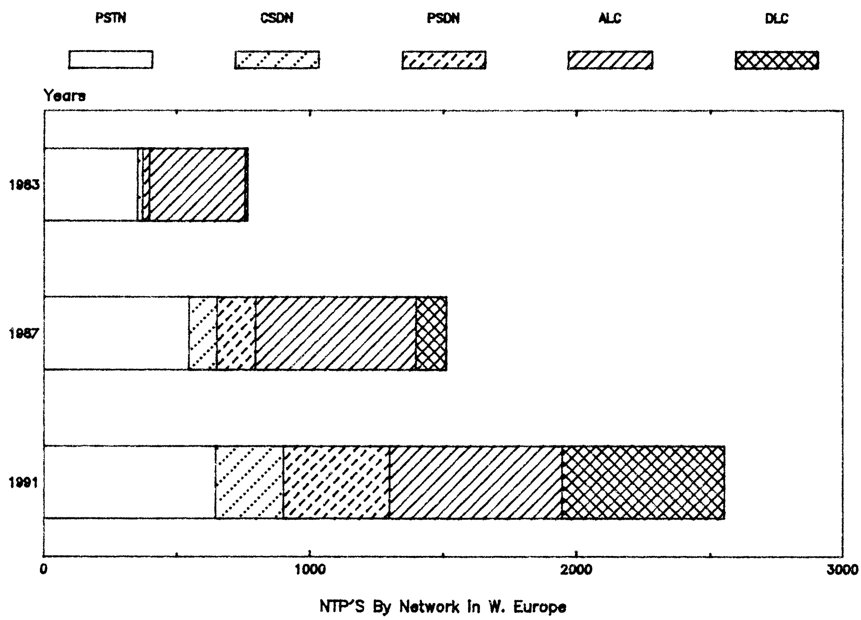
Fig. 1.



During the early 1970's digital networks were thought up. Such networks would handle digital data only and would represent much greater efficiencies in terms of communications plant utilisation. These networks involved the generation of packets of data information that are despatched from a source to a destination across a virtual network. The emergence of digital services has been slower than predicted back in the pioneering years but despite this slow start, it is currently expected that they will handle 10% of data traffic by 1989.

More recently during the early 1980's PTT's around the world have begun the implementation of dedicated digital transmission services providing high speed transparent pipelines for data transmission purposes. Such services utilise existing pcm carrier links and in the future will employ satellite tranmsission methods. With the availability of high volume data pipes it is unlikely that many organisations will confine their use purely to computer data but rather will combine the needs for voice and data communication. In the U.K. the PTT British Telecom expects that transparent digital services will account for 40% of all data transmitted by 1989.

Fig. 2.



Distribution of NTP's by Network Type in Western Europe

The combination of these data transmission methods - conventional and less conventional - will form the basis that will allow the information industry to accelerate fast. Predictions indicate that Network Termination Points (NTPs) will grow at an approximate rate of

15% per annum for the remainder of this decade (based on Eurodata Report 1983).

If previous reports are analysed the growth reality is likely to be much greater than theoretical predictions.

STANDARDISATION - FACT OR FICTION?

Data Communications embraces the need to communicate between different machines using different communications techniques often over geographical national borders. The international body that is chartered with defining data comms transmission standards is CCITT (International Consultative Committee for Telegraphy and Telephony). Over the years CCITT have generated recommendations for international communication for

Telephone Networks

Telex Networks

Data Networks (analogue based)

Data Networks (digital based)

More recent ratifications include:-

Data Transmission

V22bis	2400 full duplex dial up
V26ter	2400 full duplex dial up
V32	9600 2 wire full duplex

CCITT pioneered the standard for packet switching networks X.25 back in 1976. With the complexity of such specifications it is inevitable that every conceivable loophole is not addressed. The inevitable outcome is that many PTTs have their own particular quirks associated with X.25 which makes the standardisation for the attachment of apparatus to network difficult.

DE-FACTO COMMUNICATION STANDARDS

Two de-facto standards will dominate the communcations scene today:-

a) Asynchronous ASCII. This represents the simplest most universally adopted language for communication between terminal devices. It is estimated that 65% of the world's terminal equipment uses this interfacing method.

b) IBM SDLC. With IBM's domination of data processing it is inevitable that the alternative standard would be IBM defined. SDLC - a full duplex communications protocol - is the successor to BSC or BISYN (half duplex protocol). It is estimated that 25% of all the world's terminal devices utilize this protocol procedure.

RECENT MODEM DEVELOPMENTS

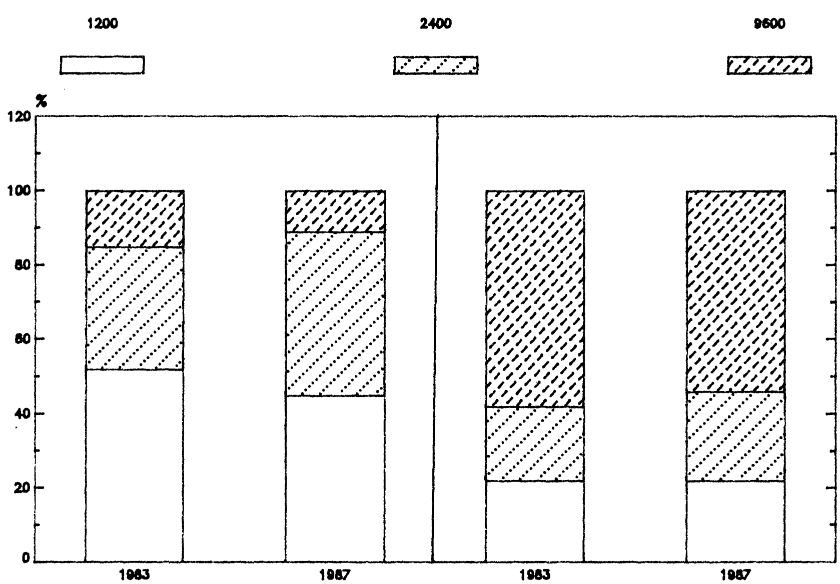
Dial up modem technology has been steadily advancing over the years. V22 modems (1200 bps full duplex over 2 wire circuits) have now been on the market for some five years. The flexibility that dial up offers is so appealing for the smaller system user that the demand for V22 modems has been enormous. Recent LSI and signal processing techniques have enabled the greater technology necessary to implement a 2400 bps full duplex modem to be developed at prices that make this speed viable. CCITT ratified two methods of modulation arrangement for 2400 bps full duplex.

V22bis	600 baud modulation rate 8 phase vector configuration.
V26ter	1200 baud modulation rate 2 frequency divided carrier 4 phase vector configuration. Waveforms coincident carrier signals, echo cancellation.

Currently V22bis is the specification that is being adopted with most modem manufacturers and several units are now being marketed around the world. Market forecasts predict that sales of 2400bps full duplex modems will escalate rapidly replacing 1200bps as the 'standard' within three years.

The provision of even higher speeds over the PSTN is the subject of considerable research and development. Already two 4800bps 2-wire dial up modems are being marketed and 9600 bps will be available shortly. Unfortunately as the modem speed increases susceptibility to telephone line conditions also increases resulting in higher error rates and corrupted data. Until higher quality telephone connections are implemented 2400 bps may be the maximum practical data rate for full duplex dial up modems.

Fig. 3.



Percentage values distribution of modem speeds.

Percentage value of revenue.

For leased line applications data rates are also increasing. A range of modems are now available operating at speeds of 14,400 bps or greater. Higher speeds are achievable as a result of rapid advancement in signal processing technology necessary to decode a 64 level carrier signal. In order to transfer data at high speeds complex digital automatic adaptive equalising techniques are required in or to compensate for line variations. Since at very high data rates modems are functioning on the edge of line characteristics it is important to adopt modems that automatically reduce speed in the event of high error rates and subsequently 'up speed' when line conditions improve.

DIGITAL SERVICES

These are communications systems that have been implemented specifically for data purposes.

PACKET SWITCH SERVICES

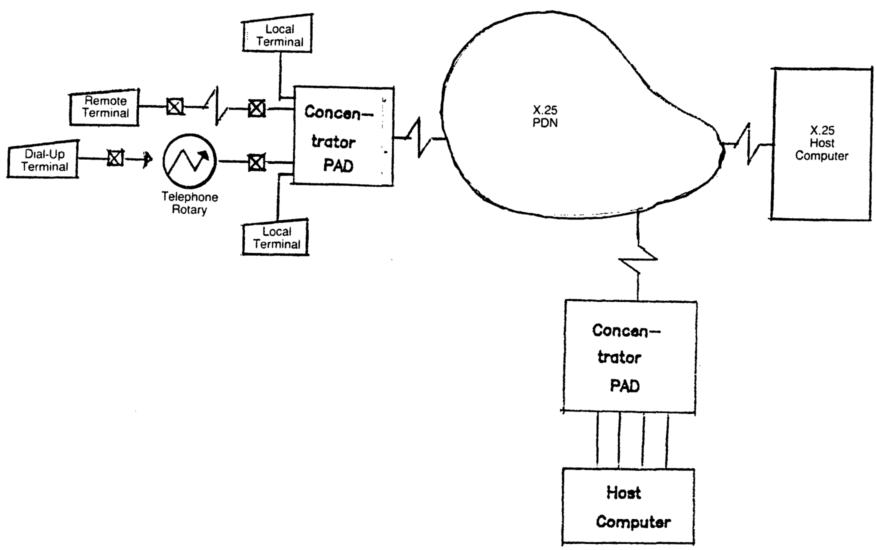
Packet switching is a means of transmitting data from terminal to computer, which uses a shared data network rather than a physical telephone line connection. The data network permits attached terminals to establish connections with other computers or terminals attached to the network and assumes responsibility for "delivering" blocks of data through the network to the correct desination and in the correct sequence. In order to achieve this, the data network assembles groups of data characters received from a terminal into blocks or packets. which are then transmitted through the network properly addressed and numbered, sharing the network's transmission facilities with packets being transmitted between many other terminals and computers.

The Packet Data Network (PDN) was developed in response to the demand for low-cost, reliable data communications. The PDN provides the communications links between computers and terminals through a packet switching concept called 'virtual circuits', which can be viewed as logically maintained connections between two end points of a data network. Virtual circuits can be permanently established between two fixed points, simulating a traditional point-to-point leased telephone line, or temporarily established at the request of an attached terminal or computer, simulating the traditional dial-up telephone line connection. Since the connection between terminal and computer is "virtual", not "physical", the network resource is utilized only when data packets are being transmitted, not permanently. Hence, packet switching charges are normally based on the volume of data transmitted, leading to lower costs in many applications compared to the fixed monthly cost of leased telephone lines.

In the early days of packet switching, each PDN defined a network access protocol to enable any computer equipped with the appropriate software to communicate, through a single physical connection to the PDN, simultaneously with multiple terminals attached to the network. Each of these network access protocols incorporated some type of multiplexing technique, enabling a computer to set up and maintain multiple virtual circuits to other computers and/or terminals. But so long as there was no **industry-standard** network access protocol, computer manufacturers were unwilling to provide the software to handle the various network access protocols. Use of PDNs was restricted to those bold enough to use custom software or those prepared to pay for the much higher cost of one physical connection to the PDN for every computer port.

In 1976, however, the International Consultative Committee for Telegraphy and Telephony (CCITT) adopted a standard network access protocol called Recommendation X.25. This recommendation, along with related recommendations regarding use of asynchronous terminals, enabled all PDNs to offer a standard network interface which could be more readily supported by computer and terminal manufacturers. With the acceptance of Recommendation X.25, packet switching became a viable, economic data communications alternative and public Packet Data Networks were established in many parts of the world.

Fig. 4.



Terminal Concentrator - PAD Application

Recommendation X.25 is now widely supported with software packages available for many different types of computers and has become the world's first industry-standard communications protocol. It provides for low-cost connection of computers to a PDN. And while, in principle, it provides the same possibilities for low-cost PDN access from terminals designed to support X.25, such support is only cost-effective for terminal clusters, not for individual terminals. Hence, for the user of dumb asynchronous terminals, an X.25-compatible cluster controller or data concentrator is a very attractive way to take advantage of packet switching.

CONCENTRATOR PADS

The basic characteristic of packet switching is that data is transmitted through the network in blocks or packets, rather than a character at a time. Packets are entered into the network using the network access protocol, typically X.25. More commonly, in the case of terminals attached to a PDN, assembly of packets is accomplished at the network access point by an element of the network node or a separate communications processor. This element of the network is called a PAD (Packet Assembler/Disassembler) and operates bidirectionally, assembling packets coming from the terminal for transmission on the network and disassembling packets coming from the network to the terminal.

The user of an asynchronous terminal pays for the PAD function as part of his network usage charges. For the user with two or more terminals, an X.25 PAD remotely located from the network access point becomes extremely cost-effective, serving the dual role of packet assembler/disassembler **and** data concentrator, permitting multiple terminals to share a single telephone line to the network as well as sharing a single network access point.

The X.25 PAD is a very cost-effective solution for connecting a small cluster of asynchronous terminals to a PDN. It is equally attractive for connecting a minicomputer which is not equipped with X.25

protocol software. **But one word of caution...**

Computer control over terminal behaviour, such as cursor positioning and character echo, must in many cases be emulated at the network access point in order to avoid transmission delays though the PDN which would change the "feel" of the terminal. Proper implementation of the PAD and configuration flexibility to fit existing systems is vital to the successful usage of a PDN. Any X.25 PAD must include all the flexibility and compatibility offered by X.25's related recommendations: X.3, X.28, and X.29, which define the interactive Terminal Interface (IT) between the PAD device and the attached terminal equipment. Many applications will require additional terminal handling features, beyond these recommendations, in order to emulate fully the normal direct interaction between terminal and computer.

One last word about X.25 PADs and CCITT Recommendations:

The CCITT Recommendations are broad and complicated to the typical user who is normally only interested in a tiny subset of the committee-generated recommendations. However, a properly designed X.25 PAD can be simple to use from the terminal user's viewpoint and easy to configure, manage, and control from the PDN viewpoint.

TRANSPARENT DIGITAL SERVICES

Packet switch networks are ideally suited to the transfer of small volumes of data to a number of differing locations. Where larger or continuous data transmission is required a digital equivalent to analogue leased lines is required. PTT's around the world are now responding to this demand by the provision of point-to-point end-to-end digital pipes. Data rates catered for vary ffrom 4800 bps through to 2048Mbps.

In the U.S. AT and T have offered digital service (DDS) for five years. Data rates offered range from 9600 bps to 56Kbps.

In the U.K. Xstream Services were initiated by British Telecom in 1983. Interfaces are available for 2400, 4800 and 9600 bps providing CCITT V.24 capability. Alternatively 2400, 4800, 9600, 48000 and 64000 bps is provided with an X.21 interface.

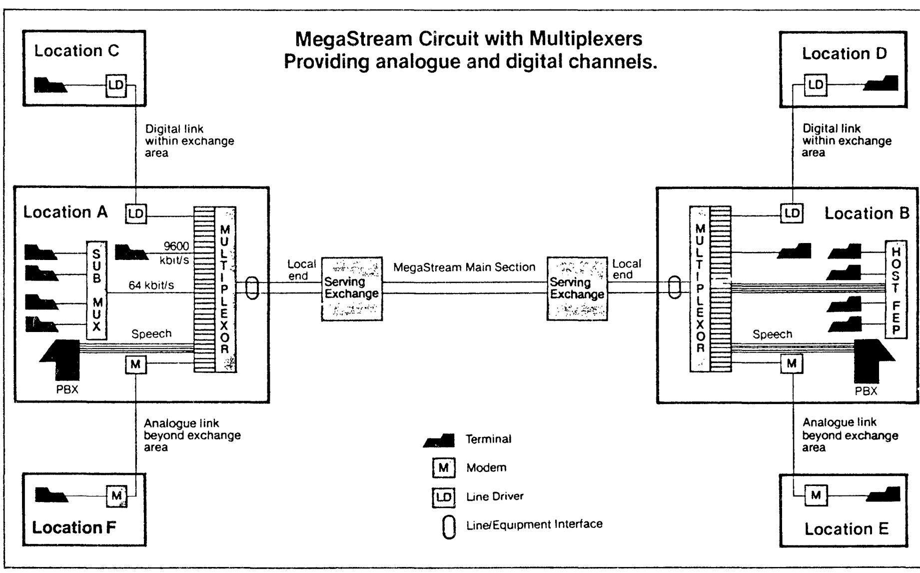
More recently BT have implemented a 2.048 Mbps service named "Megastream". Megastream offers a solution for large business users by providing a range of communications facilities such as:

Conversational data transmission

Voice PABX interconnection

Combination multiplexed voice/data transmission.

Fig. 5.



BRIDGING THE PROTOCAL GAP

As mentioned earlier two main protocols dominate the communications world. Traditionally the fragmentation of asynchronous transmission and protocols used by IBM or IBM compatible systems has necessitated totally separate networking requirements for major users. Such users would often use IBM mainframes and minicomputers for their corporate computing requirements.

Recently a number of gateway products have been developed bridging the gap between the two distinct environments.

WHAT IS A PROTOCOL CONVERTER

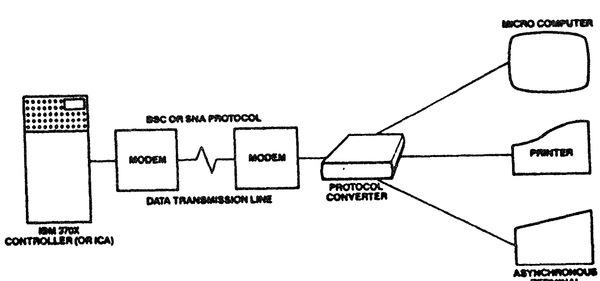
A Protocol Converter provides users of low cost asynchronous terminals, minicomputers and microcomputers with an efficient and reliable gateway into the IBM 3270 environment. It permits access to IBM or IBM compatible hosts from a wide variety of devices wihout any changes to existing hardware or software.

The Protocol Converter connects into an IBM 370X or any compatible communications controller using BSC or SNA protocols at line speeds up to 19,200 bps. Up to twelve asynchronous devices may be connected into a single unit.

The low cost of Protocol Converters permits 3270 users to achieve considerable cost savings compared with the cost of similar IBM configurations.

Not only do Protocol Converters and the attached devices cost less than comparable IBM or equivalent products, but they can use asynchronous modems and dial-up lines which are less expensive than the synchronous modems and leased lines required by IBM systems.

Fig. 6.



Typical Protocol Converter Application.

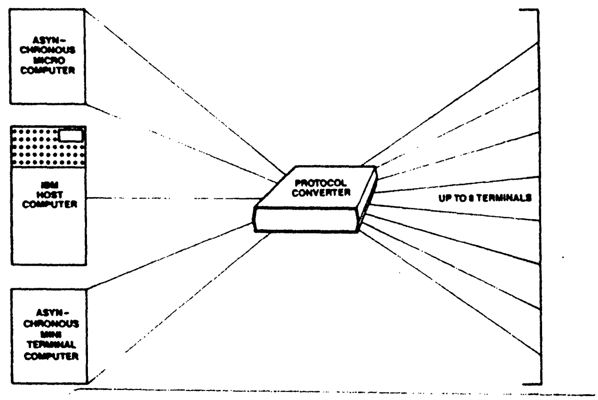
MULTIPLE HOST ACCESS

It is not uncommon for organisations to utilize both IBM mainframes and minicomputer systems for their corporate requirements.

Terminals that communicate with minicomputers are normally asynchronous.

With the adoption of a Protocol Converter, these asynchronous terminals can switch to either the minicomputer host or be clustered into an IBM 3270 host resource.

Fig. 7



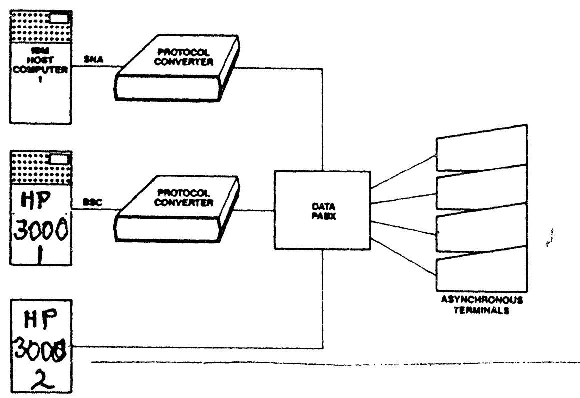
Inherent Switching Capability of Protocol Converters

It is therefore no longer necessary to dedicate a terminal population to a particular host computer, hence a significant reduction in terminals results.

In larger networks the effectiveness of the IBM gateway is dramatically multiplied when a Protocol Converter is used in conjunction with a data PABX. Here, terminal users are provided access to asynchronous hosts and also to IBM or IBM compatible hosts through the combination of port selection and contention facilities furnished by the data PABX.

The data PABX provides contention, host switching selection and network management for an asynchronous terminal/microcomputer network. Configuration of the data PABX is effected using a command port and most devices are modular and capable of handling up to 1500 line or port interfaces.

Fig 8.



Terminal Resource Sharing Using a Data PABX

<u>CONCLUSION</u>

New technology and innovation in data communication is progressing at a rapid pace. PTT's and Telecommunications administrations are implementing new services and alternate tariffs aimed at exploiting such advances. The wise data processing organisation should focus considerable emphasis on understanding services, trends and benefits in the data communications arena.

BIOGRAPHY

KEN BAYNTON

is the Managing Director of MICOM-Borer Ltd. in Reading, England with responsibility for all operations in the U.K. Ken has been with MICOM for seven years working in both the U.K. and the U.S. He has been closely involved in the market definition, development and manufacture of a wide range of data communications products, including modems, concentrators and data PABXs.

FROM TERMINAL TO COMPUTER PORT IN 232 EASY STEPS.

by Anton J.W. Buiteweg
RAET cv ARNHEM.

ABSTRACT

When a computer has been bought the next problem to occur, and one for which HP did not provide a real solution for a long time, is someone wanting to connect a terminal located more than 15 metres away from the computer.

Our experiences on this subject started in 1978 when our Research Department moved out from our headoffice into a "Portacabin" some 500 metres away from the computer and still wanted to continue using its four terminals on our HP3000 series II.

In 1980, we connected our first remote location Delft to the network and in 1983 we crossed The Channel.

Nowadays we run a rather extensive network in which two HP3000's and two service bureaux can be accessed from about 200 terminals and microcomputers that are placed in two main offices some twelve hundred metres away from each other and in various locations in our country.

How we managed and what we suffered will be described in this paper, which will also contain a number of tips and techniques on how to detect and avoid the various datacommunication pitfalls.

1. INTRODUCTION.

Although nobody likes it very much one will be involved with data communication problems sooner or later. As a matter of fact it will be the next problem to occur after the computer has been bought. There is no possibility to relax after the heavy work of selection, preparation, installation and configuration of a computer system.
On a rainy Friday afternoon when you are thinking that all is under control some fellow (or girl) walks into your office with the message that he (she) wants to connect an extra terminal to this beautiful computer of yours.
That's okay because you have been clever enough to order additional ADCC or ATP ports, but.........he (she) wants to connect a terminal that will be located in the other building. And there it is, your first communication problem.

How to get from this terminal to a computer port will be described in the next 232 easy steps.

2. THE HISTORICAL STEPS WITH ROYAL BOSKALIS WESTMINSTER.

The first steps.

With Royal Boskalis Westminster our first step "into data communications" was made when our Research Department moved from the headoffice into a so called "Portacabin" in 1978. The portacabins had been erected some 500 metres away from the headoffice and were needed as temporary offices of the main office because of the growth of the company. The portacabins were expected to be in use for at least two years.

At the time when the Research Department moved out they used two terminal placed in a kind of terminal room. As the terminals were used intensively they wanted the possibility to connect at least a third terminal in their new temporary location.

Of course we knew from HP that terminal connections over a distance like this were not supported and we contacted some manufacturers of communication equipment. After some talks and considerations it nevertheless was decided to try to lengthen the terminal cables anyway. Two kilometres of terminal cable was ordered and delivered. The cable was cut into four equal parts and installed along the fence that surrounded our yard. Doing this it became clear that someone had miscalculated the length needed and it became necessary to add another 10 metres to each of the cables.

To our relief it all worked fine and the cables along the fence never failed although the fence was damaged several times during the more than two years this connection was operational.

Encouraged by this success the department decided on a lovely layout for their terminal room in which the HP2631A printer (having a parallel interface) connected to a HP2648 terminal had been placed too far away from each other to use the standard printer cable. The newly made cable created a rather strange printout.

The standard cable had a length of about 3 metres and the new cable a length of 10 metres. Since the minimum length needed was only 7 metres we decided to shorten the cable as much as possible and were successful in getting the printer to do its job without errors.

The conclusion was that, although not supported by HP, serial datacommunication lines could be lengthened to many hundreds of metres under some conditions (noise!) and that the maximum length of parallel interface cables is rather critical.

Apart from this "remote" location in the Portacabins all other terminals were fairly close to the computer room located on the ground floor of the head office. Since the maximum number that could be connected (not speaking of being simultaneously used) to our series II was rather limited, all terminals had been placed in terminal rooms to be able to satisfy the needs of many people. Agenda's were used to reserve time on a terminal.

Although cabling to these rooms could be simple it was expensive and time consuming to add an extra terminal cable when needed. This reason and the need to bring the terminals to the desks of the end-users, which is a must if you want to go into office automation, made us reconsider how a datanet structure could be built up in the new head office in the 1980-1981 timeframe.

Step 5 to 28.

For the new head office datanet we again contacted our suppliers. Although people talked a lot about local area networks we could not find one operational in Europe. Our current communication supplier advised us to develop a four wire based star network for the following reasons:

= Cheap cables (normal telephone cord can be used)
= Easy to install by any electrician.
= Flexible
= Well known
= It works!!

Royal Boskalis Westminster is a well-known worldwide contractor and -to its staff- also well-known for the amount of internal moving of people and offices; so we wanted a datanet that would meet the following additional requirements:

= Possibility to change the network at limited costs.
= Possibility to connect up to 300 terminals
= Possibility to connect more than one computer.
= Possibility to connect service bureaux.
= Possibility to connect all remote locations in the various parts of our country by means of dial-up lines or hired fixed lines.
= Tools to manage the network.
= Resource sharing in the way that on average about 3 terminals share one computerport. (The expected usage per decentralized terminal would be appr. 30 %.)

Connecting that many terminals to our series II was impossible but a second computer was needed anyway. A series 44 was ordered from the Dutch OEM RAET and the series II was given an upgraded to a series III.

We finally choose for a portselector to control the network for the following reasons:

= Connection of that many terminal possible without the need to invest in expensive extra ADCC or ATP's and extra cardcages.
= Connection from any terminal to more than one computer or service bureau is possible without the need of extra cables to these terminals.
= Built-in linedrivers making extra linedrivers on the computer side superfluous.

= Cheap extension possibilities.
= Resource sharing possible.
= After connection the datatransmission will be transparent to the user, the terminal and the computer equipment.

The new head office of Boskalis consists of four buildings that are interconnected, each building having four floors. The so called A building has been used for meeting rooms, the computers are located in the so-called D building on the fourth floor.

To get a flexible network layout we decided on two datadistribution cabinets on each floor of each building, one on each side. From each cabinet a maximum of ten terminals can be connected which connection takes only a few hours of cabling work and a maximum cabling distance of 25 metres. Of the total network about 35% of all possible connections were made during the construction of the building.

Each time another connection was needed and each time someone using a terminal moved from one office to another we created a new connection without disconnecting the existing one.

All connections run through a datadistribution cabinet in the cellar and from there to the cabinets located in the computer room. Even the operators terminal, only some 10 metres away from the computers uses this way through the datacabinet on the fourth floor down to the cellar, up into the computer room. The only exceptions are the consoles of both computers that are connected directly.

Using this datanet concept does not necessarily mean that having a datanet connector means having access to the datanet. It only means that there will be a connector with the same number in the computers room cabinets that can be connected to the portselector. This connection can be made by simply plugging in a connector into the connection box.

The portselector (MICOM600) has been equiped with a total of 100 ports (leading to the computers) and 268 lineinputs (leading to the terminals). This dataswitch can be extended by blocks of four ports or lineinputs residing on one card. There is no difference between lineinputs and ports except for the thumb wheel setting on the card. Every time the number of ports and cards exceeds a multiple of 120 an extra cardbay is needed. Four cardbays fit into one 19" rack. The maximum number of cardbays is 12 making the potential number of ports and lineinputs 1504.

In the area where the head office of Royal Boskalis Westminster is located power failures of tenth of seconds up to some hours are rather common. Especially the short power failures sometimes not even monitored by the HP3000 computers occur regulary at an average of one failure each two month. In the last three years we have had three power failures that lasted for several hours. Power failures normally are handled correctly by the HP3000 computers unless they exceed the memory back-up battery capacity. The portselector, which in fact is the most vital part of the total network, has been equiped with double power supplies and fully independent double logic. A battery back-up can be installed on the memory but this back-up does not keep the connections alive, which makes the use of this battery back-up limited.

The short power failures mentioned above have a rather strange effect on the portselector in the way that lines and ports are randomly connected without a loss of memory. Users suddenly find themselves working in each others accounts which destroys in fact all HP3000 securities.

This security problem is the main reason why the portselector has been equiped with a no-break power supply providing a mains back-up for appr. 20 minutes without any interruption.

Since the power supply has been installed we suffered several shorter power failures that did not influence the portselector at all and one failure that exceeded this 20 minutes.

Providing for a real no-break power supply for all equipment for many hours, which would be the best solution, proved to be too expensive compared with this smaller power supply for the portselector only, which cost about dfl. 15,000.

STEP 29 to 80

The next important steps we took in datacommunication was to connect our first real remote location. Because we wanted to continue the concept that every terminal should have access to both computers and service bureaux we did not want to use the newly introduced HP clustercontroler involving an INP. We also found out that many remote sessions put a heavy load on the DSline between both HP3000 computers, in fact too heavy to get an acceptable performance.

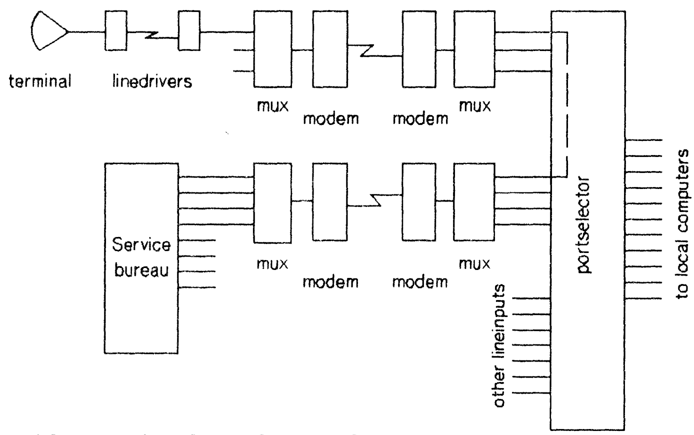
Finally, statistical multiplexers connected via modems to hired fixed telephone lines were chosen. We opted for multiplexers of the same make as our portselector because we were happy with the reliability of the system and the support provided by the Dutch representative. We ordered a telephone line of normal quality from the PTT and ordered an upgrade to extra quality some weeks later because this procedure would speed up delivery.

Almost at the same time a connection of this kind was made with Boskalis daughter NACAP, with offices in Delft and with the service bureau CDC in Rijswijk. Modems with a speed of 4800 bps have been used on the line to CDC and modems of 9600 bps on the line to NACAP. Multiplexers of 4 and 8 channels respectively have been used. The connection to NACAP was upgraded to 12 channels at a later time. The modems to the service bureau were equiped with an extra synchronous port to connect our DATAPOINT data entry machine to CDC as well.

Both installations have been done in close co-operation with the Dutch representative of the communication equipment.

A special arrangement proved to be necessary on the CDC side of the line to keep the four ports alive. The connection to NACAP gave no problems.

In the Delft office of NACAP a similar internal datanet as the one in the Sliedrecht office was built. Each of the three floors of the building in use by NACAP was equiped with a data distribution cabinet. On one floor at the location where the hired telephone line entered the building the modem, the multiplexer and a total of eight linedrivers were installed. The linedrivers were needed because the concentrators do not have built-in linedrivers like the portselector.



COMMUNICATION FROM REMOTE SITE TO SERVICE BUREAU.

Figure 1.

People working in the Delft office on the service bureau in Rijswijk had to pass through 11 boxes of datacommunication equipment before reaching CDC. How we kept this alive will be discussed in a later chapter.

One extra advantage of an internal datanet like this is that it is quite simple to install an application or development machine at this remote location when the workload on the central machines justifies the investment. The computer can be placed near the existing modem and concentrator equipment. A small dataswitch will make access possible to this machine and to the rest of the datanet on which other applications can be run e.g. an application that is to large for the local machine (see also figure 2).

In this case the telephone line can be used for two purposes: connection to the rest of the network and access to the local computer from the headoffice for system management.

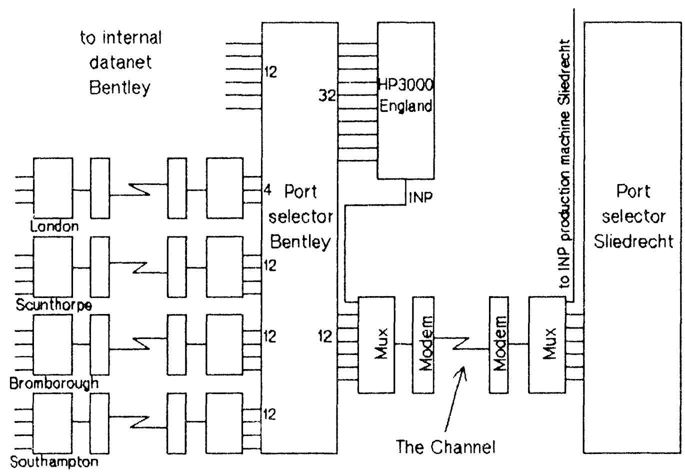
A few months later the location Dordrecht where the Construction Division resides was connected. The telephone line which distance showed to be less than 5 kilometres, could run at a speed of minimum 9600 bpi without problems. Sixteen terminals have been in use on this line. After the first successful tests the line speed was increased to 19.200 bpi which again gave no problems. On the

Dordrecht side of the line a linedriver rack containing 16 linedrivers was installed instead of 16 separate linedrivers. The advantage of separate linedrivers is the flexibility (the linedrivers can also be used on the terminal side whenever the telephone line connection would be ended); the advantage of a linedriver rack is the price.

STEP 81 TO 200

In 1983 we finally crossed The Channel. When connecting remote locations in other countries to a datanet the extra problems (apart from the high hire rates for the telephone line) are that you are obliged to deal with two suppliers of telephone lines (in the Boskalis case the Dutch PTT and British Telecom) and with two representatives of the communication equipment. For the telephone lines the advise should be: ORDER IN TIME!; for the communication equipment: be sure to get the same versions of the software in your equipment!

It was intended to connect several remote locations in England through this telephone line to the datanet. The datanet would include an HP3000 in England as well. The proposed layout is displayed in the next figure.



COMMUNICATION WITH ENGLAND.

Figure 2.

We never got that far. The line across the channel had been operational for only one year when the major activities of Boskalis in England were ended including the section of the Informations Systems Department that was located in England.

Over this line even an HPWORD session was operational for the same period. Just imagine: a wordprocessor terminal in England doing wordprocessing on a computer located in Holland and printing the document on a HPWORD printer located in England......and it worked fine.

The same printer was in use to print messages from Holland to this section in England and messages could be send back by accessing one of the HPWORD printers in Holland or by accessing the system pageprinter.

A complication in running this concept was the time difference between Holland and England; e.g. messages about back-up time and closing down time were sometimes confusing.

In April 1983 the total Information network looked as shown below:

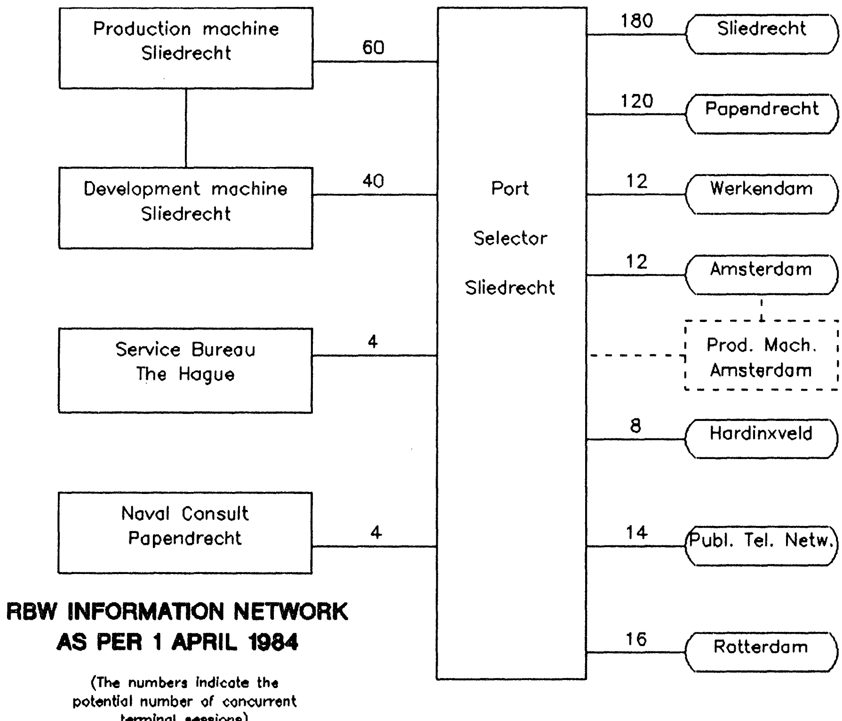


Figure 3.

THE PRESENT AND FINAL STEPS. (STEP 201-232)

At Present, (December 1984) Boskalis is negotiating with RAET (my present employer) about taking over this network and integrating it into the RAET Information Network.
The advantages are clear:

* Interfacing to larger machines (IBM mainframes) possible.
* Better service level in manpower and specialism due to the fact that RAET as a large automation company can employ almost all specialisms needed.

* Support outside office-hours possible without extreme costs.
* Back-up computers and man-power can be made available in case of calamities.
* No need to extend the Boskalis Network to meet peak hours requirements.

SOME INTERMEDIATE STEPS

In describing the steps sofar many smaller steps were omitted. Three of them will be discussed now:

1. Interfacing to "strange" equipment.

Every company produces an annual report. Boskalis produces its annual report in an English and a Dutch version. Using wordprocessing software with which Boskalis started in 1976 the time-consuming effort of correcting and retyping decreased largely. In 1982, Boskalis started using HPWORD (as the first company in Holland?). Documents created by this package and printed on the laser printer (HP2680A) are used as hand-outs at press conferences where the annual report is presented. An official copy is produced at a later time and printed with pictures in full colour by a professional Printer. In the beginning this Printer retyped the text from the documents, carefully prepared by the Boskalis Financial department responsible for the annual report. It is quite clear that this method is bound to introduce mistakes. Using this method the time-consuming work of checking and correcting starts all over again together with the discussions about which pictures should be used and the layout of the report.

The Printer bought a black box consisting of a modem, an interface, floppy disc drive and software making it possible to receive data send over telephone lines into his type setting machine. The only thing that would still have to be done was to add the special formatting commands to the documents received and to include the pictures. The total system was built to receive data from Wang wordprocessors, but how to get the annual report documents into this machine and what would be the best format to send it over? Spoolformat? Ascii file with or without CCTL? How to get the HP3000 to send data as a Wang? How to solve the handshaking problem? Even the "standard" V22 modem both of the same brand proved to be unable to talk to each other due to strapping differences and on the day it was all to take place the black box broke down and had to be repaired by the manufacturer. An 8 year old 300 baud modem finally proved to be the simplest and best working solution: dial up the 300 baud modem at the computer side (have you ever heard so much noise on a telephone line?), log on to the computer, use the file copier to print the file to the terminal and quickly switch over from terminal to interface.... and it worked!

2. Interfacing "strange" printers.

The number of printers available from HP was rather limited in 1983. The thermal printer was too slow and used unworkable thermal paper, the 2631 (or 2635) printer was too expensive.

Because Boskalis needed a number of matrix printers with a speed of appr. 200 characters per second a review of the marked supply of this printers was made.

The price which could be paid was appr. dfl. 6000.= a price in which the newly introduced HP printer type 2932 nicely fits. Although we preferred to have as few suppliers as possible (for obvious reasons), we started a selection procedure.

It was remarkable to note how many printers using the "standard" RS232C interface and Xon Xoff protocol did not print well or had to be considered insufficiency robust.

The printer needed had to be connected to an HP125 microcomputer and had to work properly in off-line (talking to the micro) and on-line (talking to the HP3000 through this micro) applications.

The protocol supported by many printers is the Xon Xoff protocol, but since the printer is slow compared to the datatransmission between the micro and the HP3000 the following problem may arise:
The printer sends an Xoff to the micro when its buffer is about 80 % full; this will stop transmission from the micro to the printer but it will not stop the transmission from the HP3000 to the micro until the next enquiry (ENQ) from the HP3000 to which the micro will not respond with an acknowledgement (ACK). But.... by using the normal termtype type 10 the computer will, after some delay, send a next block of data followed by an ENQ. This data will be lost due to buffer overflow if the printer is not ready to receive data again.

This problem may be solved in many ways e.g.:

* Decreasing!! the buffer size in the printer so that an Xon will be send from the printer to the micro before the HP3000 decides to send more data (the small buffer will be emptied fast enough).
* Decreasing the speed of the communication between the micro and the computer to e.g. 1200 bps, so that the printer can keep up with the speed. This does not work properly when many blank lines or blank pages have to be skipped.

Both methods will decrease the throughput to the printer. The second method even decreases the communication speed when the printer is not in use.

* Using Xon Xoff handshake to the computer. This method has the disadvantage that the terminal will not respond to some status requests from the computer such as a terminal type request.
* Using termtype 16 when logging on. This is the best solution. Termtype 16 prevents the computer from sending data unless an ACK has been received; the computer keeps on sending ENQ's at a rate of appr. one every 20 seconds until a ACK has been received without sending data.

3. Some final remarks on multiplexers:

To prevent data losses multiplexers normally use buffers with a control mechanism that will stop computers or terminals from sending extra data when the buffers are almost filled. Normally the multiplexers use an Xon Xoff protocol to prevent this overflow. When working like this in full duplex over a telephone line a character typed on the terminal will go into the terminal buffer, into the line buffer, be sent to the linebuffer at the other end of the line, into the computer buffer, into the computer which, in its turn will echo the character back into the computer buffer, into the line buffer, into the linebuffer at the other end of the line, into the terminal buffer and finally into the terminal. As can be expected a character typed on the terminal will be visible on the terminal some tenth of a second later. The same holds true if a computer sends a listing to this terminal and has to wait for the ACK before the next 80 bytes can be sent; the listing will slow down severely. The MICOM multiplexers therefore, can be equiped with a so called HP option that sends the ENQ and ACK characters from the local buffers to the terminal or computer respectively as long as there is space in the buffers. This local protocol between the computer and the multiplexer on the computer's side and the protocol between the terminal and his multiplexer will speed up datatransfer in a significant way. But this option cannot be used when binary data which may contain this ENQ or ACK characters as data have to be transmitted. The multiplexer will respond to these characters instead of sending them to the other side. An example of a kind of binary transfer is being used by the HPWORD package. It is therefore impossible to use the HP option with this wordprocessing package. In fact, running of HPWORD over multiplexers will only work as long as no buffer overflow can occur because neither the computer nor the HPWORD terminal will respond to a Xoff sent by the multiplexer. In our experience with this type of multiplexers we never had an HPWORD abort as a result of a buffer overflow as long as no more than three HPWORD sessions together with 9 "normal" sessions were run over a 9600 baud line but there is no guarantee!

3. TOOLS.

The network described in the previous pages requires manpower and tools to manage it. In this part of the paper some of these tools and some of the procedures will be discussed.

Unless you want to try the almost impossible a datascoop is not really needed. A datascoop requires qualified people to use it and it can always be "rented" from HP together with personnel, if required.

A datascoop as monitor can be a very useful piece of equipment when intermittent problems are reported but must include a recording medium so that a problem can be "played back".

The command console of the portselector can be used to change the configuration (temporarily), to monitor ports and lines, to enable and disable ports and lines, to change inclusion and exclusion tables, to set no activity time-outs, to start and stop logging, to enable and disable matrix switching, etc. We connected the command console to a port of the portselector making it possible to access this console from the terminal on your desk. To be able to start the portselector after a power failure, exceeding 20 minutes, the command console can be switched over to a terminal in the computerroom. A start-up procedure is necessary because the default speeds of lines and ports are set to fixed speeds of 1200 and 2400 baud respectively when the portselector is powered up. This configuration has been adopted to keep everyone out after a major power failure.

The multiplexers have command ports as well making it possible to change configuration, to send messages to the remote side of the line and to produce statistics on line quality, buffer usage, etc. The command ports have been connected to ports of the portselector also, making it possible to access the local multiplexer as well as the remote multiplexer from the terminal on your desk. Of course access to these ports should be limited to a small number of lines defined in the exclusion tables of the portselector.

A very useful option in this portselector is the possibility to perform a so-called matrix switching, which means that two input lines can be interconnected. By doing so with the terminal on your desk it is possible to "talk" to an other terminal; everything typed on your terminal normally must appear on the terminal at the other end and all information typed there must appear on your terminal. With this tool the connection lines from this remote terminal to the portselector can be tested.

The type of multiplexers in use have the possibility to test datatransmission from any terminal connected to it without interfering with communications on other lines of this multiplexer. Communication can be tested from a terminal to the local multiplexer (to check the internal datanet) and to the remote multiplexer which test will include the hired telephone line. By means of this simple tool it is possible to find out where communication is blocked.

Another useful tool is the so called monitor mode. The monitor of this type of portselector consists of two ports between which a datascoop can be connected. In the Boskalis case an interconnection cable between these ports has been made with a tap on pin 3. The tap again leads to a port of the portselector. The monitor option can be used by connecting a terminal to this port in the same way as a connection to a computer would be made. The operation is simple: give the monitor command on the console of the portselector and you can "spy" on all data send from the computer to the connected terminal. It works as a simple and cheap "datascoop". In full duplex mode the data from the computer to a terminal includes the commands send by the terminal itself. It is also possible to set a tap on pin 2 in which case only data sent from the terminal to the computer will be monitored. The monitoring function can be used to find out if problems are occuring in the transmission between a terminal and a computer port. After going into monitor mode the running session will remain transparent; at the moment of going into monitor mode the transmission between the terminal and the computer port will be interrupted for some tenth of a second. Monitoring should therefore be done with care.

The best "tool" will be a procedure for trouble shooting when a problem is reported. It is very useful to have detailed trouble shooting lists for all terminal types and all types of communication (direct, dial-up, leased fixed lines) and to keep this list up to date.

The problem reported can be anything from a problem reported by an experienced end-user to just a simple "it does not work" when an inexperienced end-user reports the problem.

= First of all be sure that a real communication problem is involved. I myself spend much time in tracing a reported problem when the only thing the end-user had to do was to wait for the completion of a programme which this time for some reason took longer in execution than the other day.

= Always talk to the end-user directly; he knows the last thing that happened and he is the only one who can answer your "strange" questions.

= Be informed of all changes in configuration, versions, software updates etc. be informed of the impact that these changes may have on the "contact" between the end-user and the software he uses.

= Find out when the problem occurred, during a programme run or first thing in the morning when the terminal was switched on.

= Do the trouble shooting by elimination of the working parts of communication.

= The way to deal with these problems depends on the kind of problem reported. When the user can not reach the computer start with the terminal side when you are sure that the computer is up and running.

= Be sure that the terminal is switched on and that there is power. Check all cabling from the terminal to the dataconnector. Check power on the linedriver. Check that the linedriver is set to remote. Check the terminal configuration starting with checking that remote mode is on and that auto line feed, modify line, modify all, block mode are switched off. When these checks must be performed by the end-user himself, talking to you over the telephone, it is necessary to have the terminal manual available to be able to tell exactly which keys should be pressed and in what sequence.

= Be sure that the end-user did not move his terminal to another room without switching over the dataconnection on the computer side.

= When dealing with remote sites using dial-up lines, be sure that the right telephone number and the right baud rate has been used. Boskalis uses three different types of modem sets: one standard 300 baud type, one non-standard 1200 baud type and one standard 1200 baud type. End-users visiting another remote location might dial up the number they are used to but this can be the wrong number. I once talked to an end-user who complained that no welcome message appeared when hitting return. He did not know that on this location he had to dial-up the computer first.

= When dealing with remote sites using hired fixed telephone lines, the testing can be continued by performing tests to the local and remote dataconcentrator from the end-users terminal to test the local datanet, the concentrator lineinputs and the telephone line. Be sure that the modem and concentrator on the end-user side have not been switched off.

= Using the matrix switching option it is possible to test whether perhaps communication is possible in one way only, which can be the case when an input module of the portselector has failed. With this option it is also possible to check on datacorruption.

= Intermittent problems can best be checked by using the monitor option or more simple by replacing the communication equipment one piece at a time to find the piece of equipment that causes the problem.

4. EVALUATION.

At this moment when even HP will introduce two different types of local area networks soon it is worth while to evaluate the concept chosen.

At the time when Boskalis started building up its network no local area networks were operational and at present there still is no need to switch over to a more "modem" version of communication.

But what would the network look like if the decision on the type of network was to be taken today?

The length of the cable which may be used in a local area network (LAN) is normally limited to some kilometres. For cabling a large building like the Boskalis head office this length is too small. If the other local area network of the office of the Dredging division should be connected in the same way as it is done now (without using modem equipment) the length of the cabling would certainly exceed the maximum length.

When comparing the cables used it is clear that those used by many LAN's (special coax) are much more expensive then simple four wire cable; even when we compared a star network with a ring network. The costs of cabling may be less but can it be done by any electrician?

Connecting a terminal to the network now takes about two hours work of an electrician when no connector is available in the room. The connection itself, a linedriver on the terminal side and 1/4 lineinput card on the portselector side, will cost appr. dfl. 600,= . These costs compared with the connection costs of a LAN are low.

The possibilities to change access capabilities to special peripherals (such as the command ports of the multiplexers of outgoing lines) are limited when a LAN is used.

The experiences of the past show that the network now in use by Boskalis can be run easily and meets the required flexibility. Also connecting remote locations is easy with the possibility to connect locations over the public datanet (that uses the X25 protocol) when this should be needed in the future.

In the Boskalis case it could be that a conventional four wired star structured datanet would be chosen again.

5. BIOGRAPHY

Anton Buiteweg started his career in 1973 with an engineering company on public address systems. In 1976 he switched over to the Research Department of the contractor Royal Boskalis Westminster and was involved in process automation, measurements and dataprocessing of these measurements. The dataprocessing part of his work made him decide to switch over to the Information Systems Department in 1981, in which function he was responsible for datacommunication and office products. Early in 1984 he succeeded the Information Network Manager and became responsible for the total network, which at that time consisted of two HP3000 computers and communication equipment serving appr. 200 terminals and microcomputers both in Holland and abroad. In November 1984 he became a System Specialist with RAET computer and software services in Arnhem.

THE EVOLVING MICRO-TO-MINI/MAINFRAME INTERFACE

Andree Driskell
Hewlett-Packard, Cupertino, USA

Summary

Personal computer advances over the years are continuing to set new standards in improving the man-machine interface and in bringing computing power to the non-computer initiated. For this advancement to continue however, most organizations realize the need to also provide access to larger computing resources for processing power and existing information bases which are not available on the Personal computer. The key therefore is the continual improvement of the micro-to-mini/mainframe interface so that users can access all computer resources in the same user-friendly manner.

This presentation will discuss the major ideas and strategies proposed to more closely integrate the micro and mini/mainframe. Also discussed are the forthcoming networking technologies which are required to make these ideas and strategies successful. The objective of this presentation is to clearly position the alternatives for the micro-to-mini/mainframe link to assist the attendee in defining their own information network strategy

Introduction

The emergence of personal computers has dramatically impacted today's computing environment. Personal computers are replacing terminals in many work environments. Over the next decade they will penetrate all business environments. The installed base for business personal computers in 1984 was 3.5 million. It is estimated that this number will grow to 9.7 million by the end of '85. The reasons behind this enormous growth are numerous. A focus on improving the quality of worklife, the push to place decision-making further down in the organization, the decreasing price/performance of information processing coupled with the increasing cost of the employee are only three reasons for the popularity of the personal commputer.

The office is a prime example of one environment in which personal computers have made an significant impact. In the past, most of the planning for office automation assumed the use of mainframes or minis serving a large number of terminal users. Today, personal computers are playing a major role in the office. Companies are now seeking the best method of allowing communication between personal computers, and between personal computers and the organizations's mini and mainframe systems, along with the best way of delivering office functions to their users. Althrough we are witnessing this immense growth of the personal computer, central data processing, and the use of department minis will continue to grow. The key therefore, is the continual improvement of the micro-to-mini/mainframe interface so that users can access ALL computer resources.

Standards and Multivendor Networking

One reality computer vendors must deal with is fact that we do not exist in a homogeneous environment. Regardless of whether a network consists of personal computers or a combination of micros, minis, and mainframes, the odds of all these systems being from a single vendor is small. Businesses are demanding the flexibility of choosing products from a variety of vendors, and they expect these products to communicate. Alherence to the emerging international standards is one method of creating an environment for multivendor networking.

The International Standards Organization(ISO) has developed a general model for data communications termed the Open Systems Interconnect. The primary goal of this model is to "define standard protocols and interfaces which enable people, terminals, computers and physical processes to exchange information among themselves without cumbersome intervention mechanisms". ISO's purpose was to create a reference model from which actual standards and protocals evolve. To date, two international standards are based on this model. CCITT X.25 which specifies the interface to a Packet-Switched Network and IEEE 802 which specifies the interface to a Local Area Network.

Another group of emerging standards are the Integrated Services Digital Networks(ISDNs). Today, separate protocals, interfaces, etc. exist for analog and digital communication. The basis behind ISDN is to provide a universal set of protocals and interfaces for both analog and digital transmission. ISDNs will handle simultaneously digitized voice and data transfer on the same digital links and by the same digital exchanges. Users will therefore have access to circuit-switched, packet-switched and other services via the same user interface. Although ISDN standards have not yet been finalized, both the technology and forthcoming implementation strategy are now well understood. CCITT has completed work on the first family of ISDN standards. Over 30 recommendations were formally adopted in 1984 and these will be the basis for the 1st phrase of evolution to worldwide ISDN. It is estimated ISDN will require from 10 to 20 years for full implementation.

Two standards which are hopeful candidates for this 1st phrase of ISDNs are the Digital Multiplexed Interface(DMI) and the Digital Control Protocol(DCP). The initial implementation of these standards will most likely be over Private Branch Exchanges(PBXs) since voice/data integration is already a feature of 3rd and 4th generation PBXs. DMI specifies the use of a 1.544 mb/sec T1 line at speeds up to 64kbps linking up to 23 users. It specifies the connection between mini/mainframe and the PBX. DCP on the other hand specifies a 64kbps link between the workstation and the PBX. DMI-based products are expected to reach the marketplace during the '85 timeframe but DCP-based offerings most likely will not be available until the '86/87 timeframe. Both standards can be migrated to final ISDN specifications once defined.

Shared Resources: Micro to Micro

The need for resource sharing among personal computer users is apparent. The use of local area networks(LANs) is becoming the perferred method of networking micros because it allows this sharing of data and peripherals. With a small number of personal computers the sharing of data by redundant copying of disks may initially be manageable, but as the number of micros grow, this method is likely to become uncontrolled and confusing. Thru the use of a LAN, a single master copy of a commonly used file can be maintained, thus eliminating the need for multiple copies. LANs also allow the cost-effective sharing of peripherals such as printers and plotters. If your need is only to share peripherals however, a LAN may not be the optimal solution for your environment. LANs are most useful in the following environments:

*There is a need to concurrently update the same data files

*Problems with redundant copies of data text or program file exist

A LAN may not meet your expectations however in the following environments:

*You're on a tight budget

*A fast and easy solution is needed

*Your only need is to share printers or access data files in read-only mode

*Security is a major concern

Addressing the issue of security, it should be noted that security facilities are in their infancy for most LANs. File security access schemes are typically tied to user names, passwords, and volumes. Each volume can have one of three security levels associated with it: Private, Public or Shared. In the future, it is desirable to be able to organize users into security levels or defining security at the file level.

The method of interconnecting stations varies widely because of tradeoffs in regards to cost, performance, ease of installation and reliability. No pattern has proved itself superior for all environments. Let discuss three popular micro LAN configuration options:

STAR

This is the simplest configuration. Every users station is connected to a single master station. Good for resource sharing if the central station controls the peripherals. Does not perform well for interactive applications involving communication between workstations. This configuration is primarily implemented today in the form of Private Branch Exchanges.

RING

Stations are connected to a closed loop of cables and in most cases transmit in the same direction. Commonly associated with the token passing access protocal

BUS

Using a straight cable that is used to carry signals between nodes. Cable segments can range from several hundred to several thousand feet. Often possible to connect several segments together for greater length.

The pattern of a few personal computers with a disc and a printer will also persist because it is still the ideal solution for small businesses with a few white collar or clerical workers. One solution for the sharing of resources without a LAN are point-to-point shared resource manager(srm) configurations. A shared resouce manager is a microprocessor-based device that acts as a file server and peripheral manager. It allows individual micros to remain autonomous for processing, but still have file and peripheral sharing capabilities.

Shared Resources: Micro to Mini/Mainframe

A micro network interface to existing minis and mainframes is essential, especially in large companies. Users need economical and convient access to high performance peripherals and storage devices, as well as access to data that may be resident on their company's mini or mainframe systems. Connections are needed to SNA, X.25, PBXs, and Mini/mainframe LANs.

SNA

Although the mainframe no-longer dictates the network as it did in the early 70s, the need for central data processing will continue to grow. Even with the proliferation of micros and department minis, users are demanding access to mainframe data, both interactively and batch. The common solution is through file transfer capabilities. Ideally, the communication link should should require no modification to the mainframe software, yet allow the personal computer user to access the host as a fully featured terminal. This is normally accomplished by software that enables the micro or mini to emulate a host device. In the IBM world the 3270 family of terminals and cluster controllers are commonly emulated. The key to SNA compatibility is the knowledge of what type of SNA devices your micro or mini is capable of emulating. All IBM SNA devices are not created equal. Certain devices have certain capabilities within the SNA network. The extent of these capabilities depend on the class of Physical and Logical units the particular device falls into. Physical units determine the level of communication functions a system can perform. For example, physical units 4 and 5 can perform system functions such as store and forward, data flow control, and session granting. Physical units 2s are cluster controller and Physical unit 1s are terminal nodes. Logical units on the other hand determine the level of end-user functions a device can perform. Logical unit 1 defines a Remote Job Entry batch device, Logical unit 2 defines a Interactive CRT device, Logical unit 3 defines a non-SNA printer, logical unit type 4 specifies a word processing document distribution, and logical unit type 6 defines program to program communication capabilities.

X.25

This standard specifies the interface to a Packet-Switched Network. The popularity of this type of network is increasing due to its many advantages. Briefly, these advantages include an alternate cost structure based on volume of data transmitted versus the distance or connect time; configuration flexibility by allowing users to connect to multiple systems over a single link; overall communication reliability because it allows the recovery of internal link failure without user intervention.
For workstation to system communication, the standard specifies asynchronous character mode communication only. Computer vendors are getting around this obstacle however by modificating certain block mode subsystems so they can operate over X.25 networks. The use of cluster controllers or multiplexers for workstation connections has a number of advantages. Not only do they allow the connection of multiple workstations over a single line for cost savings, in many cases they also eliminate the character mode application limitation since systems are connected RS232 to the multiplexor versus direct network connection. Vendor-developed file transfer programs are also available which allow the transfer of files between micros and minis/mainframes over X.25 networks.

PBX

A local area communication networking alternative is the use of Private Branch Exchanges. 3rd and 4th generation PBXs are optimized for digital as well as analog communications. A PBX is ideal for the office communications since the telephone is already in widespread use in this environment. The use of PBXs offers a number of benefits. It's port contention feature allows the connection of many more workstations than there are ports available on the mini or mainframe. Ease of cabling is realized because in many cases the installed twisted-pair telephone wiring can be used. Flexibility is another advantage since any user connected to the PBX can potentially link to multiple systems. One of the goals in this area is to decrease the cost of communicating. It costs approximately an additional $1000 per link to transmit data in conjunction to voice. There are several ways vendors are decreasing the cost of PBX connections. Placing data mixers into workstations and developing integrated telephone/micro units are two examples. Another is by the implementation of standards such as DMI and DCP. These standards produce cost savings by giving multiple workstations the capability of communicating over a single trunk line to a mini or mainframe system.

MINI/MAINFRAME LANs

Currently there are over 1000 high speed local area networks installed for minis and mainframes as compared to over 15,000 lower cost and in many cases slower speed ones for micros. The costs for high speed LANs are declining significantly as the required hardware is reduced to integrated circuits. Low cost LANs are declining even faster in price. In some environments micro LANs exist for several dozen to several hundred links. Although these LANs have some advantages in standalone environments, invariably a gateway to the mini/mainframe facilities is needed. Perferred configuration methods will have low cost networks in each work group or department, with a higher performance LAN used as a trunk service. This Trunk LAN will connect the various micro LANs as well as allow users on the micro LANs access to systems attached to the high performance LAN. A key direction to insure micro, mini, and mainframe compatibility in the Local Area Network environment is the support of mini/mainframe connections to low cost micro LANs. This is suitable for an environment in which numerous micros exist but only a very few mini/mainframes are present. This allows the mini/mainframe to be optimized as a data base server in the LAN environment. Another alternative for the integration of micros with mini/mainframes is the connection of micros on the mini/mainframe LAN. One issue is the high cost of interfacing micros to a high performance LAN. In many instances it is simply too costly an option. Workstation servers that can be interfaced to the LAN and support either micros or terminals may be the best solution.

Integrated Applications: Micro to Micro

The single most important development in the area of micro to micro networking, will be the support of LANs by independent software suppliers whose software is not designed to work with such a network. Some programs that work well in a single micro user environment will not work in an LAN environment because they are unable to allow LAN I/F software to reside in a portion of its memory. Other programs will not work because they may explicitly refer to A, B, or C disc drives. Since the files they need to access may now be on a LAN server, program instructions fail. In addition, major program rewrites may be required to provide necessary record lockout and recovery facilities for applications created for single user environments.

Integrated Applications: Micro to Mini/mainframe

Today, there are applications that run on mini and mainframe computers such as payroll, accounting, etc., and there are applications that run on the personal computer such as spreadsheet and word processing. Although acceptable, this is not the ideal environment. Users are asking for applications that are not only integrated within a particular environment such as a micro LAN, but also across environments. Of the software architectures which achieve this, the most common ones are as follows:

*Micro software is designed to work with a particular mini/mainframe application. This technique allows extensive data manipulation to occur before the file is sent to the micro, but has the disadvantage of being vendor-application unique.

*Software resides on both the micro and the mini/mainframe with mini/mainframe software reformatting the data before it is presented to the application. Such reformatting allows access to a large number of applications, but requires the additional software to describe which application requires which format.

*Software is resident in the micro which interacts with the mainframe teleprocessing monitors. An example of this technique is is HP's 3278 communications accessory board. It provides formatting of data on the micro and allows access to files through through the mainframe editor.

Whichever architecture is chosen, the bottom line is that users want applications that work together

HP's Personal Productivity Center- One Solution!

The Personal Productivity Center solves users needs in the area of shared resources and integrated applications. It provides access to mini/mainframe resouces such as disc drives, laser printers, graphic plotters, etc. The Personal Productivity Center user can print on any network printing device. The user is able to print from an application or a file. The print output is spooled automatically. This spooling allows for queue changes, multiple copies, and output redirections. The spooler will even direct text to non-graphics printers while the graphics is being spooled to the plotter.

Personal Productivity Center provides for the sharing of disc storage. Not only can files be shared, but accessing the disc is transparent to the micro application. Each user can have a private disc for personal storage, and read access to a public disc for shared information.

In the area of integrated applications, the Personal Productivity Center provides transparent application converters for a number of applications. Personal Productivity Center will let you transparently log on to a HP3000 host, execute a local micro program that initiates an HP 3000 application. The local program than opens communications to the HP3000 which executes a file extract program and downloads the requested data to the personal computer.

The Personal Productivity Center is designed to work in a distributed processing environment, leveraging off HP's strengths in data communications. Personal computers have access to the HP3000, SNA, X.25 and other remote information sources.

Biography

Andree Driskell is currently a Systems Networking Specialist for Hewlett-Packard's newly developed Networks Sales Center. She has over 10 years experience in the Engineering and Marketing fields. Ms. Driskell has a BSEE and a MBA.

Bibliography

Lucier, Eugene, "Putting a Price on File Transfer", Data Communications McGraw-Hill, July 1984

Ferris, David and John Cunningham, "Local Nets for Micros", Datamation Technical Publishing, August 1984

Stallings, William, "The Integrated Services Digital Network", Datamation Technical Publishing, December 1984

Personal Productivity Center Presentation

NETWORK SERVICES/3000 TRANSPORT:

AN INSIDE LOOK

Kevin Faulkner
Hewlett-Packard
Information Networks Division
Cupertino, California

Hewlett-Packard's Network Services (NS) product family provides communication capabilities across a variety of data communications protocols and topologies. NS evolved from HP's efforts to improve the current DS product's usability and feature set. More importantly, NS is the result of HP's decision to offer complete networking solutions across product lines. The use of industry standard protocols, the expandable software architecture, and the support of the IEEE 802.3 link, mark quite an advancement in HP networking technology. HP's commitment to NS is reflected in its planned support across other members of the HP computer line.

Initially, the NS family for the HP 3000 consists of two products: Network Services/3000 (the upper level services) and LAN/3000 Link. Network Services/3000 includes the DS/3000 services. LAN/3000 Link includes LAN hardware, related software, and the Network Transport. This paper gives an architectural and functional definition of the Network Transport.

TRANSPORT OVERVIEW

The Network Transport (NXPORT) provides reliable data transfer for both local (same node) and remote applications. It incorporates industry standard protocols, allowing for flexible and well defined communication. The first release of NXPORT supports HP TCP and IP over the IEEE 802.3 link. HP TCP and IP are based on the DARPA (Defense Advanced Research Projects Agency) standards.

NXPORT was designed with a modular and expandable architecture, as it is intended to be the cornerstone upon which a wide variety of communication protocols and network topologies may be supported. The architecture will enable NXPORT to support emerging industry standard protocols at any level without major redesign.

A modular design can be especially susceptible to performance problems. However, NXPORT retains modularity of protocols and support functions without sacrificing performance by implementing its modules as MPE Ports. MPE Ports provide a message passing system with overhead comparable to that of a procedure call. They allow the NXPORT modules to execute on the caller's stack, and thereby avoid process switching.

Supportability is enhanced by NXPORT's integration with the Network Management software. In addition to standardized logging and tracing functions, Network Mangagment provides the configuration capabilities that allow all major NXPORT parameters and resources, e.g. retranmission intervals, buffer pool sizes, to be optimized to a particular node's requirements.

Figure 1 depicts the NS/3000 architecture. The NXPORT modules are enclosed by dotted lines and are briefly described below.

NS/3000 FIRST RELEASE ARCHITECTURE

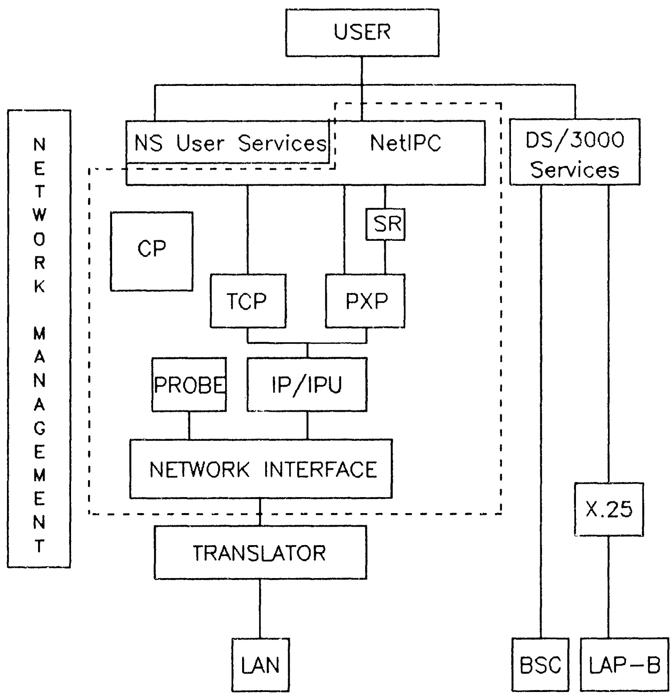


Figure 1. NS/3000 Architecture

CONTROL PROCESS (CP): The control process performs start-up and shutdown functions and processes NXPORT operator commands issued via the NETCONTROL command.

NETWORK INTERPROCESS COMMUNICATION (NetIPC)/SOCKET REGISTRY (SR):

NetIPC provides a generalized programatic user interface to selected NXPORT protocols. For NXPORT's first release, it allows the user to directly access the TCP transport protocol. NetIPC is not a protocol itself. Data packets carry no NetIPC header. The NetIPC service is more flexible than the interprocess communication offered by PTOP, as NetIPC offers peer-to-peer rather than master-to-slave communication. NetIPC provides the only NXPORT interface; the Network Services also use it as their NXPORT interface.

The generic interface offered by NetIPC facilitates application program portability across HP computer lines, and helps minimize the application's dependence on particular traits of an underlying protocol. For example, future releases of NXPORT will likely support a NetIPC interface to protocols such as X.25 and the NBS Transport as well as TCP. Applications written for any of these protocols will require few, if any, changes should they be used over a different protocol.

NetIPC users create sockets through which they can send and receive data. A socket is an endpoint through which data can be transferred. An application must know its correspondent's socket address in order to initiate communication with it, much as a telephone caller must know the callee's phone number before conversation can begin.

The Socket Registry on each node acts much like a phone directory. In conjuction with special NetIPC intrinsics, it allows a local application to "name" it's socket and a remote application to "look up" the socket address based on that name.

TRANSMISSION CONTROL PROTOCOL (TCP): TCP provides reliable, connection-oriented, full-duplex data transfer. Its standard functions include:

- o In-order delivery of data.
- o Fragmentation and reassembly of large user messages.
- o Reliablity through timed retransmission of unacknowledged packets.
- o Per connection data flow control.

In addition, the HP TCP provides several "value added" functions. These functions are implemented in a manner that remains within the bounds of the DARPA TCP standard.

- o User control of windowing values.
- o Optional checksumming. TCP normally checksums the entire contents of each packet. While the checksum provides increased data integrity over multi-hop networks it is not a reliability requirement over a LAN. In addition, checksumming is heavily CPU-intensive, hindering connection throughput. For these reasons, HP TCP allows the checksum mode to be controlled on a per-connection basis.
- o Connection Assurance. HP TCP insures that any idle connection remains up through the timed sending of protocol packets that require the remote side to send an acknowledgement. This is useful for applications that have connections which may often be idle, but require prompt notice of a remote failure.

PACKET EXCHANGE PROTOCOL (PXP): HP PXP is a "reliable" request/reply datagram protocol. It is based on the XEROX Packet Exchange Protocol, and includes enhancements for addressing and error processing. All PXP communication is in the form of requests and replies; a reply serves as the acknowledgement of a request. PXP retransmits unanswered requests much as TCP does. Duplicate requests and replies may therefore occur;

duplicate requests are not detected, but duplicate replies are detected and discarded. PXP's advantage over TCP is that PXP does not require the connection set-up overhead of TCP. However, it is useful only for server-type applications where duplicate requests are not harmful. PXP is used for NetIPC - Socket Registry interactions. Due to its limited utility it is not accessible by user applications.

INTERNET PROTOCOL (IP)/IP UPDATE (IPU): IP provides NXPORT's nodal addressing functions for each packet, and IPU performs the addressing table maintainence. The IP modules have been designed to facilitate the addition of gateway functions in later releases of NXPORT.

NETWORK INTERFACE (NI): The NI provides IP with a generalized interface to the underlying network. It isolates IP from any network type peculiarities such as the fact that an IP internet address must be mapped onto an IEEE 802.3 link address on a LAN network. It also performs the actual "loopback" function for local use of NXPORT. In the future, the NI will facilitate the support of link types other than IEEE 802.3.

PROBE: The PROBE protocol performs mapping functions that greatly simplify the NXPORT configuration task. It sends requests to other nodes in the network to determine node name to IP address mapping and IP address to 802.3 address mapping. These functions are provided on demand, that is, the PROBE is invoked only when a previously un-mapped node name or IP address is used. Once the mapping is made, it is tabled for future reference. Additionally, on network start-up, the PROBE broadcasts a message declaring its node name, IP address, and IEEE 802.3 address. In this way, nodes already started find out the new node's mapping information automatically. The mapping functions provided by the PROBE allow naming and addressing flexibility that will become increasingly important as network sizes increase.

Pictured also are: the NS User Services (upper level software); the DS/3000 software; Network Management; the TRANSLATOR, which provides the interface from the MPE Ports to the I/O System; and the LAN card.

NAMES, ADDRESSES AND PATH REPORTS

Names and addresses are used to denominate objects in the network. The source and destination addresses defined by the Internet Protocol (IP) are the basic nodal address mechanism for HP Network Services. An IP address is a 32-bit quantity which is sub-divided into network number and node number sections. When referred to as a whole it is called an internet address.

A network is a group of nodes (computers) which are attached to the same LAN cable. All nodes of the same network must share the same network number in their IP addresses. Separate LAN cables should use different network numbers since a future release of NXPORT may allow gateways to connect networks. Each node in a network must be assigned a unique node number. Node numbers may be duplicated over different networks.

Although other nodal addressing may be required for a given network type, e.g. IEEE 802.3 level 2 address, or an X.25 address, their use is generally hidden from the user. The IP internet address is the key identifier; it uniquely identifies a node regardless of the type of network on which it resides. To save the user from having to memorize and use internet addresses, HP NS allows nodes to be referred to by ASCII names. The PROBE protocol is used to resolve all node name-to-address bindings.

The internet address allows packets to be delivered to the correct node. Though, once the packet is delivered to the correct node, further addressing decisions must be made. Assuming the packet is a TCP packet, it must be determined which TCP connection the packet belongs to, and ultimately, to which NetIPC socket the packet should be delivered. The TCP header contains the information required to do this. Its source and destination address fields identify the TCP connection the packet is destined for, and the TCP connection corresponds to a unique NetIPC socket. Although this address technically is a part of the TCP protocol, we often refer to it as the socket address. To initiate communication to a socket, its address must be specified. As described earlier, NetIPC and the Socket Registry allow an application to attach an ASCII name to a socket address, enabling the remote correspondent to obtain the address by knowing the name.

The name-to-address bindings, the location, and the protocol definition of a node must be communicated across the network. HP NS uses a generalized structure called a path report to communicate this information. The path report information and format is not specific to the first release of NXPORT. In the future, path reports will play the key role in allowing flexible communication between nodes which may support a wide variety of protocols.

Each node has a global definition, called a global path report, which contains all information about itself which other nodes require in order to communicate with it. This information includes the internet address, any other required addresses, and a list of protocols which are supported by the node. The global path report is returned to any node that issues a PROBE request.

The Socket Registry returns path information specific to a particular socket. This information is referred to as a specific path report. It contains the nodal addressing information, the socket address, plus a protocol path specific to that socket. For a TCP socket, the protocol path would be: IEEE 802.3/IP/TCP.

DATA FLOW EXAMPLE

Following is an example of connection establishment and data transmission for a typical user application. It enumerates the steps as seen from the NetIPC user (see figure 2) and explains the major functions performed by NXPORT at each step.

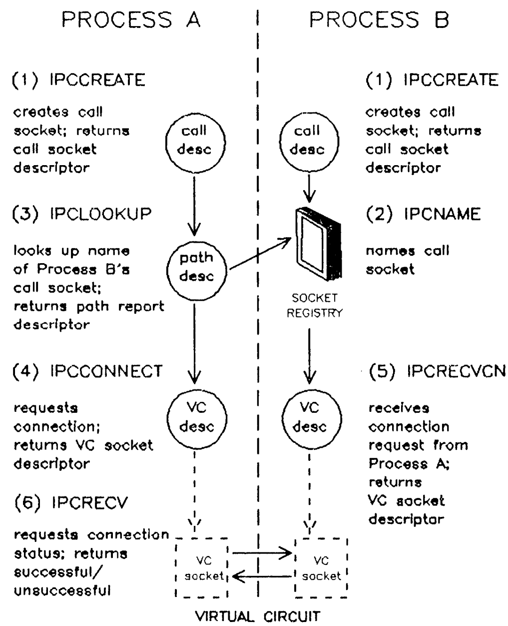


Figure 2. NetIPC Connection Establishment

(1) IPCCreate

Interprocess communication is initiated when Process A and Process B each create a call socket by invoking the NetIPC call IPCCreate. The creation of a call socket initiates the dialogue with NetIPC, who returns a call socket descriptor which must be used in subsequent NetIPC calls relating to this socket. Call socket creation allocates and reserves a TCP address on which a process may receive connection requests.

(2) IPCName

Process B associates a name with its call socket address by calling IPCName. This name-to-address mapping is placed in the Socket Registry at the local node. The name Process B assigns to its call socket must also be known to Process A because Process A must reference it later in its IPCLookUp call. In the same maner, the name of the node on which Process B resides must also be known to Process A. Although call sockets do not have to be named, a process cannot gain access to another's socket if the socket is not named.

(3) IPCLookUp

Process A uses the names assigned to Process B's call socket and node in its IPCLookUp call to "look up" the call socket in the Socket Registry at the node where Process B resides. NetIPC returns a path descriptor which must be specified in the subsequent IPCConnect call.

The mechanism behind the IPCLookUp is as follows:

- o The node name specified in the call must be translated to an internet address (and its corresponding IEEE 802.3 address). If the mapping has not previously been made then a PROBE request is broadcast on the network, requesting that the node specified respond with its global path report. When the response arrives, the mappings are tabled and execution continues.
- o NetIPC uses PXP to send a request to the remote node's Socket Registry.
- o The remote Socket Registry receives the request, performs the look up, and sends a reply to NetIPC. The reply consists of the specific path report to the desired socket.
- o NetIPC receives the reply, stores the information, and returns a path descriptor to the user.

(4) IPCConnect

Process A specifies the path descriptor returned by IPCLookUp and the call socket descriptor returned by IPCCreate in its IPCConnect call. Using these two parameters, NetIPC requests TCP to open a connection between processes A and B. TCP creates a local connection half, and then uses the addressing information in the specific path report to send a connection request (a SYN packet in TCP parlance) to the remote node. In the meantime, IPCConnect returns a virtual circuit (vc) socket descriptor to the user. This descriptor is the local endpoint of the connection and must be specified in the IPCSend and IPCRecv calls issued for this connection. Note that the connection is not yet really "up".

(5) IPCRecvCn

Using the call socket descriptor returned by its IPCCreate call, Process B calls IPCRecvCn to receive any connection requests that have arrived. In this example, Process B will receive a connection request from Process A. IPCRecvCn returns a vc socket descriptor which Process B must use in its IPCSend and IPCRecv calls.

Before the IPCRecvCn completes to the user, TCP creates a local connection half to handle the new connection. After the IPCRecvCn has completed, TCP transmits the connection reply (SYN/ACK) packet.

Note that if the user had not yet called IPCRecvCn, the local TCP would have waited for the user to do so. The remote (Process A's) TCP, however, would have begun the timed retransmission of its SYN packet since it had received no response from Process B. If Process B does not issue an IPCRecvCn within the retransmission time configured, Process A's TCP will abort, returning an error to Process A.

(6) IPCRecv

Process A calls IPCRecv using the vc socket descriptor returned by the IPCConnect call. The first IPCRecv call following an IPCConnect call returns the status of the connection (successful/unsuccessful). If the status is successful, Processes A and B can now "converse" using the newly formed connection.

Upon receiving the connection reply, TCP transmits an acknowledgement (ACK) packet, thereby completing the TCP connection establishment handshake. It then completes the user's IPCRecv call. At this point the connection is fully operational.

IPCSend

When a process calls IPCSend, TCP performs the following steps:

- o While there is more user data and TCP send window available it:
  - o Moves 1 fragment's worth of user data to a transport buffer.
  - o Adds the TCP header.
  - o Adds the packet to the retransmission queue.
  - o Sends the packet to IP.
- o When all the data has been sent, it completes the IPCSend to the user.
- o Note that IPCSend completion does not imply that the data has been safely delivered to the remote.
- o When the acknowledgement of each packet arrives, it dequeues the packet from the retransmission queue and returns the buffer to the free pool.

IPCRecv

When TCP receives data, it performs the following steps:

- o Adds the data to the TCP receive queue, reassembling if necessary.
- o Sends an acknowledgement to the remote.
- o Delivers the data to the IPC user.
- o When the IPCRecv is complete, it returns the data buffer to the free pool and sends a window update to the remote.
- o Note that an IPCRecv is not required for TCP to receive data, but it is required for TCP to deliver the data to the user.
- o TCP minimizes the number of protocol packets transmitted by acknowledging multiple packets simultaneously, and by piggy-backing acknowledgements and window updates onto outbound data packets when possible.

Kevin Faulkner is an engineer in the IND Data Comm lab. He joined HP in 1980, after receiving his BA degree in Psychology and MS degree in Computer Science from Indiana University. Kevin was responsible for the implementation of the NS/3000 TCP and PXP protocols.

The poor man's DS, fact or fiction.

By Rene van Geesbergen,
Holland House, AALST, Holland.

Abstract.

Through the years, the only way of connecting two HP3000 systems in a reasonable feasible way, was the well-known DSN/DS Software in combination with additional hardware (INP, SSLC). With the introduction of the cheaper systems (Mighty Mouse) this proves to be a rather expensive method.

For those, who do not require/demand high transmission speeds, a cheaper method could be envisaged. By connecting two HP3000 terminal ports, two systems can communicate with each other at a speed of 9600 or even 19200 bps.

This paper will elaborate on the software and hardware requirements to support this type of communication.

Items to be discussed will be:

* Opening a Line;
* Communication Internals;
* Remote Logon and Remote Commands;
* Network File Transfer;
* Program-to-program Communication;
* Remote Database Access;
* Peripheral Sharing;
* Remote File Access;

Contents.

1. Introduction.
2. Design considerations.
3. The easy part.
4. The not-so-easy part.
5. The difficult part.
6. Conclusions.

1. Introduction.

One of the big advantages of Hewlett Packard computers has always been that they offer various communication methods like RJE, MRJE, IMF, DS and more recently NRJE. The most widely used method is DSN/DS that enables the user to communicate from an HP3000 with another HP3000 or an HP1000 in a very sophisticated way. The features supported by DS are :

- Remote command execution
- Network File transfer
- Program-to-program communication
- Remote database access
- Peripheral sharing
- Remote file access

The combination of these features makes DSN/DS to a very powerful product. However, there is one big disadvantage : it is rather expensive. To make this communication possible two pieces of hardware (f.i. INP's) and two copies of the DSN/DS software are needed. Total costs approx. $ 15.000,-. This will not be a big problem when connecting two series 68 computers, it gets out of proportion when connecting two or more series 37 computers.

This paper will elaborate on the possibility to access a remote HP3000 on a DS feature level, using an asynchronous link, i.c. using two ATC, ADCC or ATP ports connected with a $ 50,- cable, or connected over a public or leased telephone line using datacomm equipment.

In this paper, it is assumed that the reader is somewhat familiar with datacomm and the DSN/DS software. Only the principle of a poor man's DS is discussed here. A detailed study reveals a lot of additional problems not mentioned in this paper. Also the programming techniques needed to make a software package like PMDS are not discussed here. This would make the problem less understandable by non-technical readers.

The following chapter will discuss a number of design considerations with regard to a Poor Man's Distributed Systems connection or PMDS as it will be called in this paper.

Chapter 3, 4 and 5 will discuss the various DS features to be incorporated in PMDS with an increasing level of difficulty.

2. Design considerations.

Principle of operation

The first thing to do is look at DS, and see what can be learned from that. Figure 1 shows a schematic representation of the DS principle.

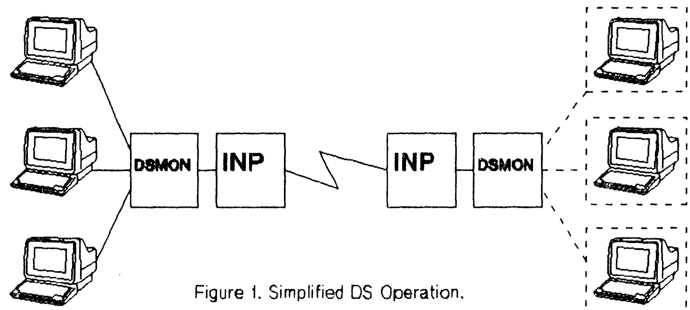


Figure 1. Simplified DS Operation.

There is one DS-subsystem process (DS monitor) for each communication link. A DS monitor (DSMON) communicates with an collegue DSMON on the other system. A user process that wants to use that link communicates with the local DSMON, and the DSMON takes care of the actual transport. On the remote machine communication is established between the remote DSMON and a remote user process. The DS processes are transparent to the user, but they are there. In order to give a user remote interactive capabilities, a virtual terminal is assigned to the local user.

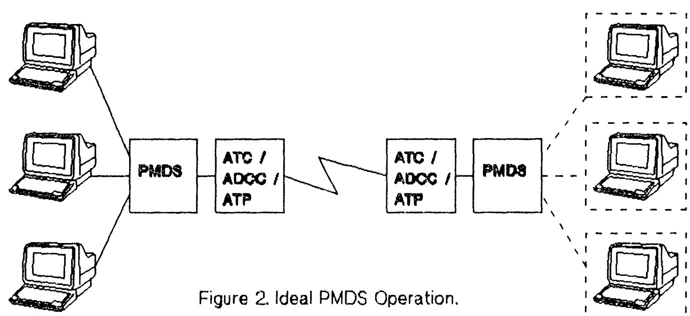


Figure 2. Ideal PMDS Operation.

Figure 2 shows the ideal situation for PMDS, because now the expensive INP connection has been changed into a cheap asynchronous port connection. However, there is one big problem : How can virtual terminals be assigned to the local users? Virtual terminals are internal to MPE, they must be configured in the I/O configuration, they must have a special driver etc. MPE looks at the asynchronous port as a physical terminal, and it will be very difficult to change that. It requires changes in the MPE internals, Privileged Mode, with the risk of brand new system failures and other kinds of nasty things. This might be the ideal operation, but it is not an ideal situation.

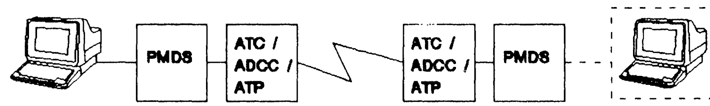


Figure 3. Actual PMDS Operation.

Figure 3 shows the simplified version of the previous one. The line is now exclusively in use by one local user. Now one 'virtual' terminal exists on the remote system, but that remote system thinks it is a physical terminal. Obviously the disadvantage of this method is that the line cannot be used concurrently by more than one user. However, for the time being this method of operation will be used by PMDS.

Protocol

The PMDS software has to take care that all data is transmitted and received correctly, and that no data is lost. Examples of what can go wrong are :

- Local system is writing but remote system is not reading
- Local system does not read while remote system is writing
- Both systems are writing
- Both systems are reading
- Etc.

The local PMDS software is almost unaware of the activities performed on the remote system, it does not know if it is talking to MPE or TDP, whether it must expect little or much data, etc. Since the output generation of the remote system must be controlled, a handshake between the two systems must be used. The major two handshake methods used on the HP3000 are xon/xoff and enq/ack.

In the terminal-to-computer communication, the xon/xoff protocol is controlled by the terminal. As the terminal transmits an xoff (dc3) to the computer, the computer suspends sending data, as the terminal sends an xon (dc1) the computer resumes sending data. The enq/ack protocol is controlled by the computer. After a block of text, the computer sends an enq and stops transmitting. When the terminal is ready to receive the next block, it sends an ack to the computer.

In the PMDS situation, the enq/ack protocol can be very useful when the remote system must be controlled when generating output. The xon/xoff protocol has no meaning in the PMDS situation, since the local PMDS software cannot send an xoff while reading. It effectively means no handshake protocol at all. This can only be very useful when the local PMDS software knows in advance how many data to expect during a read.

A problem will arise when the local PMDS software has to communicate interactively with the remote system. When a command is sent to the remote system, that remote system will not necessarily wait until the local software is ready to receive data. It may well be possible that it starts sending data immediately. In this case data may get lost, especially when using high transmission speeds and when the workload on the local system is big.

A possible method to prevent this is by using NOWAIT I/O. The only drawback of using NOWAIT I/O is that the program file suddenly needs Privileged Mode capability. The use of NOWAIT I/O makes it possible for the local PMDS software to issue a read before sending the command to the remote system. (Using Privileged Mode for NOWAIT I/O is totally harmless, Nobody ever understood why HP requires that the user must have PM capability for using NOWAIT I/O instead of some other (NW?) capability). Experimentation will probably be necessary to solve the protocol problem completely.

Data integrity

Especially when two systems are connected via modems, line distortions may occur, corrupting the data being transmitted. It is important that PMDS checks for data validity as much as possible. This will be a problem when the process that sends the data is a non-PMDS process, because that process just sends the data, and that's all. Even if the PMDS software on the remote side detects a corruption, there is no way to request a retransmission.

The situation changes when two PMDS processes communicate with each other, because then full data integrity can be ensured. A checksum can be calculated for each record transmitted, and checked again on the other system. If necessary a retransmission can be requested, the number of retries can be counted, etc.

It is therefore mandatory, that all communication, except for remote command execution, is handled by two PMDS processes, so that a high degree of data integrity can be achieved.

User interface/MPE compatibility

It would be nice if PMDS would look like DS. This would minimize the changes in UDC's, job streams etc. and it will require hardly any special training for the users and application programmers.

In order to achieve this, PMDS must initially look like MPE, it must display a semicolon as a prompt, it must accept and execute local MPE commands, it must intercept DSLINE and REMOTE commands, file equations, etc. A very nice feature would be if PMDS could handle UDC's.

It would also be nice if PMDS would include a configuration table that links a remote system name to one or more logical device numbers of the ports connected to that remote system. In that case it will be possible for the user to access a remote system using the remote system name, and PMDS will find out which of the configured ports is available, if any.

PMDS intrinsics, used for program-to-program communication must have the same names, parameters and functions as their DS-collegues, f.i. POPEN, PWRITE, GET, ACCEPT etc. They must reside in a segmented library that can be linked to a program using the LIB= option in the :RUN command. This means that existing programs using PTOP will run under PMDS without modification or recompilation. The same goes for IMAGE intrinsics, used for remote database access.

3. The easy part.

This chapter will discuss a number of features, to be incorporated in PMDS. A number of ideas, thoughts and solutions will be given on the subjects Remote Logon, Remote Command Execution and Network File Transfer.

Remark : the title of this chapter does not imply that it will be easy to build a program as described below, it merely means that it can be done without changes in MPE software and/or HP3000 hardware.

Remote Logon

The first part of PMDS is very simple. A user just should enter the MPE command :RUN PMDS. Of course this can be done using a (logon) UDC. Nothing spectacular happens. PMDS opens the file COMMAND.PUB.SYS and checks if UDC's exist for this user. If so it opens the UDC files and displays a semicolon, thus pretending it is the MPE command interpreter.

The user can enter commands like SHOWJOB, RUN, or L (a UDC command meaning LISTF). PMDS must pass the commands to the COMMAND intrinsic, or start programs, i.c. it must do everything as if it were the MPE command interpreter.

Now the user wants to start a remote session, so he enters the DSLINE command. PMDS recognizes the DSLINE command and gets the dsdevice name of the remote system that should be part of the command entered. Using the PMDS configuration file it checks the dsdevice and gets the logical device number (or numbers) of the ports connected with the remote system. Now it can check if that port (or one of these ports) is available, and open it. Note that this is the moment for PMDS to enter Privileged Mode, because the port has to be opened with the NOWAIT I/O option. After a successful open of the port, User Mode is restored. To make everything look real, PMDS can now display a message like 'PMDS LINE NUMBER = #Ln.', where n is a number identifying the remote system.

The next step for the user is to enter the 'REMOTE HELLO user.acct' command. PMDS recognizes the REMOTE HELLO command, issues a (nowait) read, sends a carriage return (cr) to the remote system, and waits for the read to complete. If everything is OK, the remote system will respond with something like cr/lf/:/dcl. If the response is garbage or nothing at all, a second, third, etc. attempt can be made to get this first and important response from the remote system. After a sufficient number of attempts, PMDS should stop trying and report to the user that the remote logon failed due to a non-responding remote system. Suppose that the proper sequence of characters was received, PMDS strips the word 'REMOTE' from, and adds a 'TERM=10' parameter to the original command string, issues a second (nowait) read, transmits the command string, and waits for the read to complete. It is not evident what will be received now, it could be f.i. 'CAN'T INITIATE NEW SESSION NOW', 'ENTER USER PASSWORD' or even 'HP3000 / MPE V etc.' But this sequence of actions, i.c. issuing a read, transmit data and wait for the read to complete can be repeated until we have a remote session established. Until now PMDS has done nothing more than emulating a terminal.

A few rules have to be observed when emulating a terminal.

- Always issue a nowait read before transmitting data to the remote system. Then complete the read.
- Configure both ports initially as terminal type 13.
- Transmit an ack to the port when receiving an enq.
- Start reading from the terminal keyboard when receiving a dc1.

Remote Command Execution

Now the user is successfully logged on to the remote system. The last thing received from that remote system was the string :/dc1, so PMDS knows that it must read input from the terminal, it will display a semicolon and wait for user input. It keeps on behaving like MPE, so the SHOWJOB command generates showjob output and EDITOR still runs the program EDITOR.PUB.SYS. But as soon as PMDS detects that a command starts with the word REMOTE, it will take special action. It will strip off the word REMOTE, and send the command to the remote system. It then will keep on receiving and displaying data until a new :/dc1 is received, and that is the sign for a new user input.

Network File Transfer (NFT)

It is well known, that users will never be satisfied with the features they already have, and the PMDS user is no exception to that rule. Now he wants to copy a file from the local to the remote system. In order to remain compatible with MPE, network file transfer is (of course) initiated by the DSCOPY command. As soon as PMDS detects this command, it will start a remote PMDS process by sending a 'RUN PMDS,SLAVE' command to the remote machine. Note that the remote version of PMDS uses a different entrypoint, so the remote process knows that it is the slave of a master process on the local machine. As soon as the two processes have shaken hands, and have identified themselves to each other, the master PMDS parses the DSCOPY command, opens the local file, and retrieves all information on that file. The master then sends a request to the slave to build that file on the remote system.

Two problems arise here, the first problem is that the master has to ensure that the slave has received the data correctly. A line distortion during the transmission could cause the file information to be garbled, or worse: slightly changed. The usual method to prevent this is by adding block check characters to the data. The master has to perform a smart calculation on the data, and add the result to the data. When the data is received by the slave, it performs the same calculation, and checks if the result is the same as the result sent by the master. If not, the slave must request a retransmission, and the master must sent the information again (and again, and again...).

The second problem is that the file information from the file label is binary data, although it will have an ASCII representation. The 'recordlength' field is one 16 bit word, but is also is 2 ASCII bytes, and those two bytes can have any value, f.i. enq, dcl, xoff, etc. It may well be that the PMDS software or the terminal driver takes undesired actions depending on the 'random' characters in the data transmitted to the remote system. The same goes for 'intelligent' datacomm equipment used for the physical connection. The master PMDS software has to make sure that only 'harmless' ASCII information is sent to the remote system. It does so by checking each character of the data. If that character is a 'dangerous' character, PMDS will change it to an 'innocent' character, and mark it for the slave by placing a 'special' character in front of it. Note that the 'special' character is also a 'dangerous' character, so if that character is part of the original data, it has to be replaced as well. When the master has finished the 'editing' of the data, it can be transmitted to the slave. Then the slave must scan the data, looking for 'special' characters, and reverse the modifications made by the master.

Now that the slave has received, checked and transformed the file label information, it can build the file on the remote system, and report success or failure to the master. If the build was successful, the master can start transmitting the actual file data to the slave. The same procedures for data checking as described above must be executed for the actual file data as well, to ensure integrity of the file contents.

While playing around with the data to be transmitted, it can be considered to try our hand at some data compression. This can give tremendous speed increase or tremendous overhead, and everything in between. It could prove wise to add some code to the compression module that tries to determine if it is worthwhile to spend valuable CPU time on compression. This is dependent on all sorts of factors like CPU load on the system, transmission speed, contents of the data being transmitted, etc. The compression module could f.i. try to compress the first 50 records of the file, and estimate if there was any benefit. If so it could decide to transmit the entire file compressed, and if not to transmit the entire file uncompressed. Of course there is a risk that the wrong decision is made, but experiments have shown that the conclusion is usually correct.

When the master encounters the end-of-file, it must sent a special control sequence to the slave, so the slave knows that the entire file has been transmitted. The slave then can close the file and report success or failure to the master. It would be nice if the master would have the possibility to request another file to be copied, just like DSCOPY, and repeat the actions described above, but eventually there will be no more files to copy. The master will then kill the slave on the remote system, display a semicolon and wait for user input.

The method described above is the same when the user wants to transfer a file from the remote system to the local system. The local PMDS then behaves like the slave, and the remote PMDS behaves like the master. The local PMDS must send a special control sequence to the remote, meaning 'until further notice you will be the master'.

One additional remark on protocol: During the file transfer, both PMDS processes have a rough idea on the amount of data that will arrive during a read. It will make the NFT code easier if the protocol is changed to xon/xoff during file transfer. This will avoid a record transmitted being spliced in two or more blocks with enq's, and it will speed up transmission, when intelligent datacomm equipment is being used.

4. The not-so-easy part.

The subjects discussed in this chapter fall in the category 'not-so-easy' for the following reasons :

- A relatively new feature in the MPE operating system, i.c. IPC (message files), must be used to accomplish the tasks discussed in this chapter.
- These tasks are normally performed by system intrinsics, that reside in the system SL. In the PMDS situation they are performed by PMDS intrinsics, linked to the program that uses them. This means that part of the MPE operating system is temporarily replaced by other software.
- Using the features described below can involve a maximum of four processes communicating with each other.

Still it looks feasible to include the features Program-to-Program Communication and Remote Database Access in a basic PMDS system.

Program-to-Program communication (PTOP)

PTOP is initiated by a user program on the local system. That user program calls special (PM)DS intrinsics, and these intrinsics take care of the actual communication. The most important intrinsics are summarized below :

Master intrinsics :

- POPEN, creates a slave user process on the remote system.
- PREAD, requests information from the remote system.
- PWRITE, sends information to the remote system.
- PCONTROL, exchanges control information with the remote system.
- PCLOSE, terminates the slave user process on the remote system.

Slave intrinsics :

- GET, receives PTOP request from the local system.
- ACCEPT, accepts and completes the local systems PTOP request.
- REJECT, rejects the local systems PTOP request.

As mentioned earlier, PMDS should use the same names for its PTOP intrinsics as DS does. In that case any PTOP program can be used both with DS and PMDS. When the program wishes to use the PMDS intrinsics, it should be run with the PMDS library linked to it.

A schematic diagram of the PMDS program-to-program communication operation is illustrated below.

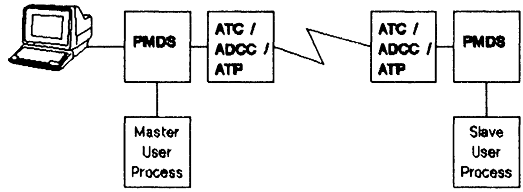


Figure 4. PTOP Principle.

In order to prevent confusion, resulting from the use of the terms master and slave, the following notations will be used to distinguish the different processes involved.

- PMDS-master is the master PMDS program, running on the local system.
- PMDS-slave is the slave PMDS program, running on the remote system.
- USER-master is the master user program, running on the local system.
- USER-slave is the slave user program, running on the remote system.

The USER-master process is started as a son of the PMDS-master process with the :RUN command. The first interesting action, performed by USER-master is that it will start USER-slave on the remote system to communicate with. It does so by calling the POPEN intrinsic. The POPEN intrinsic must get in touch with PMDS-master, and send it information such as the name of USER-slave, entrypoint, stack size, etc. This information must be sent to PMDS-slave, and then PMDS-slave can create USER-slave, with the PMDS library linked to it. In the meantime PMDS-master builds and opens two message files in order to facilitate the exchange of information between PMDS-master and USER-master via the PMDS intrinsics. Two message files are needed to support both send and receive operations. The same is done by PMDS-slave on the remote system.

Now USER-slave is running, and executes the GET and ACCEPT (or REJECT) intrinsics to confirm its existence to USER-master. These intrinsics pass the confirmation to PMDS-slave, PMDS-slave transmits the information to PMDS-master, and PMDS-master sends it to USER-master. When USER-master (i.c. the POPEN intrinsic) receives the information, it will continue running and executing other PMDS intrinsics.

The same principle goes for the other PTOP intrinsics. F.i. USER-master executes the PWRITE intrinsic. The intrinsic sends information (via IPC) to PMDS-master. PMDS-master sends the information (via ATC/ADCC/ATP) to PMDS-slave, and PMDS slave sends it (via IPC) to USER-slave, who receives the information via the GET intrinsic. Then the ACCEPT intrinsic is executed by PMDS-slave, it sends information to PMDS-slave, to PMDS-master and finally to USER-master, who receives the information via the PWRITE intrinsic that was called previously.

To summarize the PTOP principle :

- A master intrinsic sends information to PMDS-master, and waits for response from PMDS-master.
- Information received by PMDS-master is sent to PMDS-slave, then PMDS-master waits for response from PMDS-slave.
- PMDS-slave sends the information to USER-slave (GET intrinsic) and starts waiting for response from USER-slave.
- The GET intrinsic starts waiting for response from PMDS-slave. When it has received information USER-slave continues running.
- The ACCEPT and REJECT intrinsics send information to PMDS-slave.
- PMDS-slave receives the information, and sends it to PMDS-master, it then starts waiting for response from PMDS-master.
- PMDS-master sends the information to USER-master, it then starts waiting for response from USER-master. Which closes the circle.

Remote Database Access

Surprisingly the principle of remote database access is similar to PTOP. IMAGE databases are accessed using intrinsics, f.i. DBOPEN, DBGET, DBPUT and DBCLOSE. As with PTOP it is possible to define PMDS 'database' intrinsics that can be linked to the user process that wishes to use them.

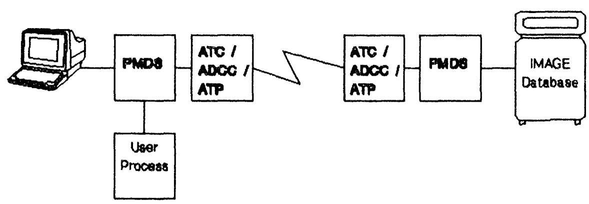


Figure 5. Remote Database Access Principle.

Figure 5 illustrates the principle of remote database access. The user process wishes to access a database on the remote system, and calls the DBOPEN intrinsic. The DBOPEN intrinsic sends the request to open the database to PMDS-master (via IPC), PMDS-master sends the request to PMDS-slave and PMDS-slave executes the actual DBOPEN intrinsic. It must return the condition code to PMDS-master, who returns it to the user process. This is the generalized method of operation used for all database intrinsics.

There is one pitfall however. Suppose that the user process wishes to use two databases simultaneously, one on the local, and one on the remote system. This means that all PMDS database intrinsics should be capable of recognizing whether a database is located on the local or on the remote system.

There are two methods to access a remote database. The first method is to set a file equation for the database, pointing to the remote system, and the second method is using a database access file. In both cases, a true DBOPEN call will fail, since IMAGE will report that the remote system was not found (IMAGE does not know of PMDS remote systems). This will be the indication that the database resides on the remote system. If the true DBOPEN succeeds, or results in a failure, other than a communication failure, then PMDS may assume that the database resides on the local system. The PMDS DBOPEN should mark the databases as local or remote, so other database intrinsics know if they must pass the information to PMDS-master, or to the actual IMAGE intrinsic.

A very nice feature of IMAGE is the database access file. This file makes it possible for the user to access a remote database as if it were on the local system. At the time the database is opened, there is no remote session running. The database access file contains information about the remote system, the remote logon id and the name of the remote database. The IMAGE DBOPEN is called using the name of the database access file, IMAGE discovers that the database resides on a remote system, and issues a remote logon automatically. It would be very nice if PMDS supported the same way of accessing a remote database. This will not be a big problem, except for the fact that the database access file is a Privileged file. This means that the database access file must be opened in Privileged Mode (sorry about that), the information retrieved, a remote session started and the remote database opened in one PMDS DBOPEN call. Technically the problem can be solved, however, Privileged Mode is needed once more. But, like in the case of NOWAIT I/O, this is safe usage of PM, since the database access file will be opened with read access only.

5. The difficult part.

This chapter discusses the two most difficult subjects in the PMDS situation. The biggest problem is that both subjects have to do with the MPE file system, and the fact that there are many ways for a program to access the file system. Just look at how f.i. SPL, FORTRAN and COBOL access a file. Almost every language has another interface to the file system. So it will not be simple to write PMDS substitution routines that replace that part of MPE, just like PTOP and Remote Database Access did.

Peripheral sharing

The next feature the PMDS user wants to have is the possibility to access a peripheral device on the remote system.

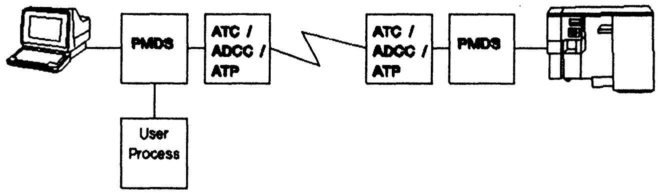


Figure 6. Peripheral Sharing Principle.

Figure 6 shows the principle of peripheral sharing. The user again has a semicolon and a blinking cursor on his screen. Now he enters a file equation like :FILE OUT;DEV=dsdevice#device;CCTL. The local PMDS software must detect that a file equation was entered for a remote device. It must NOT send this command to the COMMAND intrinsic, but it must strip off the 'dsdevice#' part of the command and send it to a slave PMDS process on the remote system. Then the slave must execute the command, so the file equation for the actual device now exists on the remote system. Now the master will create a message file, change the original file equation to :FILE OUT=msgfile;CCTL, and execute it. That completes the preparations for accessing the remote device.

The next step is that the user wants to write information to the remote device, so he starts his own program by issuing a command like :RUN MYPROG. This is a local command, so PMDS will start MYPROG as a son process. However, there is one difference compared with earlier situations : PMDS will not suspend itself, it will start looking if something happens with the message file. It is not important what actions are performed by MYPROG, but one thing will surely happen : MYPROG opens the file OUT, and writes data to it. The master discovers that data is being written in the message file, and sends a request to the slave that the remote file OUT has to be opened. Once the slave has done that, and has reported success to the master, the master can start reading data from the message file, send it to the slave, and the slave can write the data to the device. This procedure goes on until the master encounters an end-of-file condition on the message file, indicating that MYPROG has closed the file OUT. The master then can tell the slave that he can close the file OUT also.

That is the principle of peripheral sharing. However it is not as simple as it looks. The master has to know that the user wants to access a remote peripheral device. This will not be a problem when the file equation is entered directly in the master PMDS program, but suppose that the file equation is executed by MYPROG using the COMMAND intrinsic. Then the master process knows nothing about the plans of MYPROG to write to a remote peripheral. An advanced version of PMDS could support a COMMAND intrinsic, linked to MYPROG, that looks out for remote file equations, and passes all the other commands to the actual COMMAND intrinsic. That would be an improvement. But now suppose that MYPROG uses the FOPEN intrinsic to access the peripheral directly through the DEV parameter. In this case we have the same problem again, unless we also substitute the FOPEN intrinsic. But now suppose ....., etc. Things tend to get more complicated now, but the general conclusion is that peripheral sharing is only possible when the master knows in advance that a user process wishes to access a remote peripheral, and that the master knows the name of that peripheral.

The second problem to be encountered is that the communication between the master process and the user process is established through a message file. This has great advantages, as described in other Conference Proceedings, but it also introduces an additional problem. A standard disc file can be opened by a process with read/write access. A message file, however, must be always opened with read OR write access, not both. This implies that we can access the remote peripheral either with read or with write access. So full access to all remote printers can be achieved, but only limited access to f.i. remote tape units. Read or write access will work fine, but more complicated actions that read and write to the tape will not work. In that case the user process will fail due to an invalid access to the message file.

The method described above will also work when a remote user process wants to access a local peripheral. To make things more complicated, imagine the following situations :

- A program wants access to a remote peripheral more than once.
- A program wants access to more than one remote peripheral simultaneously.
- A program accesses a remote tape unit, the reply request will be issued by the remote slave on the remote system, while the local user process already writes to the message file, thinking that it is the tape unit.
- A process, accessing a remote tape unit, checks the device characteristics of the file opened, and fails when it discovers that it has actually opened a message file.
- etc.

Surely more of these potential pitfalls will come to mind while reading the above, however this paper will not present solutions for all these problems. In some cases there will be no solution at all, so this is the first time that PMDS will loose full compatibility with DS. However, the most frequently used form of peripheral sharing, i.c. remote printing, can be covered by PMDS.

Remote File Access (RFA)

The discussion on Peripheral Sharing is also valid for Remote File Access.

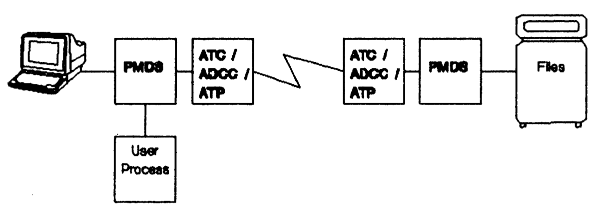


Figure 7. Remote File Access Principle.

The principle of RFA, shown in figure 7 is similar to the principle of peripheral sharing, shown in figure 6. The problem of read only/write only access is the biggest problem here, and it will be very hard to solve. An additional problem that arises using RFA is that not all operations are permitted or possible on IPC files. Things like direct read/write are not allowed, certain control operations are not allowed, and things like file locking will work fine... but, the IPC file will be locked, not the remote file. Part of this problem can be solved by writing substitution modules for the standard file system intrinsics FOPEN, FREAD, FWRITE, etc. However, there still remain a lot of other 'intrinsics' that are part of the file system. Some examples :

- IO'OPEN'C, IO'READSEQ'C, IO'WRITEKYD'C, IO'CLOSE'C (COBOL).
- FMTINIT', IIO',TFORM' (FORTRAN).
- CKOPEN, CKREAD, BKWRITE, BKCLOSE (KSAM).

All these modules reside in the system SL. They are part of the run-time library belonging to a specific language.

This is truly the difficult part of PMDS. It will be almost impossible to write (and maintain!) substitution modules for all file system modules that are part of the system library. Most of these modules are not documented, and not used directly by programmers. (How many COBOL programmers know that they call IO'OPEN'C each time they open a file).

If a user desperately needs to read from and write to a remote file, perform direct reads and file locks, there seems to be only one solution. An advanced version of PMDS could support substitutions for the 'normal' file system intrinsics FOPEN, FCLOSE etc. The functions, performed by these PMDS intrinsics would be similar to the functions of the remote database access modules. If the user is willing (and able) to use these intrinsics in his program, his problem is solved.

However, if only the read/write problem needs to be solved, there is a second, much simpler solution. Suppose the user were able to access two remote files simultaneously. The only thing PMDS should do to support that feature is passing the file number to the remote slave. That filenumber uniquely identifies each file opened by a process. Further it should maintain some table to keep track of the relationship between local and remote file numbers. If PMDS would support that feature, two way file access is possible. Simply open the remote file twice, first with read access, and then with write access. The obvious disadvantage of this method is that special programming changes have to be made, and that it will give more overhead on both systems.

Of course these limitations are partly caused by the wish that Privileged Mode must be avoided as much as possible, and should be very safe if really needed. (How about writing a replacement module for ATTACHIO ??).

6. Conclusions.

The previous chapters have shown that it is possible to build a PMDS-like system, that supports most of the features of DS, without mandatory changes in application software, operating software and/or hardware. The feature of special operations on remote files/peripherals is the bottleneck encountered so far.

Still, we believe that there will be a lot of situations where PMDS can be used very effectively, despite the lack of advanced RFA possibilities.
A quick comparison of DS and an advanced version of PMDS :

	DS	PMDS
Max. speed (cps)	7000	1920
Rem. commands	yes	yes
NFT	yes	yes
PTOP	yes	yes
Rem, DB access	yes	yes
Per. sharing	yes	limited
RFA	yes	limited
X.25/X.21	yes	yes
Concurrent access	yes	no
Hardware costs ($)	10.000	1.000
Software costs ($)	4.750	?.???

The prices mentioned above are based on the connection of a series 68 to a series 37, with full software costs.

Building a system like PMDS will not be simple, a lot of special programming techniques are required. The software must run very efficiently to prevent unnecessary overhead, and there still remain a lot of problems to be solved, f.i. break and control-y handling, detection of operator messages that disturb a critical message, detection of system failures and power failures of one of the systems, etc. However, we hope that this paper has shown that it is possible to perform system-to-system communication on a 'DS feature level', by connecting two asynchronous ports with a simple cable.

Biography

Rene van Geesbergen met his first HP3000 system in 1979, when he joined the Dredging Division of the Royal Boskalis Westminster Group as a technical programmer. In 1981 he moved to the Information Department of the same company. His new function was a combination of system manager of two HP3000's and system programmer. In April 1984 Rene and his partner Jelle Grim founded Holland House, a small company specializing in HP3000 system management consultancy, and supporting software. Within Holland House he specializes in communication products and HP3000 internals.

LOCAL AREA NETWORK : BUILD UP YOUR PRIVATE X.25 NETWORK

Christian GRESSET
Hewlett-Packard, Grenoble, France

Summary

Local networking, for which demand is quickly increasing, allows communications between systems and sharing of resources between them (eg laser printer). In the same time a significant move toward the use of X.25 concepts is observed, thanks to the fast development of national packet switching networks (PSNs) which provide worldwide low cost data transmissions.
That's why a device is now getting very attractive : the X.25 switch. It simplifies the interconnection of local systems, thus progressively building up a private data network. It also allows the reduction of costs for remote transmissions by sharing a connection to a PSN between heterogeneous on-site equipment.
Many vendors today offer multi-switches wich cover a wide range of capabilities and pricing. It appeared important for HP to allow its customers to purchase fully compatible and reliable equipment. Our work in Grenoble consists in verifying the compatibility of HP's three X.25 products :

* DSN/X.25 on the HP3000
* DSN/X.25 on the HP1000
* HP 2334A, the cluster controller

with switches which vendors have been chosen according to criteria that will be explained. We shall show the results of our tests and give the features of the devices that we have certified.

1) X.25 PUBLIC NETWORKS BENEFITS

The growth of the use of Public Packet Switching Networks (PPSN) for transporting data from one location to another is speeding up every year, especially in Europe: Transpac in France monthly installs about 700 new connections and claims more than 20.000 subscribers. Eirpac in Ireland, ITAPAC in Italy, Iberpac in Spain are in test phase and should be open soon; thus all the countries in Western Europe will have their own PPSN in operation.
This success is mainly due to the following benefits :

- The existence of CCITT X.25 Recommendation, a clearly defined standard irrespective of computer manufacturer or design.
- The reliability of a PSN is high due to the protocol itself, and to redundant connections between computers, thus allowing rerouting and retransmission of data transfer.
- The flexibility of a PSN allows multiplexing of data transfers over a single access link that is easily configured or reconfigured if the communications requirement changes.
- The cost of data transfer is based upon the amount of data transfered across the network. For a given range of data transfer volumes, the cost of an X.25 link to a PSN can be more attractive than the use of leased or dial-up lines. Most Public PSNs charge based on network usage and not distance.
- It is now possible to access to over 30 countries worldwide from Australia to Sweden, from the U.S to Taiwan.

This makes Packet Switching ideal for applications where users scattered over a wide area must access a central facility for service. The fact that the billing is insensitive to distance also makes PSN a good way to market a product which is electronically delivered.

2) X.25 PRODUCTS PROVIDED BY HP

Due to the success of the X.25 Public Networks, HP has developed a range of X.25 products : the DSN/X.25 communications subsystems on the HP3000 and on the HP1000 allow direct connection of the computers to an X.25 trunk, supporting many facilities offered by the network agencies.

PUBLIC PSN CERTIFICATION STATUS
FOR HP 1000/3000/2334A (as of December 84)

Country	Network	HP 1000 Status	HP 3000 Status	HP 2334A Status
Australia	AUSTPAC	In progress	Certified	In progress
Austria	DATEX-P	In progress	In progress	In progress
Belgium	DCS	Certified	Certified	Certified
Canada	DATAPAC	Certified	Certified	Certified
Denmark	DATAPAK			
Finland	DATAPAK		Certified	
France	TRANSPAC	Certified	Certified	Certified
Germany	DATEX-P	Certified	Certified	Certified
Hong-kong	INTELPAK		Certified	
Ireland	EIRPAC	In progress	In progress	In progress
Italy	ITAPAC	In progress	In progress	In progress
Japan	DDX	Certified	Certified	Not started
Japan	VENUS-P	Certified	Certified	Certified
Netherlands	DN-1	Certified	Certified	Certified
Norway	DATAPAK	Certified	Certified	Certified
Singapore	TELEPAC		Certified	
South Africa	SAPONET	Certified	In progress	Certified
Spain	IBERPAC	In progress	In progress	Certified
Sweden	DATAPAK	Certified	Certified	Certified
Switzerland	TELEPAC	In progress	Certified	Certified
Taiwan	PACNET	In progress	In progress	Certified
UK	PSS	Certified	Certified	Certified
US	TELENET	Certified	Certified	Certified
US	TYMNET	Certified	Certified	Certified
US	UNINET	In progress	In progress	Certified

CREATED WITH HPDRAW

FIGURE 1

Moreover, the HP2334A, the X.25 cluster controller which has been upgraded into a Statistical Multiplexer allows connection of up to 16 remote workstations (terminals, personal computers, printers, plotters) to a host computer through an X.25 network. These products have been tested by the main PPSN agencies throughout the world and obtained approvals which allow customers to connect their device to the PPSN in many countries (see figure 1).

3) PRIVATE X.25 NETWORKS

On the other hand, the demand for interconnection of multiple systems together in a limited geographical area is increasing. Network communications in a building or in a group of buildings in close proximity such as an hospital or a group of manufacturing buildings on a site is called a Local Area Network.
X.25 PSN can be used as a solution for this type of interconnection, thanks to the benefits of the protocol itself : reliability and flexibility.

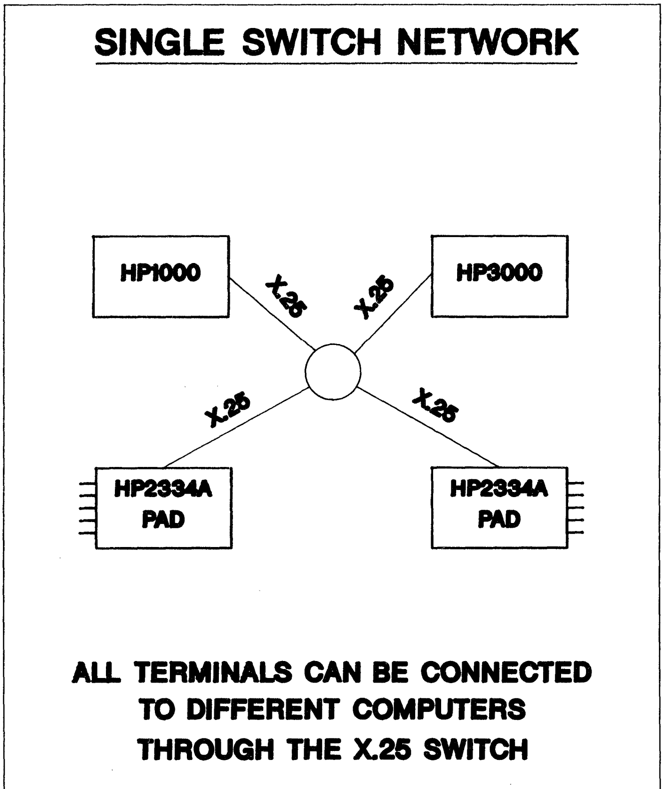


FIGURE 2

To build such a private network you need :
- cables,
- switches (at least one !)
- X.25 host computers, front-ends, cluster controllers.

The host computers and the cables are the ones used in a wide area network based on a Public PSN.
The following aims to show the features of such switches and different configurations that can be installed.

4) X.25 SWITCHES

The know-how acquired by the main PPSNs switching nodes suppliers has allowed them to develop smaller X.25 set-ups for interconnection of local computers. Other datacommunication suppliers, specialized in dataswitches for example, have entered the market of X.25 devices.
An X.25 switch is a versatile routing device which forwards data arriving from one X.25 trunk to any other X.25 trunk (see figure 2). It fully complies with X.25 levels and the X.121 international numbering plan. These features allow it to connect to PPSNs and to be used as an X.25 to X.25 concentrator :
it can concentrate multiple heterogeneous equipment to remote sites through X.25 trunks and PPSNs. (As shown in figure 3).

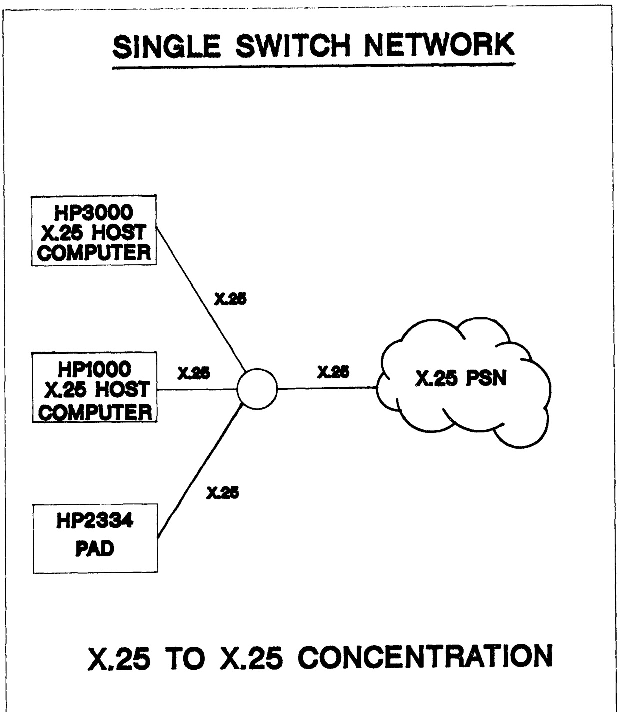


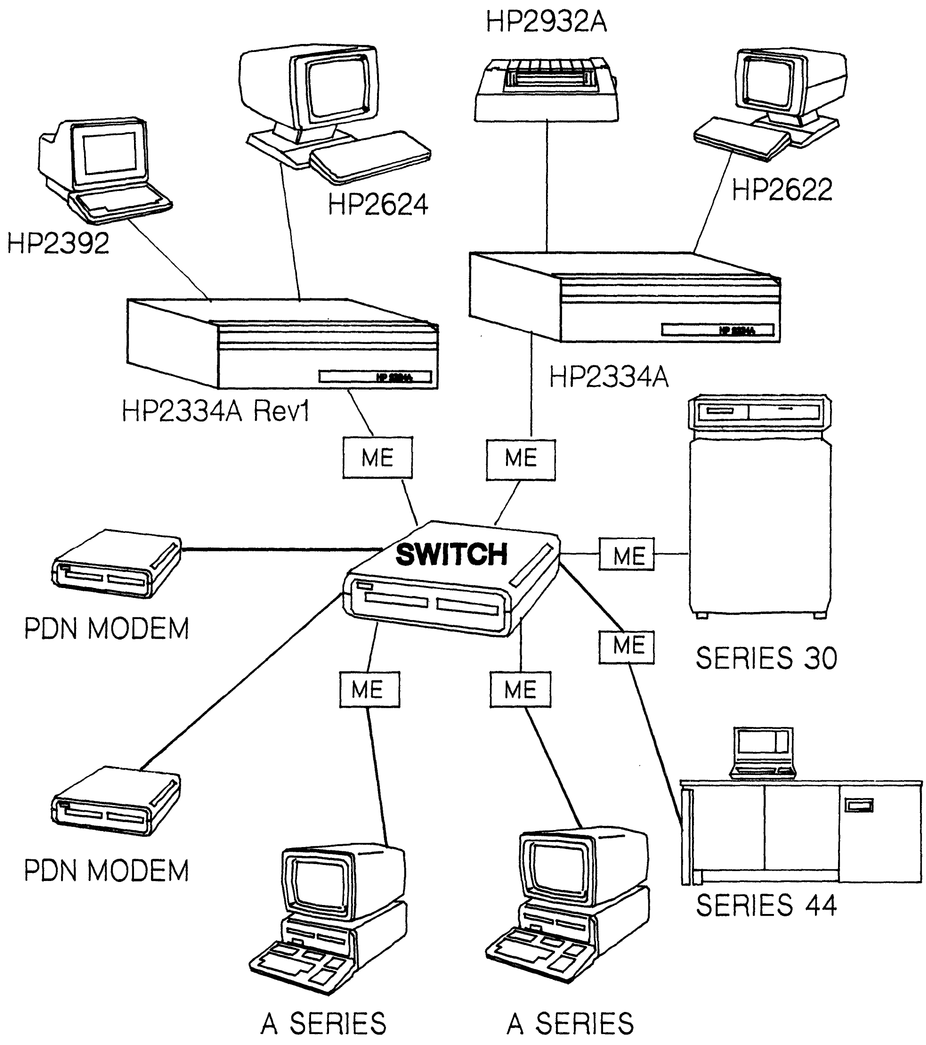
FIGURE 3

It thus includes gateway features as it allows interconnection of an X.25 based Local Area Network with a Wide Area Network.
Switch functions :
The switch services the establishment of the virtual circuits. It translates logical requests into the assignment of actual physical devices and manages the routing tables : during data transfer, the switch receives incoming packets, verifies the packet level addresses and routes the packets outbound on the appropriate communications link.
Some switches establish permanent routing for the duration of the session and pass the data over a fixed path.
Others allow dynamic routing based on traffic load and congestion on the network.

5) HP CERTIFICATION OF X.25 SWITCHES

Certification procedure
The customer demand for inter-networking several computers, whether they are installed on a local site or they are scaterred over several remote sites is increasing.

HP considers it a service to customers to certify other vendor's datacomm equipment, especially in the field of X.25 communications. Beyond the obvious regulatory aspect, certification is defined as the process through which HP can ascertain that a combination of equipment and software can work properly. Thus the customer can be confident that this type of system has already been tested.
Moreover, HP in such cases will try to establish an agreement with the other vendor to define a joint support procedure in case of malfunction of the combinaisón of equipment previously tested.



ME = Modem Eliminator

NETWORK CONFIGURATION

FIGURE 04

Support procedure.
HP chooses a vendor according to a number of criteria : quality and reliability of the devices, support quality of the Company.
A certification agreement and a non-disclosure agreeement are then signed between HP and the other vendor : both parties agree to negociate and work in good faith and to protect any Confidential Information obtained from the other party.
Certification test coverage.
Once an X.25 switch has been identified and lent by the supplier, the following tests are performed :
Configuration : A network is built as shown on figure 4. It consists of 2 HP3000, 2 HP1000, and 2 HP2334A, used as cluster controllers.
This private node network is also connected to the PPSN through two trunks, thus allowing to communicate with remote computers : HP3000 or HP1000.

TYPICAL X.25 SWITCH FEATURES

(DYNAPAC MSW.25)

* 4 – 8 trunks
* 1200 – 9600 bps trunk speeds
* 57 kbps total thoughput
* 20 packets per second
* Full X.121 routing
* PVC/SVC support
* RS–232 C
* X.25 to X.25 concentration
* Link level protocol conversion: LAP/LAPB, HDLC/BSC
* Remote control and diagnostics
* Ability to cascade switches
* Automatic alternate routing

FIGURE 5

Basic functions :
This consists of placing calls between all systems, thus verifying the routing capabilities of the switch. The file transfer is tested from one HP3000 to another, from one HP1000 to another. The main system commands are sent from the terminals connected to the HP2334A. On the HP3000 such application subsystems as HPSLATE or HPDRAW are

run across the switch. These connections are performed inside the local network and through the PPSN.

Recovery.

It is the verification of correct recovery from abnormal situations as cable disconnection, power failure on the switch and on HP devices. The behaviour of HP supplied equipment is also scrutinized under simultaneous failure conditions on the switch.

DIAGNOSTIC PORT MAIN COMMANDS
(DYNAPAC MSW.25)

CLEAR n	Clear routing table for trunk n
CONFIG PVC = n-l/n'-l'	Configure PVC port
D	Terminate call to diagnostic port
DISPLAY n	Display routing tables for trunk n
DOWN n	Disable trunk n
LST n	Display active LCNs on trunk n
ROUTE	Run ROUTE program
TRUNK n	Display information for trunk n
UP n	Enable trunk n

FIGURE 6

Support procedure.

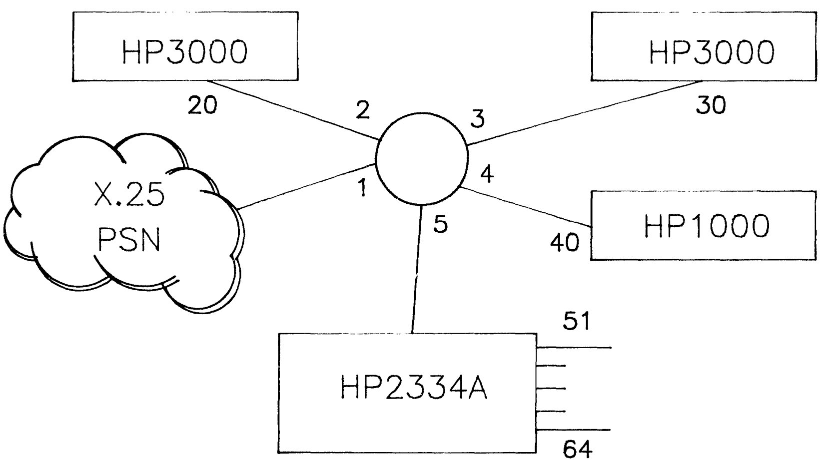
Once the compatibility of the products has been successfully tested, both parties sign a support agreement that provides support assurance to customers : they both will exchange technical information to each other's sales and support personnel through the local offices. They will consult regarding site preparation and system connection in order to prepare installation. They will ensure proper installation of both systems. Finally they will handle trouble calls jointly to resolve operational

problems.
When a customer places the call to either HP or the other vendor, whoever receives the call will manage the problem resolution regardless of the origin.

6) DYNAPAC SWITCH. MODEL 8

The Dynapac Model 8 Multi-switch went through the certification tests as previously described. This switch is a low cost X.25 Switch, wich is available with 4 or 8 ports. Its main features are described in the figure 5.

ROUTING EXAMPLE



ROUTING TABLES :

TRUNK 1	TRUNK 2	TRUNK 3	TRUNK 4	TRUNK 5
1,DDDDDDDDD	0,20	0,30	0,40	0,5D 0,6D

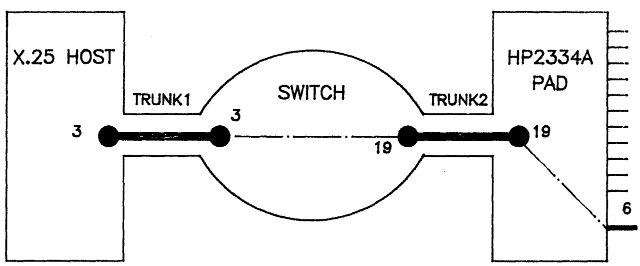
0 , 2 0
PRIORITY , SUBADDRESS

"D" stands for "DON'T CARE"

FIGURE 7

However, this switch does not provide clock timing signals so a clock source must be provided by the use of a modem eliminator device. This switch can be fully configured by the user : it is equiped with a software Diagnostic Port which can be reached from any trunk via a switched virtual circuit call. This port can be operated from an asynchron terminal connected to the switch. The Diagnostic Port is protected by a password, which can be changed via the Diagnostic Port whenever desired. This port serves two purposes : the first is to allow inquiries as to the state of the trunks and to gather statistical information. Figure 6 shows the main commands and their syntax.
The second purpose is to allow the configuration and reconfiguration of the routing table. Figure 7 gives an example of a Private network routing and its implementation in the routing table of the Dynapac switch. One can also define Permanent Virtual Circuits (PVCs);an example is given by figure 8 : when data is transmitted using that particular Virtual Circuit it will appear at the assigned destination without the need for the terminal operator to set-up the call. Instant and complete availability is one of the features of PVC services which can also be provided by some Public X.25 networks.

PVC CONNECTION



TRUNK 1 :

VIRTUAL CIRCUIT #3 CONNECTS THE HOST TO THE SWITCH

TRUNK 2 :

VIRTUAL CIRCUIT #19 CONNECTS THE SWITCH TO THE PORT # 6 OF THE HP2334A

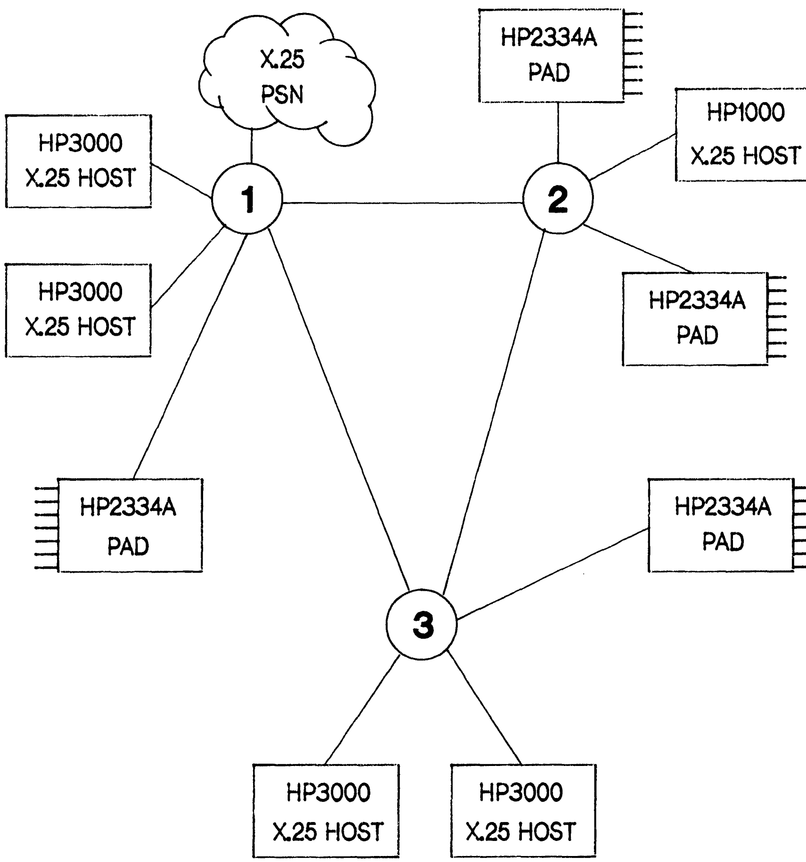
THUS, PORT#6 OF THE HP2334A
IS CONNECTED TO THE HOST
THROUGH A PVC LINK

FIGURE 08

CONCLUSION

An X.25 switch is a modular set-up which can be easily installed in most computer environments. It can be the cornerstone of a sophisticated X.25 private network as shown in figure 9.
In the wide range of X.25 switches on the market today, one can usually find a device having the required features and performance.
Its X.25 to X.25 capacity allows cost saving on access to Public Packet Switching Networks.

MULTIPLE SWITCH NETWORK



REDUNDANT PATHS :

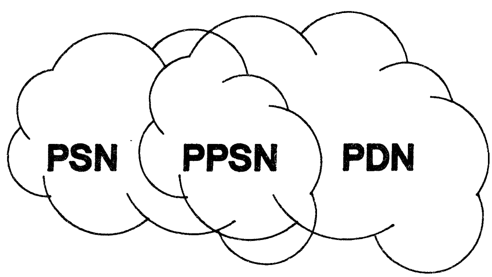
1 ⟶ 2 or 1 ⟶ 3 ⟶ 2

FIGURE 9

APPENDIX A : Glossary

Figure A1 shows the difference between the three expressions : PSN, PDN and PPSN.

SOME KEY TERMS



PDN : Public Data Network
- * Services available to any user willing to pay for them.
- * Uses packet switching or circuit switching techniques.

PSN : Packet Switching Network
- * Generally in accordance with CCITT X.25 recommendation.
- * Public or private

PPSN : Public Packet Switching Network
e.g : Datex-P , PSS , Transpac , Telenet ...

FIGURE A1

BIOGRAPHY

Christian GRESSET
is Network engineer in the European Network Marketing Center at the Hewlett-Packard computer facility in Grenoble, France. With HP for three years, he is in charge of the certification tests of datacommunication equipment and provides support for the certification of HP X.25 products on the Public Packet Switching Networks in Europe.

NETWORK SERVICES FOR THE HP3000

Brian Lynn

Hewlett-Packard

1. Introduction

Recently, HP released its next generation HP-HP networking product for the HP3000. The HP3000's new networking architecture is structured according to the Open Systems Interconnection (OSI) Reference Model. Its software is divided into two major portions: the transport protocols (Link/3000), which correspond to OSI layers one through four; and Network Services (NS/3000), which correspond to layers six and seven. The purpose of this paper is to discuss the HP3000's new networking product, with particular emphasis on Network Services.

This paper is divided into two sections. The first section describes the HP3000's networking architecture and compares the new implementation to DS/3000 (NS/3000's predecessor). The second section introduces some of the benefits of NS/3000, points out a few of its features, and presents a discussion on choosing the right interprocess communication facility for distributed applications.

2. NS/3000 Compared to DS/3000

NS/3000 is both existing DS/3000 services and new HP standard services implemented to use industry standard transport protocols. These transport protocols correspond to levels one through four of the OSI model. The Link/3000 protocols implemented for use by NS/3000 include:

- o Transmission Control Protocol (TCP)
- o Internet Protocol (IP)
- o Local Area Network (LAN) IEEE 802.3

The services implemented in NS/3000 are:

- o Network File Transfer (NFT)
- o Remote File Access (RFA)
- o Remote Data Base Access (RDBA)
- o Program to Program Communication (PTOP)
- o Virtual Terminal (VT)
- o Remote Process Management (RPM)
- o Network Interprocess Communication (NetIPC)

An important distinction to make is that NS/3000 and DS/3000 are interface compatible but not protocol compatible. The protocols used by NS/3000 are not the same as those used by DS/3000. Though programs, job streams, and UDCs which use DS/3000 will be able to use NS/3000 without modification, NS/3000's implementation is distinct from DS/3000. NS/3000 is not an extension of the existing DS/3000 software. The NS/3000 architecture is significantly different from DS/3000's, and it uses a completely different set of protocols. However, NS/3000 is designed to coexist with DS/3000, allowing a process to use both NS/3000 and DS/3000 at the same time.

Not only are different transport protocols used, but the actual services' protocols are different as well. This implies that the new NS/3000 software cannot communicate with DS/3000 software. For this reason, the DS/3000 software is included with the NS/3000 product.

2.1 Services Architecture

All NS/3000 services use ARPA's Transmission Control Protocol (TCP). TCP is a *de facto* industry standard level four protocol, providing a connection oriented reliable service. Connection oriented protocols are peer-to-peer, delivering data from one process to another. The HP3000 implementation of TCP provides the following features:

- o Guarantees in-order delivery of data;
- o Data that is not acknowledged is retransmitted;
- o Duplicate packets are detected and discarded;
- o Will perform segmentation and reassembly of data which hides from the user any data length limits imposed by the link layer;
- o Provides connection assurance to detect the failure of a peer's node, thus preventing hangs.

Unlike the DS/3000 architecture, NS/3000 does not multiplex all services' data over a single connection. Instead, each instance of a service (*i.e.*, RFA, NFT, *etc.*) has its own TCP connection. A service establishes a connection to a server process placed in the user's session on the remote node. In order to make the services more independent from one another, there is also no sharing of server processes among services.

The decoupling of services provides several advantages over DS/3000. With separate connections and servers acting asynchronously for each instance of a service, NS/3000 prevents the serialization of multiple requests found in DS/3000.

Thus if multiple processes in a session are doing RFA to the same remote session, their requests will be executed by separate servers; rather than queueing the requests to a single process, the command interpreter, as is done by DS/3000. In addition, NS/3000's server architecture permits RFA and PTOP to execute independently of each other. Unlike DS/3000 which does not permit a PTOP master to perform RFA to a slave's session, there is no such limitation by NS/3000. Another advantage of the independent server model employed by NS/3000 is that one service's failure will not interfere with another service. Aborting one server or connection used by a particular service will not affect the performance of another instance of the same service or any other service.

2.2 Link/3000 Architecture

The architecture employed by the transport modules is quite dissimilar from DS/3000's architecture. DS/3000 has a single process, DSMON, which is responsible for sending and receiving all data. Pseudo drivers are used by DS/3000 as an interprocess communication mechanism to transfer information and data between DSMON and processes using DS/3000. This results in a process switch for all inbound and outbound data. The processing of a processes' data is serialized between the pseudo driver and DSMON.

Link/3000 takes advantage of a low level interprocess communication mechanism on the HP3000 called ports. A port is essentially a procedure that is called when data is sent to it. The parameters to a port procedure include a pointer to the message that resulted in the port being invoked. Ports are analogous to HP3000 drivers. Sending a message to a port is roughly equivalent to calling ATTACHIO, and the port message is similar to an IOQ. Like drivers, ports execute on the stack of the process initiating the request, and there can be only one instance of a port procedure executing at any given time. If another process sends a port message to a port that is already executing, the message is queued to the port for the port procedure to process when it has completed processing its current message. Each NS/3000 transport module from TCP on down to, but not including, the LAN driver is implemented as a port procedure. This implementation allows data to flow from a user to the LAN without causing a process switch.

3. NS/3000 Services

An often requested enhancement to DS/3000 is to support multiple remote sessions to a single node. NS/3000 allows multiple sessions through the use of *environments*. An environment is a means of referring to a remote session by name. When users use the :DSLINE command in NS/3000, not only are they defining the destination node but are also specifying the name of the remote session.

The name given to a remote session is called an *environment identifier*. Environment ids have the same format as NS/3000 node names, and resemble MPE file names. A fully qualified node name has the form:

node.domain.organization

Likewise, an environment id has the form:

envname.domain.organization

As is done by MPE for file names, node names and environment ids are qualified by NS/3000 if not fully qualified by the user. Each portion of the name may be one to sixteen alphanumeric characters.

To create multiple sessions on the same remote node, a user would issue several :DSLINE commands to the same node, assigning a different environment id each time. Thus if a user wished to create two sessions on node X.DCL.HP, with default domain and organization DCL.HP, he could enter the following commands:

```
:DSLINE ENV1=X
:REMOTE HELLO USER.ACCT
:DSLINE ENV2=X
:REMOTE HELLO USER.ACCT
```

To refer to each session separately, the environment id is optionally permitted in the :REMOTE command. Thus if the user wanted to perform a :SHOWJOB in the first session and a :LISTF in the second, the following commands would be permissible:

```
:REMOTE:ENV1 SHOWJOB
:REMOTE:ENV2.DCL.HP LISTF
```

For compatibility with DS/3000, the current syntax of just specifying a node name with the :DSLINE command is also allowed. If that is done, the environment id defaults to the node name. Thus

```
:DSLINE X
```

is equivalent to

```
:DSLINE X=X
```

An additional benefit of environment ids is that they allow the :DSLINE command to be used in a manner similar to the :FILE command. Intrinsic parameters that referred to DS/3000 device class or node names are now interpreted as environment ids by NS/3000. If an existing PTOP program was coded with its POPEN dsdevice parameter referencing node X, the POPEN could be redirected to node Y by executing the command

```
:DSLINE X=Y
```

before running the program.

3.1 Remote File Access

The MPE file system was enhanced for NS/3000 to allow an environment id to be a natural extension to a file name. As evidenced by the :REMOTE command, a colon (:) was chosen as the delimiter to precede an environment id. The syntax

filename:envid

is accepted in both the FOPEN intrinsic and the :FILE command. An advantage of allowing an environment id to be included in FOPEN is that it permits applications prompting for a file name to reference a remote file without first issuing a :FILE command, as is required by DS/3000.

Modifying the :FILE command permits users to have a consistent view of file names and environments. For NS/3000, the following two file equations are equivalent:

```
:FILE F1=F2;DEV=X#
:FILE F1=F2:X
```

A feature of NS/3000 RFA not offered by DS/3000 is the support of nowait I/O for remote files. Using nowait RFA can provide a significant performance increase over waited RFA. If a process calls a file system intrinsic such as FWRITE in nowait mode, the process is impeded only as long as it takes to send the request off to the remote RFA server. The process is then free to perform some other function, such as get the data for the next RFA FWRITE, while the remote RFA server is executing the FWRITE to disc on the remote node.

3.2 Network File Transfer

The NS/3000 NFT protocol is a standard HP protocol. The NFT standard is currently being implemented on other HP computers, thus allowing file transfers between various nodes in a heterogeneous HP network. NFT executes in either one of two modes: interchange or transparent. The mode to be used is determined by NFT when it handshakes with its peer before initiating a file transfer. Interchange mode is used when transferring files between unlike file systems. NFT attempts to account for any differences between the file systems to make the source and destination files as similar as possible. Transparent mode is used by NS/3000 NFT when transferring files between two HP3000s. Since the file systems are the same, the source data can be written to the target file without interpretation, resulting in an efficient file transfer.

NS/3000 NFT is divided into three components: the initiator, which interfaces to the user and starts the other two components which perform the actual file transfer; the producer, which reads the source file; and the consumer, which creates the target file. Depending on how many nodes are involved in a transfer, a single NFT process can play one, two or all three of the roles.

NFT uses a true three node model for file transfers. If a user on node X wishes to transfer a file from node Y to node Z, an NFT process on each node would be created. The process on node X would be the initiator, on node Y it would be the producer, and on node Z NFT would play the role of the consumer. In this example, the file transfer would take place directly from the producer on node Y to the consumer on node Z, and the initiator would remain idle during the file transfer. This more efficient structure is different than DS/3000 NFT, where data would travel from the producer through the initiator and then to the consumer.

If the same user on node X wished to copy a file from node Y back to his node, a single NFT process would play the roles of both the initiator and the consumer on node X. On node Y, another NFT process would be the producer.

Like DS/3000 NFT, NS/3000 NFT can be invoked through either the :DSCOPY command or the DSCOPY intrinsic. NS/3000 is syntactically compatible with DS/3000 NFT. However, NS/3000 NFT also accepts an expanded syntax. This syntax includes the ability to specify transfer options and accept the *filename*:*envid* construct to achieve consistency with RFA.

NS/3000 NFT allows the user to do several things which cannot be done with DS/3000 NFT. One is that it permits the user to specify a logon string in square brackets ([]), along with the file name and/or environment id. Specifying a logon eliminates the need to do a :REMOTE HELLO before using NFT; NFT will automatically create a remote session for the duration of the file transfer.

Another very nice, but easily overlooked benefit of being able to specify a logon string is when it is used during local file transfers. If a user logged onto account ACT1 wants to make a copy of file F2.PUB.ACT2, it is quite likely MPE would return a security violation when the user tried to access file F2 across account boundaries. If that is the case, the following set of tedious commands would be required to copy the file if using FCOPY:

```
:HELLO USER.ACT2
:RELEASE F2
:HELLO USER.ACT1
:FCOPY FROM=F2.PUB.ACT2;TO=F1;NEW
:HELLO USER.ACT2
:SECURE F2
:HELLO USER.ACT1
```

However, the same results can be achieved with the following single NFT command:

```
:DSCOPY F2[USER.ACT2] TO F1
```

Another improvement to NS/3000 NFT is the ability to specify generic file sets for copying multiple files. Also provided by NS/3000 NFT is the ability to specify source file codes, including negative file codes. This capability gives a user with system manager (SM) capability the tools required for copying IMAGE databases.

3.3 Remote Process Management

A new service introduced with NS/3000 is Remote Process Management (RPM). RPM provides a program the ability to create and kill processes on the program's local node, or on a remote node. RPM uses an HP standard protocol, which is currently being implemented on other HP systems. RPM has three user callable intrinsics:

RPMCREATE - To create and activate a process.

RPMGETSTRING - To obtain a byte array passed to the created process by the creating process.

RPMKILL - To terminate a process created by RPMCREATE.

Remote Process Management allows a program to create either of two types of processes: a *local process* or a *remote process*. Local processes are equivalent to processes created by the MPE CREATEPROCESS intrinsic; they exist as a son of the creator, executing in the same MPE session. Remote processes execute in a different session than the creator. The created process' session does not have to be on the same node as the creator's session.

There are two means of creating a session for a remote process. One is the ubiquitous :REMOTE HELLO. The other is by allowing the session to be created for you by RPM. RPM permits the user to specify a logon string in the RPMCREATE call. If a logon is provided and no remote session for the specified environment exists, RPM will use the supplied logon to create a session for process. This session lasts only as long as the created process continues to execute, and may only be used by RPM.

In addition to giving the logon in the RPMCREATE call, the :DSLINE command can be used for specifying the logon. Therefore, an application need only specify an environment id and the program file name. Before running the application, a :DSLINE command can be issued which specifies the logon and maps the environment id to a specific node. Additionally, RPM permits processes other than the creator process to kill a remote process.

RPM supports all of the options offered by CREATEPROCESS, and a few more. An advantage of using RPM is that it gives the option of allowing a remote process to be created and execute independently of the creator process. Unlike CREATEPROCESS and PTOP's POPEN, an independent process will not automatically be terminated when the creator process terminates. There are many benefits of this feature. From one session, a user can run a program which initiates subsystems on many nodes, after which the user can logoff without interrupting the execution of any of the applications.

3.4 Interprocess Communication

Along with NS/3000 comes another interprocess communication (IPC) mechanism for the HP3000, Network Interprocess Communication. NetIPC joins PTOP, Mail, and Message Files on the HP3000 as a means for users to exchange information among processes.

NetIPC provides an HP standard interface to TCP through a set of user callable intrinsics. These same intrinsics are used by the NS/3000 services as their method of communication between nodes. Since NetIPC is an interface to TCP, it does not introduce a protocol of its own. Use of these intrinsics provides the hook necessary for linking an HP3000 to a non-HP3000 supporting the TCP and IP protocols.

NetIPC uses entities called *sockets*. Sockets allow users to establish a *connection* from one process to another. Once a connection is established, a process may send data to and receive data from its peer process using the connection.

NetIPC provides two methods of locating a peer's socket for establishing a connection. The first method is through the use of TCP addresses, and is available only to privileged users. Privileged users may assign a TCP address to a socket when it is created, otherwise NetIPC will choose the address. Another privileged user then needs only to specify the address and node to establish a connection. This is the method a privileged HP3000 user would need to use for talking to non-HP systems.

The second method of locating a socket is through the use of names. A process can create a socket and then give it a name. After it has been named, any other process that knows the name can reference the socket by its name and node to establish a connection. This is the method which must be used by nonprivileged users. By forcing the use of socket names, NetIPC hides from the user the peculiarities of addressing and prevents the user from randomly probing the network.

NetIPC can be used for communication between processes on either the same or different nodes. Local communication is achieved through the transport's software loopback capability. Software loopback permits TCP to establish connections on the same node. Data sent on a loopback connection flows through TCP and IP, and is turned around at the Network Interface module. Through the use of software loopback, :REMOTE HELLOs and other NS/3000 services can be executed on a process' local node.

With the various means of IPC available on the HP3000, it can be a difficult task choosing which one to use. MPE's Mail is generally the least desirable. It can be used only between a father and son process on the same node. Mail uses a shared data segment for exchanging data, which requires independent synchronization between the communicating processes. If the processes are not synchronized a process can overwrite data that it had previously sent, or the process may overwrite data sent to it which it had not yet received.

Through the use of software loopback, NS/3000 PTOP can be used for communicating between processes on the same node, in addition to communication between processes on different nodes. A limitation of PTOP is that the processes must be executing in separate sessions; they cannot be members of the same family tree. PTOP does provide the synchronization lacking in MPE's Mail. However, it is overkill in this area that is PTOP's downfall. PTOP forces a master/slave relationship between two processes. A slave process cannot initiate communication to its master, the master must request any data sent by the slave. In addition, whenever a master process sends data to a slave, the master is impeded until the slave receives the data and sends a reply. Another of PTOP's limitations is that a process can be a slave to only one master. PTOP does have its advantages, though. One is that PTOP can be used to communicate with processes on an HP1000. Another is that PTOP contains the ability for a process to create and destroy the process with which it will be communicating.

With the availability of RPM, PTOP is no longer the only method of managing remote processes. NetIPC and MPE's Message Files, when used in conjunction with RPM, provide the functionality of PTOP and more. Both NetIPC and Message Files offer similar features: nowait I/O for nonprivileged users, the ability to timeout on a receive, processes can be in either the same or separate sessions, both local and remote IPC are provided, and no master/slave restrictions. NetIPC and Message Files are both attractive interprocess communication facilities for distributed applications. But the choice of which one to use remains. Out of the context of an application's requirements, neither NetIPC nor Message Files can be considered better than the other. However, given the demands of an application, one can be more desirable than the other. Before choosing between Message Files and NetIPC for a particular application, the following trade-offs should be considered.

Since Message Files are built on top of the MPE file system, using Message Files between two systems requires the use of RFA. RFA, like the other NS/3000 services, uses NetIPC to send the data to the remote computer. Once the data has arrived on the remote node, the remote RFA server writes the data to the Message File. It should be evident that NetIPC has a significant performance benefit over Message Files when the communicating processes are on different nodes.

Message File's use of the file system does offer a benefit over NetIPC. In the event of a system failure or aborted program on the receiving process' node, queued data will not be lost if using message files, which is not the case if using NetIPC. If an application expects data to queue up and cannot afford to loose that data, Message Files provide a nice solution.

Since Message Files use RFA between systems, they can only be used when all processes are on HP3000s. Therefore, if a distributed application is not limited to HP3000s, NetIPC should be the choice since it is simply an interface to standard protocols.

A useful feature of Message Files is that they permit multiple senders and receivers on a single file. For a process to service many senders using NetIPC, the process must have connection for each process that will be sending it data. This requires the process to use nowait I/O so that IOWAIT can be called to wait for data from any one of the processes. Multiple receivers allows several processes to service a queue of incoming requests. Since NetIPC's connections are strictly between two processes, a single process must handle all data on the queue.

In order to maintain MPE's security, the use of Message Files (*i.e.*, RFA) requires a :REMOTE HELLO to have been previously executed. NetIPC provides security through the use of socket names, which are not public, and does not require NetIPC users to be logged onto the destination node. If requiring a process to logon to perform IPC presents a possible security risk, the use of NetIPC should be considered.

4. Conclusion

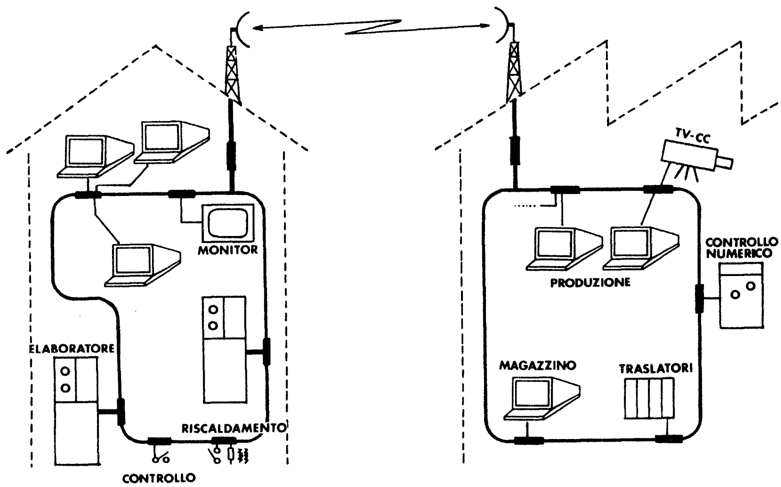
To a user already familiar with DS/3000, the HP3000's new networking product initially looks very much the same. However, internally the two are quite different. The new architecture is layered, and uses both HP and industry standard protocols. NS/3000 offers many advantages over DS/3000. Existing services have been enhanced and new services have been added, allowing the user to do many things that were previously either very difficult or impossible to do with DS/3000.

This paper has shown how NS/3000 is structured, and how it differs from DS/3000. This paper has also discussed some of NS/3000's new features. It is hoped the information in this paper will enable a user to take greater advantage of the HP3000's networking capabilities.

Brian Lynn joined HP in 1980 after receiving his BA degree in Computer Science from Indiana University. He is an engineer in the Information Networks Division Data Communications Lab. Brian was responsible for the NS/3000 implementation of NetIPC and PTOP.

INTERACTIVE LOCAL NETWORK FOR HP COMPUTERS. M - NET

G.F. MARTINA and S. NICOTRA
FRAKO INFORMATICA ED ELETTRONICA SRL
22052 CERNUSCO LOM. (CO) ITALY
Phone (39) - 039 - 599 539



Summary

Primarily intended for HP3000 systems it is now available for all HP Computers using RS232/V24 interface.

It is based on single ring made by coax and/or fiber optics and/or microwave link and specially designed interface boxes.

Where terminals/computers are located, a special (low cost) box is inserted into the link (no extra hardware/software necessary).

More than one computer can run any code (ASCII,EBCDIC,BCD...) at any mixed speed up to 9600 baud/async or 19200 baud/sync using any protocol MTS,BSC.. at the same time.

Cable length up to 1000 meters from terminal to terminal allow long distance, error free, optically insulated transmission.

M-NET units are produced by FRAKO srl

TODAY EXISTING METHODS AND LIMITATIONS FOR DATA TRANSFERS

The Point To Point connection

The simplest and the most used connection between HOST and peripherals is a point to point network in a star configuration

The network is fully interactive but every information channel requires a phisical support medium (cable) which has the following limitations:

- cable length up to 15 meters
- each peripheral needs a cable
- longer cables not supported for EMI problems
- no ground path protection
- upgrade and extensions are costly
- for longer distances modems are needed

Multipoint/Multidrop

To improve layout flexibility and allow longer cable length, multipoint can be used.

This configuration solves some of the point to point limitations but introduces the following new ones:

- Specialized software needed
- Specialized interface
- Specialized peripherals used
- Non interactive transmission (only one at the time)
- Delays due to polling
- Only one computer can talk on this link
- All peripherals need the same speed/protocol/code
- Maximum number of peripherals is limited by troughput

Local Area Networks

The local area networks are derived from previous methods adding new possibilities to data transmission.

Mainly they can be divided in two categories:

- Interactive local networks

- Contended local networks

Althougth interactive LAN's do not have pratical limitations the contended LAN's oblige peripherals to work at:

- same baud rate
- same code
- same protocol
- same interface
- they contend the use of the link
- only one at the time can transfer data
- computer manufacturers do not have any standard
- existing hardware is not usable on LAN's

INTERACTIVE LOCAL AREA NETWORK   M - NET
==

M-NET is a novel approach to local area networks, a flexible and low cost choice for small and medium systems.
It has been designed to overcome the following problems:

- Cable length	Up to 1000 meters from terminal to terminal
- Single cable	Only one cable connect all peripherals
- EMI protection	Using coax cable or fiber optics allow high rejection to EMI
- Grounding path	avoided by optically decoupling all units
- Upgrading	extensions are made easy and low cost
- Transparency	network is transparent to all peripherals allowing different codes,protocols,speed at the same time.
- Hardawre	Hardware independent. Virtually all peripherals which can be connected in poit to point connection can be used.
- Existing hardware	can be used without any limitation
- Software	Software independent. No special software is necessary nor user software modifications.
- Interactivity	All data are transfered without delay allowing full duplex and echo transfers All special codes are transferred
- Multiple CPU's	On the same network more than one computer can be be connected.
- Maximum troughput	All terminals can transfer at their maximum rate contemporarily up to a composite rate of 2.0 Mbaud
- Mixed link	Coax cable/Fiber optics/Microwave are allowed contemporarily in the same network

TEORY OF OPERATION

The phisical link consists of coax cable (or fiber optic, or Microwave unit) to form a ring, routed to all potential users points.

A main unit generates the syncronism (which is fed throgh the link) and recicles the data frames coming from the link.

An interface unit is inserted in the link where a terminal or a computer is to be connected.

The interface unit acts as a multiplexer, it samples all data bits and inserts them into a frame and then transmits the frame to the next unit via link.

Every unit receives the frame and recicles it into the ring, so the data frame can circulate indefinitely until it is intercepted by another interface unit programmed, manually or automatically by a controller, whit the same address.

In a data frame up to 60 data channels are allocated, each of them can transfer sampled data bits at 32Kbit/second rate. An address is associated to each channel.

Each channel is virtually independent and can transfer any data (single Point to Point, Multipoint sync/async, Bsc ...) exactly as on a cable.

The sampled data at the specified address (channel) is extracted from the frame and sent to the peripheral in RS232/V24 format being exacly the same of the original data delayed only by propagation time.

At his place a new data provided by the peripheral is inserted into the same frame.

FIBER OPTICS AND RADIO LINK

To avoid the ground loop each unit is optically insulated, and to obtain the gratest protection from interference, fiber optics can be inserted in any point of the ring.

Signals flowing trough the link are particurally suitable to be fed into a fiber optics electronics system due to their maximum 5 Mhz bandwidth.

Where it is impossible to lay out the cable, because the M-NET signals are particulary suitable to be applied to radio link, a simple Microwave unit for short distances can be supplied.

This configuration allows very high composite data transfer rates without any delay as with traditional statistic muliplexers.
All available data channels (60) can be routed via radio link.
It is phisically considered as a coax cable extention.

NON COMPUTER DATA TRANSFERS

The very strong difference between M-NET and other existing networks is not a communication arrangment but the interface with the logical devices.

By making a correct sample of RS232/V24 signals the data can be transferred to the network with an absolute independence from the speed.

So the 9600 baud or 2400 baud or ON/OFF signals can be mixed and transferred trough the network without any special option or without making any configuration on the interface.

Data to be transferred no necessarilly has to be computer like data but can be any ON-OFF signal, low frequency pulses, switch generated pulses: the only requirement is that the electrical format of the data should comply RS232 or V24 specification.

The electronics is very simple so the cost per connection can be many times fewer than any traditional solution.

FUTURE ENHANCMENTS

M-NET has been designed interactive and transparent on all the available channels to be immediately usable with existing peripherals.

This statement does not imply that it can never be a switched LAN. A special box (under development) will allow any peripheral to ask the link to 'switch' the data from a certain time on to another channel.

The switch request sequence could be simple :

BREAK //*3

so it can be done on existing hardware even manually.

No special software will then be necessary but obviously all peripherals to be switched will have the same speed, code ecc.... but at the same time all non switching peripherals will continue to run as today do.

Biography

Sebastiano Nicotra
Born in the 1949 in Catania (Italy), he receaved the degree of Doctor in electronic engeneering from the University of PADOVA. From 1973 to 1980 he was with TELETTRA SPA as Radio Link Designer. Since 1980 he works as consultant in Microwave, Brodcast, Telecomunication and Data Transmission.

G.Franco Martina
He was prouction line responsible in an IBM factory, then in Product Test Laboratory as Data transmission specialist.
He joined Hewlett-Packard and become HP/3000 Specialist at the very beginning of HP3000 production.
M-NET idea comes from his actual needs as OEM.

TRAVELLING LIGHT FOR DATA COMMUNICATIONS

David Roy Williams.

VRS Software Ltd., London, U.K.

Summary

Fibre optics have been around for some time, in fact much longer than electronic computers, however they have only recently come to notice. They are now really poised to take off and make a major impact in the communications field. As far as the computer user is concerned they can affect the whole way in which he or she uses data communications.

But what are they? Just how does a system based on fibre optics work? What are the advantages and disadvantages of a fibre optic system? This paper will TRY to answer these questions and to give some idea of what is on the market.

They work by a phenomena known as total internal reflection. This was first discovered in 1879 by the British physicist John Tyndall who demonstrated that light shining in a stream of water emerged very much as it had entered. However it wasn't until 1954 when Van Hell, Hopkins and Kampany published articles describing how this phenomena could be used in glass and plastics to send undistorted images over a distance. Since then the industry has quietly but successfully progressed to the stage it is at the moment. The actual expression total internal reflection means in real terms that 80 to 90% of light is reflected in the fibre so in the data communications field, data can be sent with very little degradation and a great deal of accuracy.

ADVANTAGES OF FIBRE OPTICS

* Immunity from RFI and EMI
* No Crosstalk
* Secure transmission - no radiated RF, hard to tap
* Elimination of sparking and fire hazard
* Electrical isolation - no ground loops
* High strength/weight ratio
* Wide bandwidth
* Low losses
* Corrosive resistant

FIBRE OPTIC THEORY AND PRACTICE

In this paper I attempt to explain some basic optic theory, present day problems involved in design and hope to aid the understanding of manufacturers data sheets. Examples of calculations encountered in the design of Fibre Optic products are also given.

TOTAL INTERNAL REFLECTION

When light meets a boundary between two transmitting media it will experience refraction - a change of direction, according to Snell's Law (i) (Fig.1).

$$n_1 \sin \theta_1 = n_2 \sin \theta_2 \qquad \text{(i)}$$

where n is the index of refraction for each material given by

$$n = c/v \qquad \text{(ii)}$$

where c is the speed of light in a vacuum and v the speed of light in the medium.

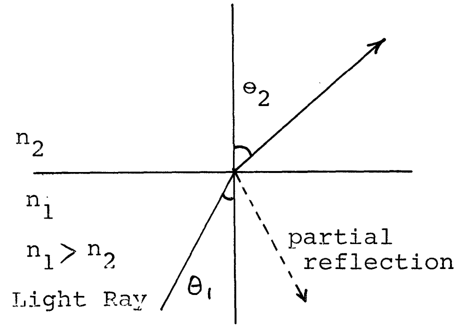


Fig 1

REFRACTION

An angle exists ($\theta_1 = \theta_c$) where the light will no longer pass into the second medium but is reflected back from the boundary (Fig.2). In fibre this marks the minimum angle at which light will usually propagate. This phenomenon provides the basis of all optical transmission and is called Total Internal Reflection (TIR). The minimum angle that supports TIR is given by

$$\sin \theta_{min} = n_2/n_1 \qquad \text{(iii)}$$

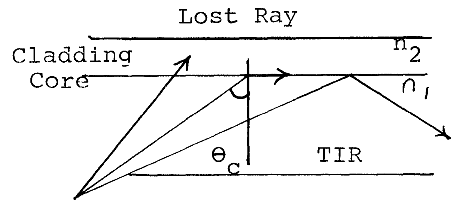


Fig 2

TOTAL INTERNAL REFLECTION

FIBRE THEORY

The index profile of a fibre describes the refractive index of the cross-section (Fig 3.). There are three types of fibre: multimode step index graded index and single mode. The previouse example describes step index, referring to the abrupt jump in index between the core and cladding We can now consider a number of rays propagating along the fibre following different paths. Known as modes, they can vary from the axial, lowest order, to the highest order that just fulfils the minimum angle criterea (iii) (Fig. 4.). But rays do not exist; this is a simple representation that avoids the rigour of electromagnetic field solutions and Maxwell's equations but if we use a parameter called normalized frequency f_{norm} given by

$$f_{norm} = \frac{nd}{\lambda} \sqrt{n_1^2 - n_2^2} \qquad \text{(iv)}$$

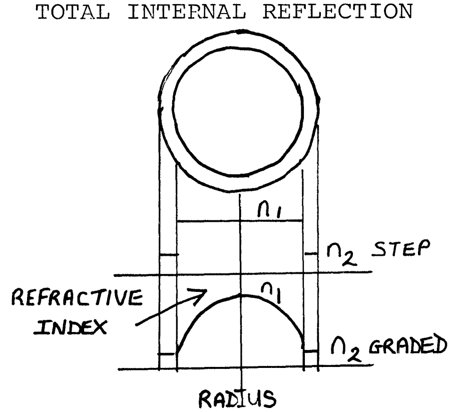


Fig 3 INDEX PROFILE

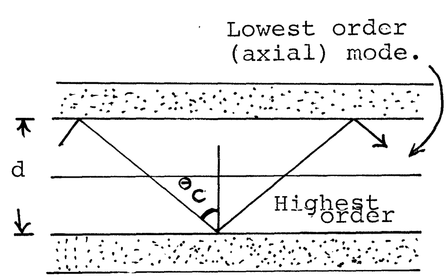


Fig 4 EXTREME MODES

(where d is the core diameter)

we can use the modal volume formula

$$N = \tfrac{1}{2}\left\{f_{norm}\right\}^2 \qquad (v)$$

to give the theoretical number of modes, N, and will use its concept again later.

There are three factors that contribute to fibre loss; scattering absorbtion and mode coupling. Scattering is mainly Rayleigh scattering which is an intrinsic material property and is inversely proportional to the forth power of wavelength. Other scattering is due to imperfections in the purity and physical dimensions of the cable which excite higher order no propagating modes. Absorbtion peaks appear around the range of most interest communications - namely wavelengths at which most low cost sources radiate $\sim$900 nm. Impurities such as transition metals and -OH groups have a fundamental peak at 2.7μm ($\sim$900nm is the third overtone). High quality cables are therefore made of water free materials.

During propagation, modes tend to interchange energy levels or couple. Although this has an improvement in the effects of dispersion (see later) it is a loss process.

FIBRE TRANSMISSION

Reassuming the ray model of fibre propagation, it will be seen that a path difference exists between the axial and the highest order modes, this leads to frequency limitation and distortion called Modal Dispersion.

e.g. for a 1 km length of 50μm fibre $n_1 = 1.48$; $n_2 = 1.46$; $\theta_c = 80.6'$ the difference in path length is 14m, a delay of 69ns ($v = c/n_1$).

On manufacturers data sheets the Modal Dispersion is given in ns/km or MHz/km. By inspection of (iv) we may conclude three methods of reducing the effect of modal dispersion;i) reduce core size ii) increase the wave-length or iii) decrease the difference between n_1 and n_2. Reducing core size can result in a fibre with no modal dispersion. If f_{norm} is less than 4.405 only the axial mode will propagate (rays do exist!) and would offer the ultimate bandwidth. Problems arise though in handling a fibre 3μm diameter, launching sufficient power into such a size and connecting the ends. (Human hair is typically 60μm in diameter). Increasing the wavelength increases the attenuation (Fig 5.) but scope exists at the second window at 1.3μm.

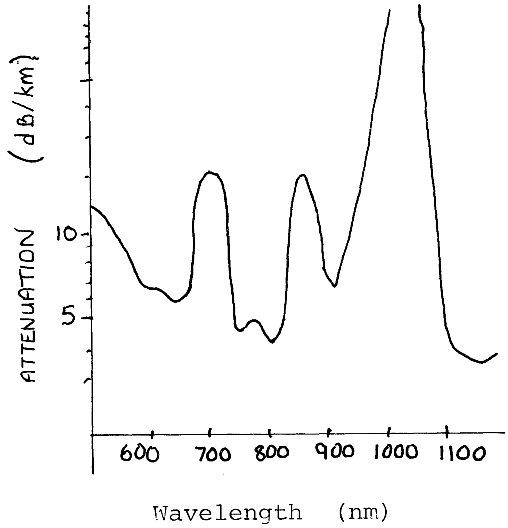


Fig 5

TYPICAL FIBRE ATTENUATION

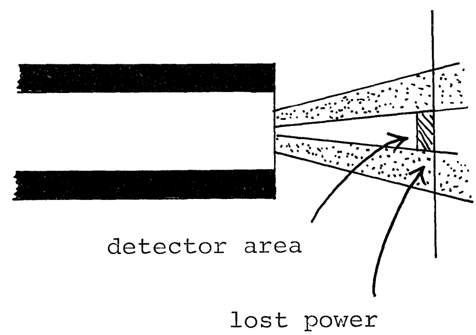


Fig 6.
AREA MISMATCH

The final parameter requires that the difference in the refractive indices is minimised. This is a parameter under the control of the manufacturer when dopents are included into the silica. The ratio must alway exceed one in order to maintain TIR.

Graded Index fibre combines the advantages of single mode fibre with the easier handling of a larger diameter. Light is gradually bent back towards the centre of the fibre as the refractive index varies (parabolically) with radius, its speed increasing as it diverges reducing modal dispersive effects.

The impulse response pulse width is dependent upon the difference between the core index and cladding index for step index fibre but for graded index fibre is proportional to the square of the difference. Because present day fibres have a very low difference (typ. 0.02) graded index fibres show a considerable improvement over step index fibre.

A second form of dispersion occurs because of the frequency dependence of refraction. It is known as Material Dispersion and is source bandwidth dependent. The degree of refraction is determined by the frequency content of the ray (as water refracts light in the atmosphere to give a rainbow). After repeated refractions the ray will be spread into a broad band of signal despersing the pulse. On data sheets it is usually shown as the 3dB fibre bandwidth or displayed in graphical form for a 1km length.

The launch conditions must be taken into account when considering this fibre performance figure.

SYSTEMS ATTENUATION

Apart from the inherent fibre attenuation quoted on data sheets there are other system losses that good design will minimize. The physical relationship of two fibre ends has an important effect on attenuation coupled with one final property of fibre. Light will not pass effeciently when :- i) the emitting area is greater than the receiving area (Fig 6.) ii) when there is a NA mismatch. As seen before light will only propagate along a fibre as long as it fulfils the minimum angle criterea (iii) this defines an area of acceptance at the end of the fibre outside of which any incident ray will not propagate. The standard measure of this angle given on data sheets is the Numerical Aperture (NA) and is a measure of the half angle of the acceptance cone. (Fig 8). It is given by

$$NA = \sin \theta_A = \sqrt{n_1^2 - n_2^2} \qquad \text{(vi)}$$

Thus even if two fibre ends are butted perfectly together there will be losses if the NA of the receiving fibre is smaller than that of the transmitting fibre. i.e. the higher order modes will propagate up to the joint and will be "stripped" as they cannot pass onto the second fibre that has a smaller modal volume.

A very much smaller loss exists at every joint due to the light reflected from the inside end surface. It is called Fresnal Loss and has a typical value of 0.2dB per surface.

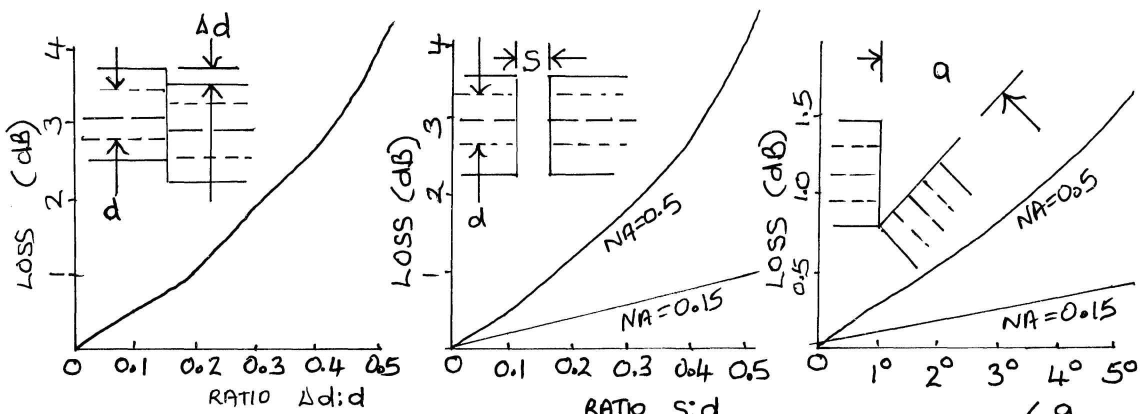


Fig 7 FIBRE MISMATCH LOSSES

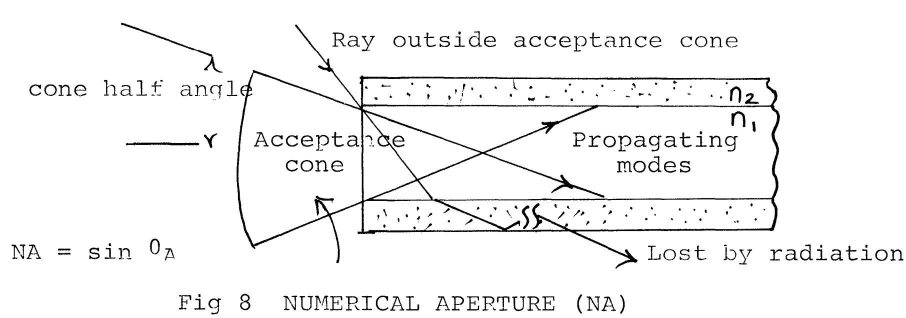
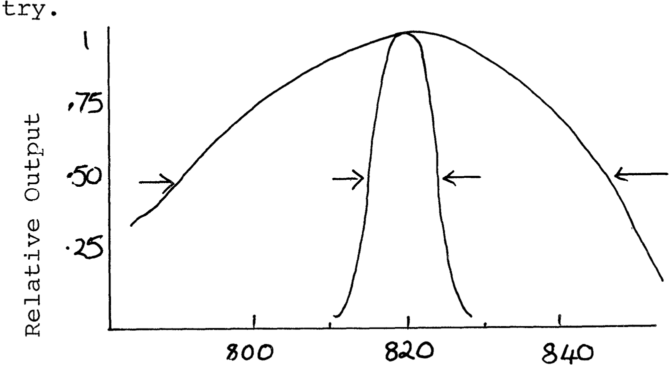


Fig 8 NUMERICAL APERTURE (NA)

SOURCES

We are now in a better position to analyse the features of available emitters. There are two forms of LED the Burrus diode and the edge or side emitting diode. These diodes are similar in construction to standard LEDs or may be designed to give a higher radiance in one direction. Sometimes a short length of fibre, called a pigtail, may be positioned and cemented to the region of highest output. The typical output from Burrus Diode is 1mW and 7mW is expected from an edge/side type. LEDs are relatively cheap, have good lifetimes (10^5 hours) but have wide spectral bandwidths, leading to dispersion problems and high NAs limiting the useful available light, (see example later.) The LED is easier to use as an analogue device or lower speed digital device as the output power varies linearly with input current.

Many of the disadvantages of the LED are overcome by the use of the ILD. ILDs are faster, give greater output, have improved spectral performance, (Fig. 9) and smaller NAs ideal for digital transmission but are non-linear, temperature dependent, expensive, have shorter lifetimes (10^4 hours) and require greater complexity in the drive circuitry.



Wavelength (nm)

Fig 9
SPECTRAL EMISSION

3 dB point
LED 40nm typ
ILD 5nm typ

RECEIVERS

The receiver is the heart of the system. Here too the designer has a choice of detecting elements:- i) the PIN diode, ii) the Avalanche Photo Diode (APD), and iii) Phototransistors.

The mnemonic PIN is derived from the semiconductor construction. A large lightly doped area is sandwiched between a p and n type layer. Photons absorbed in this region generate an electron-hole pair that are separated under the unfluence of the high internal electric field and create a current in the circuit. In the APD the internal electric field is so large that it imparts to the liberated electron sufficient energy to ionize surrounding atoms - resulting in an avalanche effect.The responsivity of a receiver is given as the output voltage or current arising from the incident light power in Volts/Watt. Because of the internal gain effect of the APD it is up to 20 dB more sensitive than the PIN diode. The penalty being the high potentials required by the APD (sometimes in excess of 300v). The APD is also faster and costlier than the PIN diode. The speed of the PIN diode is restricted by the CR product of the diode capacitance and resistance given on data sheets and the load resistor. A trade off exists between speed and sensitivity.

Phototransistors are sensitive but restricted to very low speed operations.

In all cases the sensitivity is limited by the noise current. The level of noise is dependent upon signal level and temperature. Two standard performance figures are the dark current (the current flow in the absence of light) and the Noise Equivalent Power NEP (expressed in Watts/ (Hz).

The absolute power level of the minimum detectable signal (MDS) is given by

$$MDS = NEP\sqrt{B} \qquad \text{(vii)}$$

where B is the detector noise bandwidth. The signal to Noise Ratio (SNR)/Bit Error Rate (BER) is given in Fig 10.

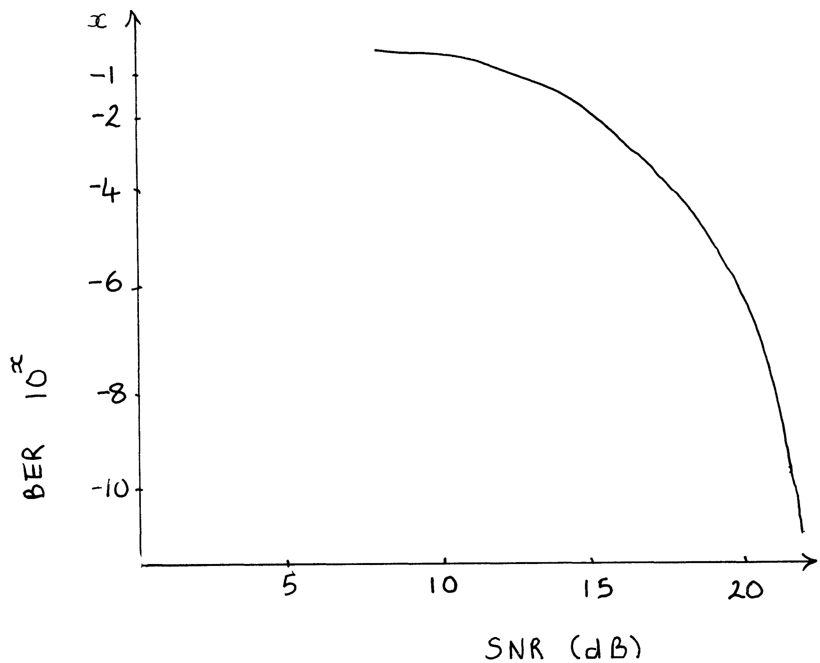


Fig. 10 BER- SNR RELATIONSHIPS

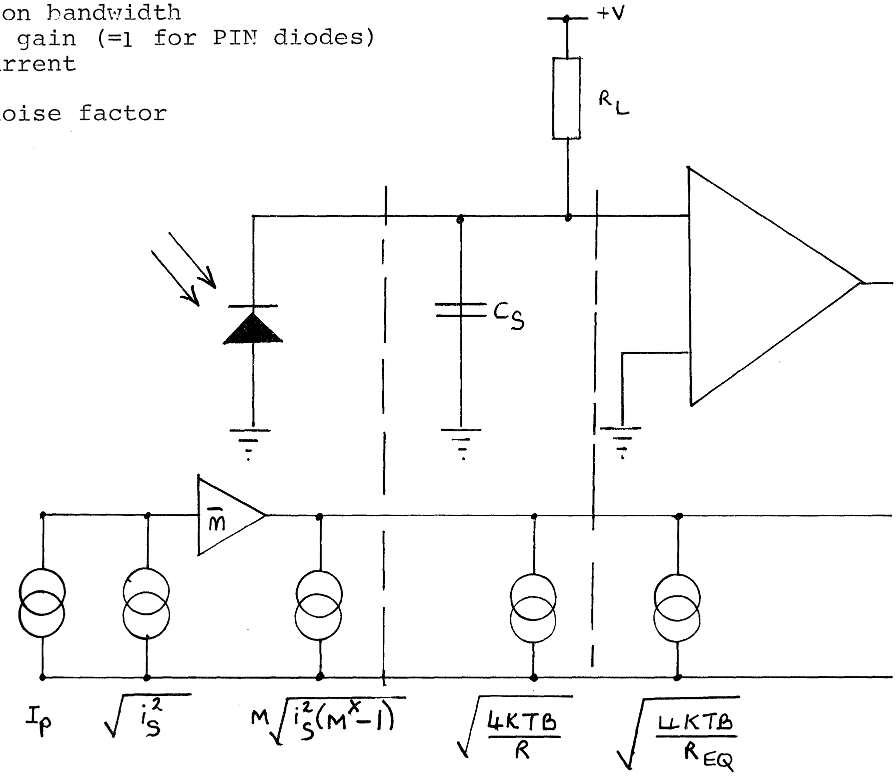
Note the significant reduction in the BER for a small increase in SNR at BERs of 10^{-9}. A simplified receiver model is shown in Fig 11 where

P_L is the incident light

I_P the resulting current

e the magnitude of electronic charge
h is Plank's constant
f is optical frequency
n quantum efficiency
r responsivity
i_s^2 shot noise (mean square)
B modulation bandwidth
M avalance gain (=1 for PIN diodes)
i^2 noise current

x excess noise factor



$$\sqrt{\overline{i_n^2}} = \sqrt{M^2 + x\overline{i_s^2}}$$

$$I_p = \eta \frac{e}{hf} P_L$$

$$\sqrt{\overline{i_s^2}} = \sqrt{2eI_p B}$$

Fig 11 RECEIVER DESIGN

Biography

David R. Williams.
is a consultant in the London Office of VRS Consulting Ltd. David has over 10 years experience in Data Processing and 5 years data communications experience. David is responsible for the installation into HP3000 sites of multiplexers manufactured by VRS Communications Ltd.

OA - OFFICE AUTOMATION AND NETWORKS

ERGONOMICS OF PERSONAL COMPUTER SOFTWARE IN DATA COMMUNICATION NETWORKS

Marc Burch
Hewlett-Packard, Santa Clara, USA

A presentation to the HP3000 International Users Group, European 1985 Conference, Amsterdam, The Netherlands March 31 - April 5

Abstract:

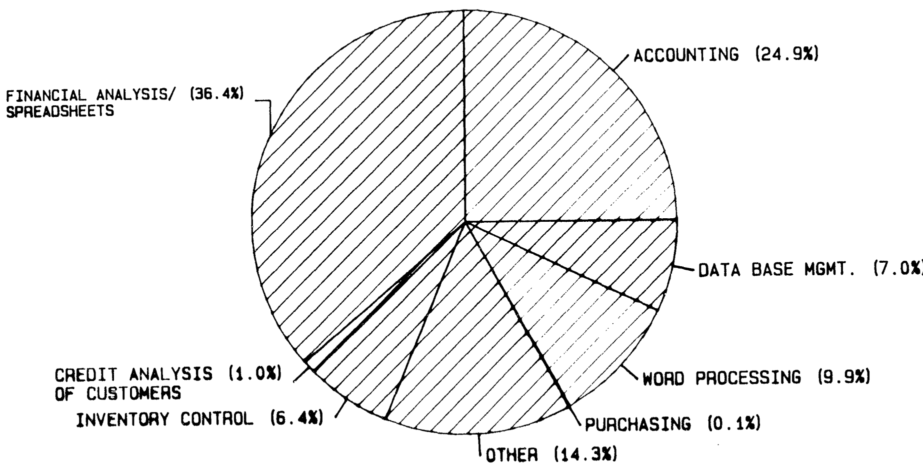
As the personal computer market continues to expand, more attention must be given to the end user and their needs. Thus, Independent Software Vendors (ISVs) must design and develop not only user intuitive, intelligible software but software that will allow true multi-user, multi-tasking with a wide range of data communication networks and peripherals.

There is a need for standardization and a common methodology (i.e. windows, touchscreen, mouse, voice recognition, artificial intelligence, etc.) among Independent Software Vendors to allow the sharing of information. Users should not be burdened with the task of having to develop expertise with a variety of different personal computers, minicomputers, mainframes, and software applications.

The presentation will describe these software and network trends and what Independent Software Vendors are planning for the future. The end goal being the productivity increase through end users becoming more effective and efficient in using personal computers in networked systems.

Independent Software Vendors (i.e. Lotus, Microsoft, BPI, Open Systems, PFS) will play an important role in the development of networks by increasing the productivity of business professionals, managers, and support personnel. Many of the current application software (i.e. financial analysis / spreadsheets, database, word processing, graphics) were not designed to work within a network.

PRIMARY PERSONAL COMPUTER APPLICATIONS
IN BUSINESS FOR THOSE WITH PCs



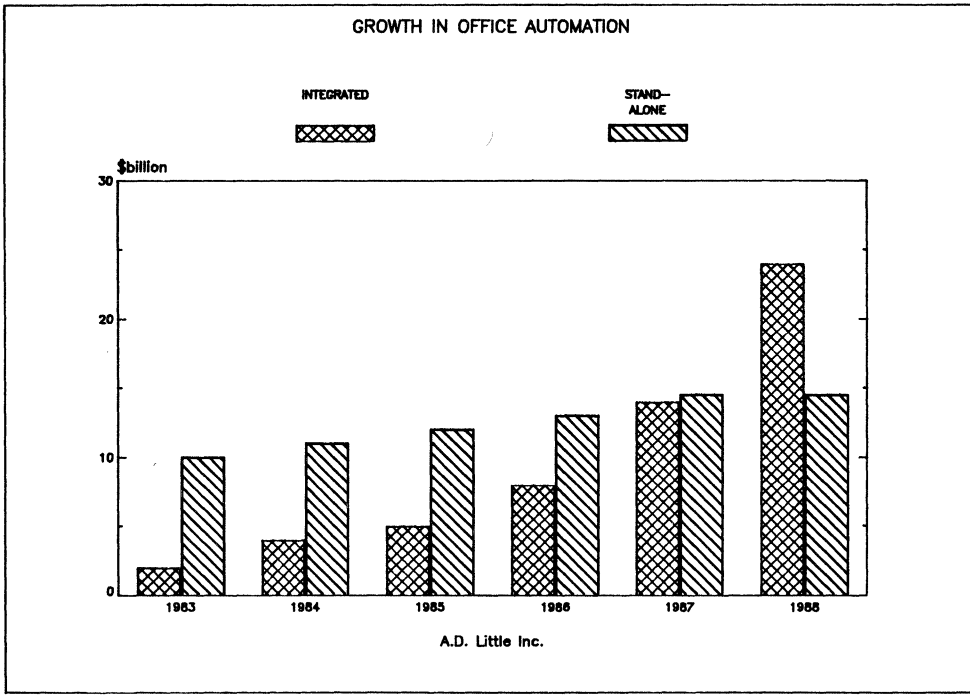
SOURCE: DATAQUEST, INC.

This is starting to change as the ISVs look at the very diverse and exciting potential markets. The markets will be in four main areas:

- o office environments / automation
- o manufacturing plants and complexes
- o transaction services (financial distribution and reservation systems)
- o educational establishments

The largest market will be in the office environments of major corporations, where automation of office functions and increasing use of electronic storage media will require high capacity network systems with multiple channel capabilities.

Studies by the A. D. Little Research Organization expect growth in the office automation market of $11 to $36 billion by 1988. Even more impressive is the expected five year growth of $1 to $23 billion for Integrated Office Systems (i.e. data processing, personal computers, office automation and communications).



Every office in every company, from the smallest to the largest, in every industrial nation in the world is a potential candidate for data communication network. Most observers believe networks will become as widespread in the office as the telephone system. Applications for intra-company networks include:

- Data and File between Workstations
- Peripheral Sharing
- Electronic Mail
- Access to Data Bases and Computing Power of a Mainframe Installation
- Video Conferencing
- Voice Store and Forward
- Material Flow Control
- Inventory and Cost Control
- Engineering
- Information Access
- Reduced Communications Costs
- Others

In manufacturing environments, networks will be increasingly used to connect terminals, office data and word processing equipment and process control and surveillance systems. These environments include:

- Factory Data Collection
- Building/Environmental control and monitoring (Commercial Buildings, hospitals schools, etc.)
- Process control, computer aided manufacturing (CAM) and Computer Aided-Design (CAD)
- Energy Managementations
- Aircraft Communications
- Photo composition
- Others

Transaction services has a great need for networks to help with the following applications:

- o Financial Transactions
- o Point-of-Sale Systems
- o Reservation Systems (Hotels, Airline, etc.)
- o Others

In educational establishments, networks will be used to connect the ever increasing number of microcomputers in use for computer education, computer aided instruction (CAI) and research purposes. However, since most high schools and elementary school budgets are small, they typically purchase only two or three discs and a few printers, and make them available to all other computers via some type of local area network or multiplexing arrangement. This will change with the development of new storage and information sources that will significantly increase the use of computer/communication networks within the school system. Other applications include:

- o Scientific
- o Laboratory

Several different technological developments allowed the evolution of new and exciting approaches to networking.

First, the much heralded advances in LSI (Large Scale Integration) and VLSI (Very Large Scale Integration) technology (Computers or Microprocessor chips) made it economically possible to distribute minicomputers and a whole array of intelligent, task-oriented peripheral equipment.

Second, many important gains in communication protocols have been combined with this very sophisticated LSI and VLSI technology in nodes and network interfaces to provide the functions and performance levels needed for network communications.

Third, much more of the lower level protocols responsible for interfacing to the network and controlling network functions are being designed into the network hardware. As a direct result of this:

- o Nodes have less overhead associated with network control and now can better perform their designated functions.
- o The network hardware components can be mass produced in volume and at lowered cost, simplifying connection to the network and encouraging greater participation by users and equipment.
- o Communications protocols that become standardized can be incorporated into the network hardware, thus allowing a variety of nodes from many different manufacturers to communicate without the need for expensive custom interfaces, and giving users increased vendor hardware and software independence.

Forth, software ergonomics that allows users input through touch, mouse, voice, keyboard and friendlier screens (i.e. Pam, Windows, Menus).

Fifth, multi-user software applications that greatly facilitate information exchange and resource sharing.

The trend today is towards greater numbers of separately identifiable computer and software based systems. This is due to the decrease in prices, computer processing components, and multi-user software sharing within an organization. This has caused a greater awareness (at both individual and organizational level) of the benefits of convenient interconnection of systems, often supplied by different manufacturers, and ISVs, to achieve coordinated access both to common resources (such as databases, analysis programs, development tools and office-style memos and reports) and to sophisticated or specialized (and therefore, expensive) resources such as mainframe processor, file archive and management facilities, printers, plotters, etc. There is also a need for overall management control of system proliferation, duplication and/or dilution of effort, synchronization of activity and any other factors that can hurt an organization resources and productivity.

Let's now look at the need to set up a compatibility standard for the sharing of data. Standards make it technically, ergonomically, and economically feasible for independent software vendors to write network based software. This standard is in the process of being defined by the International Standards Organization. The ISO has defined an Open Systems Interconnection (OSI) reference model which defines a network architecture. The OSI model is made up of seven layers.

7 - Application
6 - Presentation
5 - Session
4 - Transport
3 - Network
2 - Link
1 - Physical

Open Systems Interconnection Model

1. The Physical Layer - defines the mechanical and electrical interfaces to the network and the bit level data flow. Includes the different software device drives.
2. Data Link Layer - defines the individual packet format and sets up an error-free communication path between network nodes, checks received data for errors, ensures proper sequence of transmitted data and controls access to the channel.
3. Network Control Layer - addresses messages, governs the switching and routing of information between networks, and performs the gateway function for data bound for other networks.
4. Transport Layer - provides end-to-end data integrity between processes, allowing processes to exchange data reliably and sequentially, independent of which systems are communicating or their network location.
5. Session Layer - establishes and controls communication sessions between specific nodes in the network and handles coordination between processes.
6. Presentation Layer - encoded data is converted into usable formats which enable display on terminal screens and printers. Functions include data formatting, code conversion, text compression and encryption.
7. Application Layer - user interface to the network services and software applications that provide direct services to the users. The layer is very important to the ISVs as the following functions are provided:
   - o Resource Sharing - users and applications share local and remote peripheral devices.
   - o File Transfers - data files are copied from one system to another, and than formatted and converted for the end user.
   - o Remote File Access - this allows access to files residing in a remote system and the ability to perform read, write, and update data on the file.
   - o Database Management - allows manuipulation and management of databases distributed in a network by providing a sophisticated user interface. It also provides transparency of operational complexity to the user.

At present, standards are well established for layers 1-4 only. It's critical for Independent Software Vendors that layers 5-7 are defined within the next two years. Having well defined upper layer standards will allow the end user to transfer a file created with LOTUS, the ability to process it with MULTIPLAN. It is the application software, more than the network software and hardware, that allows the end user to resolve a business problem.

Application software in a network can make you more productive in your job if you have access to programs and data from your coworkers. Let's talk about three levels of software that are used in a network (i.e. 3COM)

- o Networked Software
  - tailored for multiple users
  - file / record locking
  - usable on shared volumes
  - licensed and priced accordingly

- o Networkable Software
  - usable on private volumes
  - single-user only
  - public volume use may have legal or data integrity implications
- o Partially Networked Software
  - store data volumes at server
  - may require system disc at user PC
  - incorporate copy protection

Independent Software Vendors are being encouraged to tailor their single user software products to run in a multi-user environment as more companies network their computer together.

In the near future, not only will ISVs be developing more multi-user software products but these products will take advantage of the ergonomics of windows, PAM, touchscreen, mouse and voice recognition, etc. ISVs must also develop software that is:

- o Intuitive
- o Natural language based
- o Knowledge based
- o Artificial intelligence based

Also important is having an on-screen help facility that will allow the end user to ask for help. After exiting from help, the software should return the user to where he/she left off. The future software trend is to write software applications that are knowledge based with help facilities. These knowledge based applications would sense if a user was having problems and than automatically put the user into a help screen. The software would "teach" the end user.

Software applications must be developed to be intuitive based for the end users. Documentation would not be needed as the user would have quick and ready insight to the application. For the few applications that would not be intuitive, documentation could be on-line. That would save the cost of printing the documentation and would allow the ISVs to mail out updates on a disc. The end user would be able to not only ask for a help menu but also for a documentation menu. This is a step towards artificial intelligence where the software and end user "learn" together.

Independent Software Vendors must develop software applications that have the following characteristics:

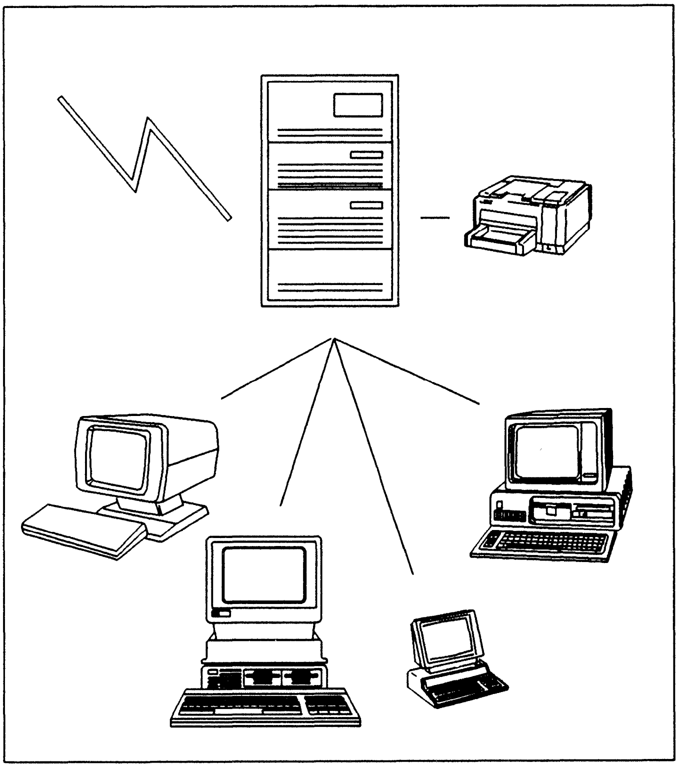
- o Ease of use
- o High performance
- o Flexibility
- o Ease of learning
- o Security
- o Ease of installation
- o Expandability
- o Ease of support
- o Software drivers

Software applications should also include: learning aids, such as tutorials, key labels, application templates, quick reference cards and other aids which facilitate the learning process.

Software vendors must also look at the following three personal computer networking products and decide where to put their limited development resources.

1. Terminal emulators to provide connections to mainframes and public sources for stand alone PCs.
2. Local area PC networks to support multi-user applications and to reduce costs for small businesses. The networks will have concurrent file and peripheral sharing capabilities. (i.e. HP Advance Net)
3. Mainframe to micro networks to optimize PC and mainframe use in medium to large corporations. The networks will have upload/download of data and easy access to mainframe applications.

Independent Software Vendors are also developing software for Hewlett-Packards personal productivity center.



Office and ISV software solutions along with Hewlett-Packard proprietary software can be found in the personal productivity center. This enhances the office by showing that the integrated office systems represent the convergence of office automation, data processing, personal computers and communications.

The degree of software intelligence needs to increase by adapting to the new standards as they emerge. Also, the trend toward increasingly complex personal computer graphics and multiple windowing. In summary, the future of Independent Software Vendors will be built on providing ever increasing services to the users to help them become more effective and efficient in their jobs.

Biography
Marc Burch
is the Independent Software Vendor Applications manager at the Hewlett-Packard Personal Software Division in Santa Clara, California, with responsibility for vertical markets, software distribution and application software for the touchscreen and portable personal computers.

BUSINESS DECISION SUPPORT SYSTEMS (MDSS)

Wesley S. Christman
Huck Manufacturing Company
Irvine, CA USA 92718

I. SUMMARY

II. PREFACE

III. INTRODUCTION

- A. THE CHANGING ENVIRONMENT
  1. CENTRALIZED vs. DECENTRALIZED
  2. OPERATION vs. MANAGEMENT CONTROL
     - a. STRATEGIC PLANNING
     - b. MANAGEMENT CONTROL
     - c. OPERATIONAL CONTROL
- B. ORIGIN OF MDSS
- C. OBJECTIVES AND BENEFITS

IV. BUSINESS PROCESSES

- A. PLANNING & CONTROL
- B. PRODUCT/SERVICE
- C. RESOURCE

V. DATA CLASSIFICATION

- A. BUSINESS/PROCESS ENTITY
- B. CONSOLIDATIONS

VI. MANAGING MDSS RESOURCES

- A. MDSS CONCEPTS
- B. MDSS OVERVIEW
- C. DATA BASES
  1. ELEMENTS
  2. PROCESSING

VII. BIOGRAPHY

VIII. EXHIBITS

I. SUMMARY

This paper will introduce the notion of a Management Decision Support System software tool for purposes of supporting the executive management decision-making process.

The feature is visualized in a general framework encompassing the strategic planning and management operation control functions while embracing business processes, data classes and relational data base technology. Moreover, the selection of distributed data processing represents an exciting challenge for MDSS design, development, and implementation. It is reasonable to expect that, in this environment, MDSS as represented herein be subjected to alterations and modifications to specific components and techniques which will have a definite impact on its final configuration. Nevertheless, it can be viewed as a responsible contributor to improving the efficiency and productivity of the executive management function.

Just as MRP II Application Systems software packages migrated toward integration of numerous business functions, more than likely MDSS will evolve along the same lines. The result of integrating MDSS with the applications software represents a "total system" and another step in the evoluation process of MRP II Application Systems.

II. PREFACE

Business Management Decision Support Systems (MDSS) as a structured approach can provide an effective "tool" for management that will satisfy the near and long term planning and decision-making processes of a corporation. Its primary infra-structure offers an on-line interactive environment that is very "user friendly" and manipulatable while giving considerable respect to the human dynamics of a senior executive with precious little time for the planning and decision-making activity.

Although Generic Decision Support Systems (DSS) have been widely written about in recent years, MDSS applications are relatively new. The function is rather scarce as a totally integratable feature of most major business systems software packages currently available. "Decisionmaker" to be released by ASK Computer Systems, Inc., Los Altos, CA, in early 1985, incorporating some basic concepts of MDSS, represents an aggressive introduction to Management Decision Support Systems. It is totally integrated and concentrates on the individual modules data bases as its primary information resources. A general discussion of "Decisionmaker" is provided in the MDSS overview of this paper.

This paper is organized so that readers familiar with the Generic MRP II Business System can readily visualize MDSS applications, while others may relate to its principles, concepts, methodology, overview, and finally, appropriate alternatives for its overall capabilities.

The INTRODUCTION explores the relationship of MDSS and the changing environment associated with the trend toward decentralization of business systems. Further, the origin of Business Management Decision Support System (MDSS), along with its objectives and benefits, are discussed in order to establish the basic premise for its applications.

BUSINESS PROCESSES address a logical format or business organization that when executed permits effective, efficient, and meaningful application of MDSS principles.

DATA CLASSES deal with categorizing logically related information dedicated toward developing data bases with a long life, thus precluding major reworks in support of customization and future enhancements.

MDSS CONCEPTS AND OVERVIEW outlines a methodological overview and continuity of activity encompassing the feature. Concepts and a general form are presented so that the reader can understand the basic principles without becoming embroiled in the details of specifics of MDSS applications.

DATA BASES outline the information architecture in order to define the data created, controlled and used. Further, the relationship of MDSS to the business systems data elements and essential processing to support it are discussed.

This paper and presentation was made possible through the understanding, encouragement and support of many people who among them are Mr. Lawrence K. "Lorrie" Volland, Marketing Rep., ASK Computer Systems; Ms. Deon Durant, Project Manager, ASK Computer Systems; Ms. Christine Tange, Business Systems Specialist, Pulse Engineering, Inc., Mr. Donald Davis, Vice President and General Manager, Aerospace Fastener Division, Huck Manufacturing Company, and last but not least, my company and especially my boss, Mr. Roy Taylor, Vice President of Finance, Huck Manufacturing Company.

III. INTRODUCTION

In an article entitled "Corporate Information Systems in Transition" Ed Witter states, "Information technology is advancing faster than ever before, presenting an array of opportunities and threats that corporations must assess and manage." He goes on to say, "At the same time centralized control over the use of information resources is eroding." It is in this environment that executive management must adjust quickly and apply appropriate procedures in order to have up-to-date information available at all times for the decision-making process.

Traditional centralization of the information resources entails a cohesive group of individuals engaged in working with very large and expensive pieces of equipment known as the mainframe computer, BIG HUMMER, or other affectionate labels of identification. Very few people other than this cohesive group know or understand what goes on in the corporations's data center. Because of this, the entire function takes on a mystical aire, while individuals in the group are acknowledged as the experts. Very rarely are their absolute responsibilities or authority ever questioned.

"Micro-Computery" coupled with rapidly decreasing costs for comparable capabilities and computer power are drastically changing the information resources environment. The increasing trend toward decentralization has created a new breed of user with a hunger, innovativeness and capability to capture direct and unilateral access to all available information resource technology. Certainly, Ed Witter was accurate when stating in the same article, "Management of information resources has entered a new era of complexity and change." This then is the challenge of Management Decision Support Systems.

The emergence of the decentralized environment is overwhelmingly supported by new technology encompassing applications that just a short time ago were unheard of. This technology is single handedly responsible for what John Naisbitt in his book, "Megatrends" identifies as the information revolution which has transported us from an INDUSTRIAL SOCIETY to the INFORMATION SOCIETY.

DECENTRALIZATION has several implications relative to the planning and control functions within an organization.

First, different characteristics of information are needed in a decentralized environment. Typically, lower levels (the User Community) need detail, volume and frequency. The next level (Management) needs summaries, exception reporting and inquiries, while the top (Executive Management) needs cross-functional summaries, special requests and information that supports "what if" analysis. Because of this, it is inappropriate to expect one single applications system to accommodate all activities for all levels of management.

R. N. Anthony in his article, "Planning and Control Systems," Harvard Business School Research 1965, proposes that three distinct but concurrent planning and control levels exist in any organization. These are:

- Strategic Planning, the process of deciding on objectives of the organization, the resources used to attain these objectives, and the policies that are to govern the acquisition, use, and disposition of resources.
- Management Control, the process by which managers assure that resources are obtained and used efficiently in the accomplishment of the organization's objectives.
- Operational Control, the process of assuring that specific tasks are carried out effectively and efficiently.
- Resource Management, while not a control or planning function per se, is also desirable because it gets across organization boundaries -- vertically across management levels and horizontally across functional levels. In view of this, a framework of a management decision support system (MDSS) can be established and applied within this framework. Exhibit No. 1 illustrates the characteristics of the planning and controlling function while giving consideration to structuring the MDSS.

The origin of the Business Management Decision Support System (MDSS) presented herein is based in part on an article entitled "Development Tools for Decision Support Systems," Robert Bonczek, Clyde Holsapple, and Andy B. Whinston - PRESPECTIVE ON INFORMATION MANAGEMENT, May 1982. This article discusses the traits, framework, and organization of the Decision Support System while suggesting "that the next major development in the DSS field will be the appearance of general software tools for facilitating the creation of application-specific Decision Support Systems." It is no coincidence that MDSS is an expansion of DSS, thus providing an infra-structure for the exploration of development and application of the feature.

The application of the approach and methodology of MDSS offers benefits to the following essential management groups concerned with effective planning and control. These are:

Executive Management

- A defined logical approach to aid in solving management planning and control problems from a business perspective.
- The ability to aid in the assessment of future needs based on business related impacts and priorities.
- Information and data that are relatively independent of organizational structure.

Functional and Operational Management

- A defined, logical approach to aid in solving management and operational control problems.
- Consistent data to be used and shared by all users.
- An assurance of top management's involvement that establishes the informational objectives and direction, as well as agreed-upon priorities.

MDSS as an idea encompasses customized features while integrating to business application system software offers a unique ooportunity to build application-specific decision support systems. It is hoped that MDSS will evolve within the broad outlines presented in this paper. In any event, its introduction at this time suggests an intensification of interest in the features and capabilities of Business Management Decision Support Systems.

IV. DEFINING BUSINESS PROCESSES

For purposes of Management Decision Support System (MDSS) considerations, business processes are defined as groups of logically related decisions and activities required to manage the resources of the business. These processes are conceived without regard to organization responsibilities, thereby providing an objective environment that leads to:

- A Management Decision Support System (MDSS) with a large degree of independence from organizational restriction and changes.
- A basis for identifying and separating the fundamental relationship between the STRATEGIC PLANNING, MANAGEMENT and OPERATION

planning and control processes.

- A basis for defining the required information data bases architecture.

Ideally, a manager's job is to manage the resources within his realm of responsibility. Execution of this responsiblity entails decision-making activities encompassing a four-stage decision life cycle as follows:

- Requirements - Decisions that involve a determination of how much, a plan for getting it and a measurement that illustrates performance of control against that plan for those resources.
- Acquisition - Encompassing decisions that develop a means for getting the resources required to produce.
- Stewardship - Involving decisions that form, refine, and modify the resources.
- Retirement - Requiring decisions that terminate the resources.

Business processes encompassing all resources while subject to the life cycle of a decision serves as the vehicle for the structured, logical and comprehensive format for MDSS.

The next logical step in the application of business processes for MDSS design and development is defining and grouping these processes into STRATEGIC planning and MANAGEMENT control. Strategic planning is often times referred to as the long range plan, or development plan, covering 5-7 years, while the management plan is referred to as the product plan, the operating plan, or sometimes the contract plan. Generic MRP II defines them as the management and production plans respectively.

Depending on the individual organization and associated terminology, business processes in most companies should not be too difficult to identify. Be that as it may, their identification is paramount to a logical and orderly structure for the Management Decision Support System. Identification and assignment of the processes associated with strategic planning and management control, first, will provide a cleaner distinction between management control and operation control thereby reducing the redundancy usually present in the management control processes.

Examples of strategic planning and management planning and control processes are shown in Exhibit 2 & 3. More specifically, the business processes along with their attendant functions that were adopted at the Aerospace Fastener Division, Huck Manufacturing Co., Carson, CA, in November of 1982 are illustrated in Exhibit 4. It should be noted that these business processes are management planning oriented as opposed to strategic planning and control. This is acceptable because the Aerospace Fastener Division is one of three domestic divisions within Huck Manufacturing Company and, therefore, represents an "element" or item in the company's Management Decision Support System.

Well-defined and structured business processes are important to the implementation strategy of the Management Decision Support System. Ideally, implementation is built upon corresponding modules or implementable pieces of the business system. As such, they represent a depository or management focus within the overall system. MDSS within the business process concepts should be considered flexible in nature; that is, certain steps and applications can be achieved in order to adapt certain variations without diluting its basic purpose. However, the basic concepts of MDSS must be considered inviolate. It is within itself a logical business process.

V. DEFINING DATA CLASSIFICATION

Just as business classes are defined as logically related decisions and activities, data classes can be defined as categories of logically related information.

There are specific types or categories of data associated with a typical business. These are:

- Planning Data - representing objectives or expectations supporting requirements activities.
- Inventory Data - maintaining source status and supporting maintenance activities.
- Transaction Data - affecting changes caused by the acquisition activity.
- Statistical/Summary Data - employing extraction techniques to provide the feedback essential for determining a measurement.

Most companies are concerned about and collect, store, retrieve, and display information about customers, products, material, and personnel. These items or entities include the resources around which the business processes are defined. An effective management decision support system is dependant upon the ability to establish a direct relationship or identity with the data and each individual business process using that data. There are basically three steps in accomplishing this.

First, a data class/item or business entity matrix as shown in Exhibit 5 can be developed. Secondly, a cross-reference or regrouping on the basis of commonality is developed in order to compile a manageable list of data classes. The application of the cross-reference, as illustrated in Exhibit 6 can yield as many as 60 different data classes for a typical manufacturing company.

Finally, consolidation, encompassing the three essential elements of a management decision support system, the decision life cycle, the business process, and the data class is the final and essential consideration prior to establishing the data bases structure of the MDSS. This can be readily accomplished as a result of incorporating the decision life cycle into the cross-reference or regrouping activity performed while defining data classes. In doing so, a matrix, as illustrated in Exhibit 7, can be developed to aid in visualization of the existing interrelationships.

Although the business process/function may have one or more decision life cycles, the primary or predominant cycle is selected in structuring the data bases. As such, the focus becomes "exception management" based as opposed to coping with details associated with each major decision.

A configuration hierarchy encompassing the three MDSS ingredients and relationships illustrated in Exhibit 8 envisions the data bases as triangular in form, with BUSINESS PROCESSES at the top, and DECISION CYCLES and DATA CLASSES as interchangeable at the bottom of the triangle.

VI. MANAGING MDSS RESOURCES

MDSS Concepts:

The generic framework of MDSS can be best characterized as having three components as follows:

- A language - permitting the user to state problems for the MDSS to solve.
- A knowledge - presenting pertinent facts about an application in order to solve a problem.
- A processor - combining the stated problem (language) with the facts (knowledge) and generating information for decision support.

The heart of MDSS is the processor. Broadly speaking the MDSS accepts problems represented to it through the language and utilizes application of specific information within its knowledge. The processor as the dynamic (executable) component combines the language and knowledge thereby providing information in specified formats that aid in the decision-making process.

These three components can be organized in many ways depending upon the specific application of the MDSS. The knowledge component is expressed as a file or data bases containing the knowledge for decision support. The processor component is the software that determines the activities and behavior of the MDSS. The language component can vary, but must be compatible to both the knowledge and the processor. In any case, this combination of all three must have the explicit ability to state the problem and translate it into an appropriate plan of action. Exhibit 9 illustrates the identification and relationships of the application systems modules from ASK Computer Systems, Inc., and the MDSS component concepts existing at Huck Manufacturing Company's Group Headquarters in Irvine, California. These systems modules support executive management in measuring the business performance of three domestic divisions and one international division.

MDSS concepts have been presented in a generic framework incorporating an actual infra-structure, thereby demonstrating the nature of its components. It then becomes possible within this broad outline to develop a specific management decision support system that conforms to the basic requirements of its components and user community.

<u>MDSS Overview</u>:

Management Decision Support Systems, having its basic structure as a stand-alone, integrated module, is concerned with implementation methods that can economically achieve a direct link between a mainframe or mini computer, and micro computers or other devices such as intelligent terminals. MDSS, which is oriented toward executive management, addresses the inquiry, retrieval, and modeling of summary exception-based information associated with established business processes. Additionally, the following three functions are provided:

- Reporting - encompassing financial status, sales and operational results.
- Performance measurement - assessing the effectiveness of operational management, planning, controls and data base integrity.
- Diagnostics - researching, through modeling and predicting problem conditions and environments.

"Decisionmaker" from ASK Computer Systems, Inc., Los Altos, California establishes a specific MDSS framework involving reporting, performance measurement, and diagnostics as a result of the "MANAGEMENT REPORTING DATA BASE." This data base is the depository for summary data from all core modules (MANMAN, OMAR, A/P, and G/L) with appropriate exits to permit integration at a later date with the total package's remaining modules; namely, FIXED ASSETS, PAYROLL and SERVICEMAN. The summarization necessary to build the MANAGEMENT REPORTING DATA BASE is accomplished through the use of utility programs which are run against the specific module's data bases comprising the application software.

"Decisionmaker" permits the user to focus on discrete aspects of the business by establishing "key indicators" or business "pulse points." From these the user can define the exception levels on individual files within the data base. This exception level then serves as the starting point for investigating the "why" questions.

"Decisionmaker" with its ability to store application system unrelated data such as value of fixed assets, level of debt, interest rate, etc., while being able to copy historical data to files to aid in implementation, is a cautious but bold venture into the MDSS field. Moreover, its initial release scheduled for 1985 establishes the infra-structure for enhancements and further development within the ASK user community.

The balance of "MDSS Overview" will examine a "generic" management decision support system in general terms while using the foundation established by "Decisionmaker," along with its attendant features, Exhibit 10.

Fundamentally, a management decision support system should achieve an ideal blend of existing manual management planning and control practices with selected new automated features so as to provide an on-line-interactive decision support capability. The system allows controlled access to its data bases at assigned levels and the inclusion of specific user requirements so that a "TOP DOWN" philosophy can be employed with regard to the application of the corrective action function.

Operation of MDSS begins with establishing a (physical and conversational) link between an intelligent terminal or micro computer, and the CPU of a mini or main frame computer. Upon confirming acquisition of this link the MDSS continues with a series of three prompts and associated passwords.

The first of these prompts indicates the user has established a working relationship with the management decision support system. Entry of a password at this time offers a choice of four discrete functions (strategic planning, management control, operational control or resource management).

The second prompt defines the planning and control levels or categories as STRATEGIC PLANNING or MANAGEMENT PLANNING and OPERATIONAL CONTROL along with their associated business processes. Entry of a password here gives access to the desired level and the user is prompted to select a business process along with its associated functions. A complete response to all active prompts and entry of all authorized passwords provides access to RELATIONAL structured files or data bases. Having gained access, the user is able to format, manipulate, and consolidate the selected data independent of the MDSS data base structure. Upon completion of the function, the user can return the data to the confines of the data base.

MDSS is sensitive toward executive management's need for specialized friendliness for a data processing system. It employs the "conversational mode" supported with phrases and terminology in "layman English" as well as numerous detail prompts and menus. Additionally, specifications, detail operating instructions, and diagnostical statements, along with appropriate corrective actions are provided as part of the "help" section of the system.

This overview very briefly discussed a specific MDSS structure, "Decisionmaker" while envisioning it in a generic environment. It is hoped that the reader can identify with its application somewhere between these extremes while keeping in mind that while MDSS may still be in its embryonic state, it nonetheless qualifies as a viable developmental product. As such it represents another important chapter in application systems software development.

Data Bases:

This topic examines a philosophical point of view relative to data base construction and processing. It attempts to visualize a conceptual design for MDSS while giving consideration to the growth, in recent years, relative to on-line transaction processing systems involving large volumes of daily updates and terminal handling facilities while keeping in mind the importance for the need for specific or customized user friendliness when operating the system.

Given the fact that MDSS is a tool for executive management, its driving force, the data base, then can be constructed on a foundation employing relational or near-relational data base technology encompassing five critical characteristics as follows:

- Modular Growth - precluding the need for expansion or upgrading of hardware in order to achieve higher performance.
- Continuous Availability - providing use during regular working hours while compensating for erratic use by the senior executive at odd hours from unrelated and diverse locations.
- Data Integrity - incorporating a "soft failure" without corrupting other components of the overall system.
- Ease of Use - achieving a non-threatening, simple, and direct interface with the processing functions of the system.
- Performance - dedicated toward increasing productivity as a result of one or more users of the system at the same time.

MDSS data base design requirements have other implications. These are:

- Quality - based on the durability of the data base as a result of applications system software/hardware configuration as the "total system."
- Data Representation - encompassing the flat two-dimensional

tabular file incorporating the trait that established them as logically interdependent but physically independent.

The MDSS data base is structured with elements essential to determining status, permitting manipulation, allowing accumulation, providing visual (screen and hard copy) and graphic displays, and storing of historical data. Further the type (internal, inter, intra, and external) must be accommodated. An example of this, illustrating management report data elements from Huck Manufacturing Company, Irvine, California is provided in Exhibit 11.

Perhaps, the ideal processing environment for MDSS is Distributed Data Processing (DDP) because of its ability to incorporate all types of computers. Besides linking of computers via telecommunications - Global, Continental, or local area networks, DDP will support remote terminals and micros while permitting a download from the host computer to perform other tasks. As the optimal environment for MDSS, Distributed Data Processing can provide processing capabilities that permit:

- Data Entry
- Data Inquiry
- Data Transfer - (Mini to Micro and Micro to Mini)
- Data Modeling
- Data Graphics
- Data regression Analysis
- Customized Data Storage

Development of a Distributed Data Processing capability emboding a relational data base in the DDP environment is encouraged because data distribution can be focused at its point of need or highest utilization. In doing so, the advantages to be gained from this are: improved reliability, continuous availability, elimination of redundancy, and achievement of simplicity. Nonetheless, the spreading of the required computing power consumed by MDSS, outward to the user, has a positive impact on functionality while reducing risk and complexity. Management Decision Support Systems operating in a DDP environment suggests maximum utilization of the feature for its intended purpose.

VII. BIOGRAPHY

Wes Christman is Group Manager, MIS, at Huck Manufacturing Company, Irvine, California. He joined Huck in their Aerospace Fastener Division, Carson, California, as a Business Systems Analyst in August, 1981. He served as Project Leader for the Aerospace Fastener Division and, since his promotion in February, 1983, has directed the decentralization effort for the company's information system resources. This effort involved the implementation of ASK Computer System's MRP II Software Package consisting of MANMAN, OMAR, A/P and G/L on the Hewlett-Packard 3000 Series Computer. A simultaneous implementation effort encompassed three domestic divisions; Aerospace Fasteners, Carson, California; Industrial Fasteners, Waco, Texas; Installation Equipment, Kingston, New York; one international division, Huck Canada Limited, Toronto, Ontario, Canada, and the company's headquarters in Irvine, California. Huck Manufacturing is one of five operation groups within Federal-Mogul of Southfield, Michigan.

Mr. Christman's prior experience encompasses project management and manufacturing systems analysis with IMB's "MAPICS" System 34 & 38 and "COPICS" on an IBM 4341 computer. Mr. Christman holds a BS Degree in Business Administration from the University of Iowa, Iowa City, Iowa, with postgraduate studies toward a MBA with the University of Santa Clara, Santa Clara, California.

Mr. Christman, since authoring this paper, has resigned, effective January 4, 1985, the Group Manager MIS position with Huck Manufacturing Company. He has accepted the position of Director, MIS at TRW-LSI, La Jolla, California. His appearance at the conference is under the auspices of his new employer.

VIII. EXHIBITS

EXHIBIT	#1	PLANNING AND CONTROL LEVEL
EXHIBIT	#2	STRATEGIC PLANNING AND MANAGEMENT CONTROL PROCESSES
EXHIBIT	#3	STRATEGIC PLANNING AND MANAGEMENT CONTROL PROCESSES
EXHIBIT	#4	BUSINESS PROCESSES AND FUNCTIONS FOR THE AEROSPACE FASTENER DIVISION, HUCK MANUFACTURING CO.
EXHIBIT	#5	DATA CLASS/ITEM OR BUSINESS ENTITY MATRIX
EXHIBIT	#6	DATA CLASS/ITEM AND BUSINESS ENTITY MATRIX
EXHIBIT	#7	DATA CLASS/ITEM AND BUSINESS CONSOLIDATION MATRIX
EXHIBIT	#8	MDSS INGREDIENTS RELATIONSHIP
EXHIBIT	#9	APPLICATIONS SYSTEM MODULE MDSS COMPONENTS RELATIONSHIP
EXHIBIT	#10	ASK COMPUTER SYSTEMS, INC.
EXHIBIT	#11	TYPICAL MANAGEMENT REPORT DATA ELEMENTS, HUCK MANUFACTURING CO.

Exhibit #1 Planning and Control Level

Decision Characteristic	Planning and Control Level		
	Strategic Planning	Management Control	Operational Control
Management involvement	General management Functional management	General management Functional management Operational management	Functional management Operational management
Time horizon	Long range (1-10+ years)	Year-to-year Monthly	Day-to-day Weekly
Degree of structure	Unstructured and irregular; each problem different	More structured, cyclic, largely repeating	Highly structured, repetitious
Data requirements	Summaries, estimates, difficult to pre-define much external to business	Summaries, definable, need for unanticipated forms, largely internal	Detail, operational, definable, internally generated
Resource management	Establishment of policies pertaining to the resource	Allocation of the resource	Efficient use of the resource

Exhibit #2 Strategic Planning and Management Control Processes

Strategic Planning— encompassing activities associated with;
Five-Year Business Planning—Establishment of a long-range business plan, taking into consideration such things as product, product demand, capacity, manpower, financial strategies, and information systems to achieve company goals.
Annual Operating Planning—Development of a medium-range operating directive to control the business activities in achieving the five-year business plan.
Quarterly Operation Planning—Quarterly restatement of annual operating plan, reflecting a "firm" operating plan for three months and a "forecasted" operating plan for the following three months.
Reporting and Control—Review of the performance of operating units to ensure the success of the plans. Change directions of operating units and/or plans when necessary to meet the objectives of the company.
Capacity Planning—The determination of production and test capacity required to meet the planned production volume, allocation of this within the company and supplier facilities, and acquisition of any additional capacity.
Determination of Optimum Capacity—Required productive and test capacity is determined on the basis of the planned production volume.
Capacity Allocation—The assignment of productive and test capacity among the company and supplier plants. Make-vs.-buy analyses are made to determine internal and external (supplier) requirements.
Market Research, Market Development—Identify markets, lay plans, and set strategies for penetrating those markets.
Pricing—Establish as part of market penetration and product planning strategies.
Publications, Advertising, Promotion—Support and augment strategies and tactics of market development.
Sales Analysis and Forecasting—Provide necessary information to plan and measure strategies and tactics.
Warranty, Policy and Other "Software"—Establish as integral with strategies and tactics of market penetration.

Exhibit #3 Strategic Planning and Management Control Processes

Management Planning and Control—
including activities dedicated toward;

Plant Operations—The process of producing products or parts according to a given schedule and specifications in the most efficient manner.

Planning, Scheduling and Control—Determination of quantities to be produced on a daily basis and the corrective action taken if necessary.

- Performance Reporting—Communicating the quantities produced in a given time.
- Expediting—Short-term replanning as required.
- Planning of material, manpower, and optimum use of equipment.
- Order Processing—The process of maintaining a plant order board of all products to be shipped, including finished engines, components, kits and service parts, both to customers and other plants.
- Product and Process Specification—Maintenance of records regarding product definition and the incorporation of engineering changes.

Performance Reporting—The process of feeding back information on actual production against the master production plan on a periodic basis, and provision of input for any required modification of plans and schedules.

Inventory Control—Determination and control of the levels of inventory required to meet a predetermined level of production and service, with the objective of minimizing the overall cost.

Financial Control—Establish, coordinate, and administer, as an integral part of management, an adequate plan for the control of operations.

- Budget/Expense—Provide expense budgets and cost standards together with the necessary procedures to effectuate the plan.
- Managerial Accounting—Compare the results of operations with plans and standards; interpret them for all management levels.

Operations Control—Development of a sourcing plan with and between the plants, the review of delivery performance, and corrective action necessary to accomplish the business commitments.

- Master Production Planning—Translation of a central order board into a coordination production plan and placing requirements in plants at a meaningful part level.
- Transportation Planning—Determination of transportation strategies and capabilities necessary to support the operations plan.

Personal Planning

- Policies—The establishment and implementation of policies and practices concerning employment, development, compensation, and evaluation of employees.
- Requirements—Determination of manpower requirements in line with business objectives.

Exhibit #4 Business Processes and Functions for the Aerospace Fastener Division, Huck Manufacturing Co.

Strategic Planning and Control	Management Planning and Control
Marketing • Planning • Research • Forecasting • Pricing Sales Operations • Quotations • Order History • Order Management and Servicing Product Engineering • Design and Development • Product Specification Maintenance • Product Data and Information Management Management • Business Planning • Organization Analysis • Review and Control • Risk Management	Manufacturing/Industrial Engineering • Item Master • Methods & Routings • I.E. Standards • Tool Design • Work Flow Layout • Machine Tool Selection • Raw Material Evaluation & Selection Production • Manning • Facilities Maintenance • Shop Floor Operations/Data Collection Quality Assurance & Reliability • Inspection • Product Quality • Vendor Approval • Statistical Analysis Accounting & Finance • Cost Accounting • General Accounting • Accounts Payable • Capital Acquisition • Cash Management • Payroll • Accounts Receivable Human Resources • Manpower Planning • Skills Inventory • Recruiting • Compensation • Regulatory Agency Reporting Material Management • Shipments • Scheduling/Dispatching • Purchasing • Shipping/Receiving • Material Planning • Inventory Control

Exhibit #5 Data Class/Item or Business Entity Matrix

DATA CLASS TYPES	BUSINESS ENTITIES OR ITEMS						
	Product	Customer	Facilities	Material	Vendor	Cash	Personnel
Plans/ Models	Product plans	Sales territory Market plans	Facility plans Capacity plans	Material requirements Production schedule		Budget	Personnel plans
Statistical/ Summary	Product demand	Sales history	Work in process Equipment utilization	Open requirements	Vendor performance	Financial statistics	Productivity Benefit history
Inventory	Product Finished goods Parts master	Customer	Facilities Machine load Routings	Raw Material Cost Bills of material	Vendor	Financial General ledger accounting	Employee Payroll Skills
Transaction	Order	Shipment		Purchase order	Material receipt	Recipts Payments	

Exhibit #6 Data Class/Item and Business Entity Matrix.

Customer Product Sales Territory	Marketing Planning	Sales Analysis
Financial Customer Product	Finance Analysis	Profitability Analysis
Raw material Inventory Vendor Order	Materials Management Purchasing	Purchase Order
Product Customer Customer Request	Sales Operations Order Servicing	Order
Product Open Requirements Bills of Material Facilities	Production Scheduling	Production Schedule
Employee	Human Resources Personnel Planning	Government Reports

Exhibit #7 Data Class/Item and Business Consolidation Matrix.

Customer Product Sales Territory	Marketing Requirements Planning	Sales Analysis
Financial Customer Product	Finance Stewardship Analysis	Profitability Analysis
Raw material Inventory Vendor Order	Materials Management Acquisition Purchasing	Purchase Order
Product Customer Customer Request	Sales Operations Stewardship Order Servicing	Order
Product Open Requirements Bills of Material Facilities	Production Requirement Scheduling	Production Schedule
Employee	Human Resources Acquisition Personnel Planning	Government Reports

Exhibit #8 MDSS Ingredients Relationship

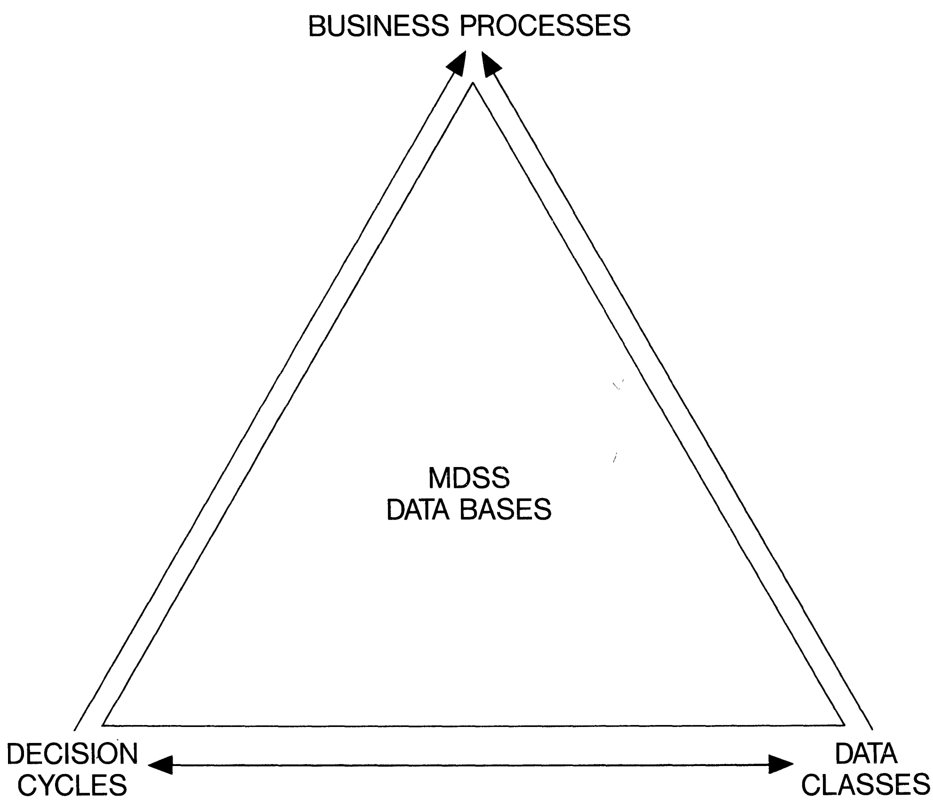


Exhibit #9 Applications System Module MDSS Components Relationship

APPLICATIONS SYSTEM MODULE	MDSS COMPONENT
DATA BASES MANMAN OMAR G/L A/P PAYROLL	KNOWLEDGE
HP FUNCTION QUIZ QUERY COBOL	LANGUAGE
HP IMAGE EXPERT POWER PLAN MP-VE	PROCESSOR

Exhibit #10 ASK Computer Systems Inc.

"DECISION MAKER" PRODUCT FEATURES
Detail data from the application databases is summarized into one MANAGEMENT REPORTING database.
Utilities initiated by the user perform the data summarization.
Summarization utilities have an option to print the detail information or summary totals.
Display of summarized information is on-line using LIST type inquiry screens. This information can optionally be printed as well.
Users can select specific Key Indicators or critical information from the summary database to trigger Exception Reporting.
When a user logs into the MGMT REP system, automatic notification of exceptions will occur.
User defined data can also be stored in the summary database (such as: Asset information, Debt level and interest rates, EPS, Number of Shares Outstanding, Stock Trading Price, Personnel levels (direct and indirect employees).
Information stored in the summary database is at an aggregate level; however, there are optional levels of summarization (such as: Inventory Location and/or Inventory Account, WIP Account, Inventory vs. Expense (and Miscellaneous) P.O.s, Bookings and Shipments by Product Type / Customer Type / Sales Region, etc.).
Historical data can be stored in monthly buckets for the current year and last year, and the totals for the four years previous can also be stored.
Optional automatic posting of past historical information.
If information is available (but not on-line) for the past five years, data can be manually input into the MGMT REP system (facilitating trend analysis).
Before all ASK packages are implemented data can be manually entered into the MGMT REP database, so the system can be fully utilized immediately upon installation.
Month and year-end close utilities (similar to the ASK General Ledger System) automatically move current month/year data into last month/year data cells.
Actual summary information can be compared to a monthly, quarterly, or yearly budget previously entered.
On-line help available at the program prompts.
Default values and field lengths are displayed in input routines.
Menu or command driven from one central dispatcher.
The ASK Password database system provides security at command and transaction level.

Exhibit #11 Typical Management Report Data Elements, Huck Manufacturing Co.

Status Data Elements
- Actual
- Forecast
- Variance—Favorable/Unfavorable
- Adjustments
- Projected

Accumulated Data Elements
- Prior Month
- Current Month
- Next Four Months
- Next Four Quarters
- Prior Year
- Current Year
- Year-to-Date
- Trade
- Inter
- Intra
- Total
- Capitalized
- On Order
- Not on Order

Manipulated Data Elements
- Incoming Orders
- Backlog
- Sales and Shipments
- Contribution Margin
- Operating Profit
- Investments
- Cash Flow
- Profit and Loss
- Cost of Sales
- Expenses
- Balance Sheet
- Product Order and Sales

Historial Data Elements
- Product Codes
- Customer Location
- Geographical Data
- Quantity
- Value
- End User Industry
- Part No.
- SBA Code
- Channel of Entry
- Dollar Value
- Profitability

External Data Elements
- Airline Profitability
- Commercial Air Transport Orders & Shipments
- Import-Export Trade
- Economic Indicators
- Industry-Association Dues
- Access/Integration to Other Existing Data Bases
- Competitors/ICK Data

Integrating Series 100 Personal Computers with the HP 3000

Bill Crow

Learning Products Center
Personal Software Division
Hewlett-Packard Company
3410 Central Expressway
Santa Clara, CA 95051

Introduction

The HP Touchscreen Personal Computer (HP 150) and The Portable (HP 110) are powerful personal computers, capable of handling a variety of tasks. But, by combining the local processing resources with the storage and data handling capabilities of the HP 3000, a whole new range of capabilities is possible.

Many tools are available today for integrating applications on The Touchscreen PC, or on The Portable, with applications on the HP 3000. This paper will describe several available techniques, and introduce some public domain utility programs that can aid the process. The public domain programs described here have been submitted to the Amsterdam Conference Swap Tape, and are also available online in North America through the HP Series 100 Forum on the CompuServe computer network.

Adding the Extra Touch

One of the easiest ways to take advantage of the unique features of the Touchscreen Personal Computer is to modify your HP 3000 applications to make use of the touch sensitive display. All the features of HPTouch can be accessed by sending escape sequences to the Touchscreen PC while in terminal mode. The Touchscreen returns touch reports by transmitting either escape sequences or predefined character strings (in a manner very similar to using the softkeys.)

There are two basic touchscreen reporting modes and various operating modes within each. The touchscreen can either report a specific row/column address when it is touched or return a specified string when a predefined region is touched. If desired, both modes can be active simultaneously.

Using the touchscreen to position the cursor in MemoMaker is a good example of row/column reporting. MemoMaker must be able to detect any possible touch location, from a wide range of possibilities, and report the specific position. On the other hand, touch field reporting is used by P.A.M. to select the application to run, since there are a finite set of touch regions required, each with a defined purpose. The touchable function keys at the base of the display are controlled by their own escape sequences, compatible with many other HP block mode terminals.

All touchscreen escape sequences begin with <Esc>-z (Escape, hyphen, lower case z). The first sequence sets the desired reporting mode.

<Esc>-z<rmode>n<tmode>M

Values for <rmode>:
- 0 - Turn off touchscreen reporting
- 1 - Only report row/col touches
- 2 - Only report defined field touches
- 3 - Report both row/col and field touches

The row/column reporting mode has three different operating modes, to detect touch, release, or both, determined by the value of <tmode> as described below:

Values for <tmode>:
- 1 - Report when the screen is touched
- 2 - Report when the touch is removed
- 3 - Report on both touch and release

When operating in row/column mode, the display reports the requested event by returning its own escape sequence, which can be read by a program as if it were typed at the keyboard. (Like all information returned by the teminal, the escape sequence is not transmitted until a DC1 prompt is received.) The returned escape sequence looks like this:

<Esc>-z<col>x<row>y<type>Q<cr>

<col> and <row> are two-digit numbers corresponding to the screen-relative position that was touched (or where the touch was released). <col> will be in the range 00 to 79, and <row> will be between 00 and 23. In all cases, coordinate (00,00) is the upper left corner of the display.

<type> indicates what type of event occurred; it is equal to 03 for a touch report, and 04 for a release report. <cr> is the carriage return character (decimal ASCII value = 13).

Since <tmode> is only required when row/column reporting is activated (<rmode> equals 1 or 3), the argument can be omitted when only touch field reporting is required. Therefore, the following escape sequences are legal abreviations:

<Esc>-z0N {turns off all touch reporting}

<Esc>-z2N {turns on touch field reporting only}

Row/column reporting is useful for many applications where specific cursor addressing or cursor positioning is required, but it is a cumbersome tool for menu selection tasks. However, the touch field reporting feature simplifies this type of operation. With one escape sequence per touch field, the power of the resident firmware can transform a command driven program into a touch-controllable menu driven program. By using predefined touch fields, touch control can be easily added to almost any HP 3000 program. The following escape sequence is used to define each touch field, and all the parameters that control its operation:

<Esc>-zg<srow>,<erow>r<scol>,<ecol>c<cpos>p<beep>b
<eoff>e<eon>f<type>a<tmode>m<len>L<string>

While this may look complicated, many of the arguments are optional, so the sequence is often much shorter in actual usage. Following are the definitions of the various parameters:

<srow>, <erow>r — <srow> specifies the starting (uppermost) row of the field, and <erow> defines the ending row. <erow> must be greater than <srow>, and both values must be in the range 00 to 23 inclusive. If omitted this parameter will default <srow> to the current cursor position and <erow> to <srow>+2.

<scol>, <ecol>c — This parameter is similar to the preceding one, but it defines the field's columns. <ecol> must be greater than <scol>, and both values must be in the range 00 to 79 inclusive. The default for <scol> is the current cursor position, and <ecol> defaults to <scol>+7.

<cpos>p — An optional parameter that will cause the cursor to be moved to the field when it is touched. The default is 0, which creates no effect; a value of 1 will cause the cursor to move.

<beep>b — If 1, a beep will sound when the field is touched. If the argument is 0 or omitted altogether, no beep will sound.

<eoff>e <eon>f — These values specify the display enhancement for the field for its "off" and "on" states. They may be any value from 0 to 15, corresponding to the standard display enhancement specifications as shown in Table 1.

<type>a — This parameter specifies the field type as follows:

1 - ASCII field
2 - not used
3 - Toggle field
4 - Normal field

These are described in more detail below.

<tmode>m This is the same as the <tmode> parameter used in row/column reporting, specifically:

1 - Report when the display is touched
2 - Report when the touch is released
3 - Report on both touch and release

<len>L
<string> This specifies the length of <string> and the text string to be returned when the specified event occurs. Its interpretation depends on the value of <type>.

There are three different types of touch fields, as specified by the <type>a parameter described above. They provide distinctly different operating modes, to accommodate a variety of application requirements.

ASCII field - <type>=1 - When the specified event occurs (as determined by the value of <mode>, the character string specified by <len>L<string> is returned. The field is in the "on" state whenever it is being touched and is in the "off" state at all other times.

Toggle field - <type>=3 - Each time the field is touched, it changes state between "on" and "off". It returns the following escape sequence:

<Esc>-z<buf><state>Q

<buf> is the first two characters of <string> (so <len> should be 2, and only two characters should be specified for <buf>), and <state> is equal to 1 if the field is turning on or 2 if the field is turning off.

Normal field - <type>=4 - This type of field is similar to a toggle field, except the field is only in the "on" state when it is being touched. The escape sequence that it returns is as follows:

<Esc>-z<buf><action>Q

<buf> is the same as above, and <action> is 5 when reporting the field has been touched or 6 when reporting that the touch has been released.

Finally, we need an escape sequence to eliminate touch fields once they have been created. Once defined, touch fields remain in the same position relative to the 48-line alpha display memory, regardless if the alphanumeric or graphic information on the display is scrolled, cleared, or changed. Since they are positioned relative to the 48-line display memory, they can be moved by scrolling the screen, and even scrolled off the active display area. However, they are still defined, and if scrolled back to the visible 24-line window, they will still be active. Therefore, it's important for a program to be aware of the state of the display when touch fields are active, and clear them with the following sequences when touch is not being used. The first escape sequence is used to remove a single touch field:

<Esc>-zd<row>r<col>C

where <row> and <col> specify the field's beginning location. All touch fields can be deleted at once with this sequence:

<Esc>-zD

Because of the powerful escape sequences provided in the Touchscreen Personal Computer's firmware, adding touch to your own programs is a simple task. Of course, it still requires careful design on your part to put them all together in an easy-to-use application that makes the most of the unique capabilities of HPTouch.

Exchanging Information Between PC's and the HP 3000

For many applications, the best set of tools is a combination of HP 3000 and personal computer resources. The HP 3000 provides the large data storage resources, multi-user access, and processing power for corporate applications, while the personal computer provides the flexibility, response time, and interactive capabilities for local processing and individual data analysis.

The following example demonstrates a technique to extract information from an IMAGE/3000 database, download the information to the personal computer, and process it locally using 1-2-3 by Lotus. The techniques presented focus on automating the entire process, freeing the user from the details of performing a series if individual steps.

The IMAGE/3000 database is a simple system, simulating a portion of an inventory control and order processing system. It contains a manual master dataset (ITEM-MASTER) with one record describing each item in the inventory, and a detail dataset (DAILY-SALES) that tracks one day's sales. Figure 1 shows the schema. A complete system would contain additional information, but this will be adequate for our example.

In this example, the entire ITEM-MASTER dataset is extracted and downloaded to create a local inventory database. A summary of the DAILY-SALES database is also downloaded. Using 1-2-3 by Lotus, the daily sales prices are compared with the list and cost prices in the inventory database to compute daily profit margins and discount levels. Also, the inventory levels are compared against a locally maintained order point list, and a recommended restocking purchase is computed. While these tasks could probably be accomplished with less complexity on the HP 3000, our purpose is to demonstrate the integration capabilities, not produce a complete application.

The Touchscreen PC to HP 3000 Link

Fundamental to any integration of applications between the Touchscreen PC and the HP 3000 is the ability to transfer information. AdvanceLink, a sophisticated communications program from Hewlett-Packard provides a full set of capabilities to meet most communications requirements. Text or binary files can be transferred between computers with full error correction, and command files can be created to automate virtually any function. While most users only exercise the terminal emulation, file transfer, and data logging capabilities, the ability to create complex command files is clearly AdvanceLink's most powerful feature.

AdvanceLink command files can be called directly from the P.A.M. menu, or an MS-DOS Batch file. This allows the programmer to create an application that completely hides the complexities of data communications from the end user.

In this example, AdvanceLink controls all access to the HP 3000, including running QUERY and transferring the files. The command file is presented in figure 2. AMSTERDM.ALK first sets up the logon and logoff sequences, and then logs on with the &REMOTE TOGGLE command. Next it purges any leftover files if there was an aborted run, builds a new file for the REPORT, and sets the file equation. QUERY is called to create an SD file with the ITEM-MASTER data, and prepare a REPORT summarizing the DAILY-SALES data by part number. The &DSCOPY command is used to download the two files. Note that AdvanceLink automatically converts the SD file (ITEMS) created with the SAVE command, to a DIF file (ITEMS.DIF) during the download. This standardized data file format can be interpreted by many applications running on the TouchScreen PC, including 1-2-3 by Lotus. The REPORT output file (SALESUM) is downloaded as a simple ASCII file. Finally, the intermediate files are purged and the &REMOTE TOGGLE command is used to log off.

Making The Portable Connection

While AdvanceLink provides an excellent solution for the Touchscreen PC, The Portable's ROM-based Terminal program does not offer the same level of capability. However, proper use of the autologon feature, download configuration menu, and HP 3000 User Defined Commands (UDC's) can provide most of the necessary capability to create an automated application.

Terminal does not provide a command file capability, so manual intervention is required to initiate the file downloads, and terminate the online session. The user must be able to access the Download Config menu, load a configuration file, and then start the download. If there was only one file to download, most of this process can be automated.

Terminal provides file upload and download capability, but it only supports the XMODEM protocol, and does not support the Link protocol used by AdvanceLink. Once again, a public domain program provides the solution; XMODEM/3000 allows the HP 3000 to send and receive text or binary files using this popular protocol. Written by Cliff Looyenga, and HP SE in Spokane, Washington, USA, XMODEM/3000 interfaces directly with The Portable's upload and download configuration menus by using a simple User Defined Command (UDC).

The automatic translation from SD to DIF file cannot be performed by Terminal, but the translation utility can be called explicitly to perform the necessary conversion before download.

The Hidden Power of 1-2-3.

1-2-3 by Lotus is a powerful integrated application, built in to the read-only memory (ROM) of The Portable, and available as a disc-based application for the Touchscreen PC. While many users regularly excersize 1-2-3's worksheet, graphics, and database capabilities, few have uncovered it's most powerful feature: keystroke macros. Complete applications can be developed using keystroke macros, including menus, input prompts, and help screens; these help insulate the end user from the many complexities of 1-2-3.

Before we can use the DIF file containing the inventory data (ITEMS.DIF), it must be converted to a 1-2-3 worksheet file (ITEMS.WKS). This could be accomplished using the Convert program on the 1-2-3 utilities disc, but the procedure involves several manual steps. However, the appropriate conversion program (DIFWKS.EXE) can be called directly from an MS-DOS command, passing the necessary arguments for the conversion, and avoiding the need for intervention. (See the section on Batch files below for the details.) This data is loaded in the worksheet as shown in figure 3.

In our example, I have deliberately avoided many of the advanced features of keyboard macros in the interest of simplicity. The macro in figure 4 is named \0, causing it to execute automatically when the worksheet is loaded. This macro first clears old data from the inventory and sales summary areas and loads the two files that were downloaded.

The 1-2-3 IMPORT command (/FI) is used by the macro to read the sales summary file (SALESUM.PRN) directly. Figure 5 shows this data loaded in the worksheet.

Once the files are loaded, the worksheet is recalculated, computing average profit margin and average discount for each item. It also prepares the recommended replacement orders, based on the local orderpoint list. A final summary is displayed when the calculation completes. Figure 6 shows the summary report produced, and figure 7 displays the calculations used. This information is derived from the inventory table, shown in figure 8. The calculations used in each column are shown in figure 9. The Order Point column looks up information in the local Order Point table, shown in figure 10.

It is important to note the use of range names in the calculations. Because the inventory data and sales summary data could have any number of records, it is difficult to predefine the ranges used in the calculation. However, by using a keystroke macro, the ranges that define the data, and the columns of equations can be redefined, allowing them to fluctuate based on the data. This is a fairly sophisticated use 1-2-3, creating a dynamic worksheet that grows with the data. It also adds a considerable amount of computing overhead. If the table sizes were fixed (or at least had a reasonable maximum assigned), the \0 keyboard macro would be dramatically simplified, and the performance could be improved. But since you can't always predict the nature of imported data, it is handy to be able to adapt to any requirement.

Pulling It All Together with Batch Files.

The Personal Application Manager (P.A.M.) provides a friendly, easy-to-use executive interface for the Touchscreen PC and The Portable, and protects the user from the more complex details of MS-DOS command mode. However, the powerful features of MS-DOS Batch files are left behind in the process. Batch file capability is still available, and can be easily accessed from MS-DOS COMMANDS. With the help of a handy little public domain utility program (CALL.COM), MS-DOS Batch files can also be called from P.A.M., providing the best of both worlds.

A Batch file is simply a set of MS-DOS commands, executed in sequence; they are similar to User Defined Commands (UDC's), available under MPE on the HP 3000. Like UDC's, parameters can be provided, substituting for variables within the Batch file. Various conditional testing and error checking features all also provided.

The example created here is designed for execution on a winchester disc. The winchester is configured as drive A:, and the necessary applications (AdvanceLink, 1-2-3, MS-DOS Commands, DIFWKS, CALL, and the Batch file) are installed in the root directory. The data and command files reside in a subdirectory (A:\AMSTERDM) on the winchester disc. The floppy disc (B:) is reserved to hold the 1-2-3 key disc.

The application could also be structured to run from a dual floppy disc system, by using A: for the applications and B: for the data and command files. PAUSE statements must be added to the batch files, with message lines to instruct the user to change application discs at the appropriate times.

By creating the appropriate MS-DOS Batch file, we can pull together a series of sequential tasks, creating an automated, integrated application. The Batch File (AMSTERDM.BAT) in Figure 11 does some housekeeping, then calls AdvanceLink (running the command file discussed earlier) to download information from an IMAGE/3000 database. Next, the downloaded .DIF file is converted to a .WKS file by calling DIFWKS.EXE directly.

The batch file sets up a predefined 123.CNF file (copying it from AMSTERDM.CNF), to insure that 1-2-3 looks for data in the appropriate subdirectory. (AMSTERDM.CNF was created by setting the default directory to A:\AMSTERDM in 1-2-3, updating the configuration, and exiting. Then 123.CNF was copied to AMSTERDM.CNF.) Before calling 1-2-3, the Batch file renames the worksheet template (AMSTERDM.WKS) to AUTO123.WKS. This special worksheet file name is recognized by 1-2-3 and ia automatically loaded when the program is started. Combined with the automatic keyboard macro (\0), 1-2-3 can begin a series of tasks with no intervention required. When the user exits from 1-2-3, some final housekeeping is attended to, then control is returned to P.A.M. via the batch file's EXIT command.

A strange quirk in the way P.A.M. works on the Touchscreen PC makes it impossible to call Batch files directly. A useful public domain utility, CALL.COM, solves the problem by calling any program (in this case, COMMAND.COM), and passing it a complete paramater string. When COMMAND.COM is called, it can be passed an initial command, by preceeding it on the command line with the /C switch. Figure 12 shows the Touchscreen Install file (AMSTERDM.IN$) that creates a P.A.M. application to call the Batch file (AMSTERDM.BAT). This P.A.M. application is installed in the root directory.

The full features of MS-DOS Commands, MS-DOS Batch files, and the procedures for creating P.A.M. Install files for your own applications are documented in the HP 150/MS-DOS User's Guide (p/n 45624A).

Calling Batch files from P.A.M. on The Portable is eaiser. Simply add the following two lines to the file PAM.MNU:

```
Amsterdm Demo
COMMAND /C AMSTERDM.BAT
```

A P.A.M. menu item called "Amsterdm Demo" will be displayed, and when chosen, it will call MS-DOS Commands, executing the Batch file.

Summary

This paper has reviewed several techniques available for creating applications that integrate tasks on the HP 3000 and HP Series 100 personal computers. The integration examples presented make use of the public domain programs CALL.COM and

XMODEM/3000, MS-DOS batch files, AdvanceLink command files and SD-to-DIF file conversion, 1-2-3's DIFWKS.EXE conversion program being called from an MS-DOS command line, 1-2-3 configuration files, automatic worksheet loading, keyboard macros, and dynamic worksheets. We have only touched the surface of each of these features. The objective was to make you aware of their existence; refer to the appropriate documentation for more detailed information on their use and operation.

There are many other capabilities built in to applications on the Touchscreen PC, The Portable, and the HP 3000 that facilitate automatic integration of applications. Using these features, programmers can easily create applications for the end user that take advantage of personal computer capabilities, and offload numerous tasks from the HP 3000. To uncover this wealth of opportunities, you must explore past the basic capabilities an application provides, and investigate the many advanced features that can be tapped to create sophisticated, integrated solutions.

--- ### ---

1-2-3 and Lotus are trademarks of Lotus Development Corporation. MS is a trademark of MicroSoft, Inc.

--- ### ---

Table 1. Touch Display Enhancement Codes

<parm> value :	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Half-bright :									X	X	X	X	X	X	X	X
Underline :					X	X	X	X					X	X	X	X
Inverse video:			X	X			X	X			X	X			X	X
Blinking :		X		X		X		X		X		X		X		X

Figure 1. AMSTER Data Base Schema

```
BEGIN DATA BASE AMSTER;

PASSWORDS:
     10 READER;
     20 WRITER;

ITEMS:
     PART-NUM,           I2;
     DESCRIPTION,        X22;
     RETAIL-PRICE,       R2;
     ITEM-COST,          R2;
     INVENTORY,          I2;
     SELL-PRICE,         R2;

SETS:
     NAME: ITEM-MASTER, MANUAL (10/20);
       ENTRY:
         PART-NUM (1),
         DESCRIPTION,
         RETAIL-PRICE,
         ITEM-COST,
         INVENTORY;
       CAPACITY: 101;

     NAME: DAILY-SALES, DETAIL (10/20);
       ENTRY:
         PART-NUM (ITEM-MASTER),
         SELL-PRICE;
       CAPACITY: 1000;

END.
```

Figure 2. AMSTERDM.ALK AdvanceLink Command File

```
&! AMSTERDM.ALK - AdvanceLink Command File to download IMAGE/300 data
&!               From the HP 3000.  Part of a complete demonstration.
&!
&! 12/84  Bill Crow    (c) Copyright 1984  Hewlett-Packard Co.
&!
&!  (May be used or modified for any noncommercial purpose)
&!
&HOSTACCESS
&LABEL STARTLOGON
&! Set DC1 terminator off
&TERMINATOR OFF
&! Set timeout for response to carriage return
&WTIMEOUT 1

&WAITDC ^Q
&! Check if timeout expired
&IF &TIMEDOUT,=,TRUE
&! Try again
&GOTO STARTLOGON
&ENDIF
&WTIMEOUT 60
HELLO Demo.BC,Amsterdm
&! Set terminator to DC1
&TERMINATOR ^Q
&! Wait for host response
&WAITDC ^Q
&! Check for timeout
&IF &TIMEDOUT,=,TRUE
&! Try again
&GOTO STARTLOGON
&ENDIF
&ENDLOGON
&! HP 3000 Logoff
&!
&WTIMEOUT 30
BYE
&ENDLOGOFF
&REMOTE TOGGLE
&TERMINATOR ^Q
PURGE ITEMS
PURGE SALESUM
BUILD SALESUM;REC=-72,1,F,ASCII;DISC=200
FILE QSLIST=SALESUM,OLD;DEV=DISC
```

Figure 2. (continued)

```
RUN QUERY.PUB.SYS
B=AMSTER
WRITER
1
SET=ITEM-MASTER
FIND PART-NUM<>0
SAVE ITEMS
SET=DAILY-SALES
FIND PART-NUM<>0
OUT=LP
REPORT
NOPAGE
S1,PART-NUM
T1,PART-NUM,10
T1,PART-NUM,20,COUNT
T1,SELL-PRICE,30,ADD
END
EXIT
&DSCOPY ITEMS,REMOTE TO A:\AMSTERDM\ITEMS.DIF,LOCAL
&DSCOPY SALESUM,REMOTE TO A:\AMSTERDM\SALESUM.PRN,LOCAL;ASCII
PURGE ITEMS
PURGE SALESUM
&REMOTE TOGGLE
&EXIT
```

Figure 3. \0 Automatic 1-2-3 Keyboard Macro

```
{goto}A22~
/RE{right}{right}{right}{right}{end}{down}~
/FCCEitems~
/Ca19.e19~~
{down}/RFF0~{end}{down}~
{right}/RLL{end}{down}~
{right}{right}{right}/RFF0~{end}{down}~
{right}/Cf23.m23~{down}.{left}{end}{down}{right}~
/RNCquan_sold~g23~/RNCquan_sold~.{end}{down}~
{right}/RNCtotal_sales~h23~/RNCtotal_sales~.{end}{down}~
{right}{right}/RNCavg_margin~j23~/RNCavg_margin~.{end}{down}~
{right}/RNCavg_discount~k23~/RNCavg_discount~.{end}{down}~
{right}{right}/RNCorder_cost~m23~/RNCorder_cost~.{end}{down}~
{goto}U23~
/RE{right}{right}{end}{down}~
/FINsalesum~
/RNCsales_data~U23.W23~
/RNCsales_data~{end}{down}~
{home}{calc}
```

Figure 4. Item Data as Displayed in 1-2-3 Worksheet

	A	B	C	D	E
19	-----	----------------------	---------	--------	--------
20			Retail		Inventory
21	P/N	Description	Price	Cost	on Hand
22	-----	----------------------	---------	--------	--------
23	1	Advanced Widget	474.00	327.06	24
24	2	Standard Widget	329.00	184.24	32
25	3	Widget Upgrade Kit	166.00	89.64	40
26	4	Widget Accessory Kit	259.00	95.83	46
27	5	Widget Customizer Kit	477.00	181.26	20
28	6	Widget Tourque Adapter	244.00	148.84	33
29	7	Vibration Synchronizer	481.00	283.79	42
30	8	Load Factor Integrator	93.00	37.20	39
31	9	Backlash Damper	231.00	157.08	46
32	10	Quarterturn Lockwrench	384.00	203.52	23
33	11	Overlink Adapter	180.00	95.40	31
34	12	Feedback Breakstop	298.00	151.98	40
35	13	Sync Bypass Valve	259.00	173.53	21
36	14	Integration Filter	274.00	169.88	25
37	15	Widget Powerpack	419.00	209.50	41
38	16	Manual Linkstop	289.00	205.19	48
39	17	Linkpin Adjuster	240.00	141.60	40
40	18	Linkpin Remover	140.00	65.80	39
41	19	Linkpin Installer	198.00	146.52	41
42	20	Linkstop Filter Kit	94.00	47.94	34
43	21	Linkpin Sychronizer	356.00	263.44	26
44	22	Linkpin Reserve Clip	60.00	41.40	42
45	23	Linkpin Overstop Clip	83.00	29.88	43
46	24	Automatic Linkstop	450.00	220.50	43
47	25	Widget Linkpin Holder	440.00	184.80	42
48	26	Linkpin Cable Assembly	373.00	205.15	27
49	27	Widget Power Adapter	125.00	85.00	37
50	28	Widget Dust Cover	40.00	18.80	35
51	29	Deluxe Power Adapter	416.00	149.76	25
52	30	Deluxe Dust Cover	76.00	34.20	30
53	31	Synchronizing Linkpin	464.00	343.36	40
54	32	Syncrhonizer Kit	175.00	129.50	37
55	33	Linkpin	297.00	219.78	46
56	34	Reverse Sync Linkpin	367.00	168.82	22
.	.	. . .	.	.	.
.	.	. . .	.	.	.
.	.	. . .	.	.	.
.	.	. . .	.	.	.
70	48	Powered Reflex Tool	472.00	254.88	34
71	49	Reflex Tool (220V)	478.00	282.02	40
72	50	Slipring Backstop	261.00	182.70	36

Figure 5. Sales Summary Data in Lotus 1-2-3 Worksheet

	U	V	W
19	--------	--------	--------
20			Total
21	P/N	Quantity	Sales
22	--------	--------	--------
23	1	5	2,062.03
24	2	24	6,806.88
25	3	2	332.00
26	4	10	1,821.63
27	5	11	4,074.44
28	7	12	5,083.16
29	10	20	6,196.95
30	11	19	2,935.59
31	13	16	3,873.29
32	14	24	5,957.97
33	15	18	6,440.65
34	21	11	3,718.22
35	22	11	583.49
36	23	9	645.75
37	24	7	2,729.83
38	25	15	5,484.38
39	26	18	6,092.03
40	27	15	1,722.61
41	28	17	594.12
42	29	15	5,054.67
43	30	13	861.79
44	31	7	3,058.16
45	32	9	1,486.52
46	33	7	1,925.64
47	34	12	3,261.12
48	35	24	10,348.40
49	36	12	3,796.48
50	37	14	786.46
51	38	22	708.48
52	39	9	1,604.48
53	40	14	1,888.88
54	47	19	2,539.11
55	48	16	6,433.01
56	49	9	3,367.05
57	50	10	2,554.68

Figure 6. Executive Summary Report in 1-2-3 Worksheet

```
              B                   C
      -------------------- ------------
  3 | SUMMARY REPORT         31-Dec-84
  4 | Total Items Sold             476
  5 | Total Daily Sales     116,829.95
  6 | Average Margin             46.2%
  7 | Average Discount            8.9%
  8 |
  9 | Order Cost             10,315.31
```

Figure 7. Calculations for Executive Summary Report

```
B3: 'SUMMARY REPORT
C3: (D1) @TODAY
B4: 'Total Items Sold
C4: (F0) @SUM(QUAN_SOLD)
B5: 'Total Daily Sales
C5: @SUM(TOTAL_SALES)
B6: 'Average Margin
C6: (P1) @AVG(AVG_MARGIN)
B7: 'Average Discount
C7: (P1) @AVG(AVG_DISCOUNT)
B9: 'Order Cost
C9: @SUM(ORDER_COST)
```

Figure 8. Computed Sales and Inventory Table

G	H	I	J	K	L	M
Quanity Sold	Total Sales	Average Price	Average Margin	Average Discount	Suggested Order	Order Cost
5	2,062.03	412.41	26.1%	13.0%	9	2,943.54
24	6,806.88	283.62	53.9%	13.8%	0	0.00
2	332.00	166.00	85.2%	0.0%	1	89.64
10	1,821.63	182.16	90.1%	29.7%	0	0.00
11	4,074.44	370.40	104.3%	22.3%	0	0.00
0	0.00	0.00	0.0%	0.0%	0	0.00
12	5,083.16	423.60	49.3%	11.9%	0	0.00
0	0.00	0.00	0.0%	0.0%	5	186.00
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.

Figure 9. Sales and Inventory Table Calculations

```
F23: (F0) @IF(A23=@VLOOKUP(A23,$ORDER_POINT,0),    <<Order Point>>
               @VLOOKUP(A23,$ORDER_POINT,1),@NA)
G23: (F0) @IF(A23=@VLOOKUP(A23,$SALES_DATA,0),   <<Quanity Sold>>
               @VLOOKUP(A23,$SALES_DATA,1),0)
H23: @IF(A23=@VLOOKUP(A23,$SALES_DATA,0),          <<Total Sales>>
          @VLOOKUP(A23,$SALES_DATA,2),0)
I23: @IF(+G23<>0,+H23/G23,0)                <<Average Sell Price>>
J23: (P1) @IF(G23<>0,+I23/D23-1,0)              <<Average Margin>>
K23: (P1) @IF(G23<>0,1-(+I23/C23),0)          <<Average Discount>>
L23: (F0) @IF(A23=@VLOOKUP(A23,$ORDER_POINT,0), <<Purchase Quan>>
               @MAX(0,+F23-E23),0)             <<Purchase Amount>>
M23: +L23*D23
```

Figure 10. Local Order Point Table from 1-2-3 Worksheet

P/N	Quantity
1	33
2	30
3	41
7	28
8	44
9	30
11	22
12	28
14	20
15	35
16	24
17	50
23	28
25	24
26	16
27	24
28	45
29	21
31	31
34	13
35	29
38	38
39	32
40	32
41	40
42	38
44	43
45	24
46	21
47	25
48	40
49	34
50	30

Figure 11. AMSTERDM.BAT Batch Command File

```
. This Batch file is a demonstration prepared to accompany a paper
. submitted to the INTEREX 1985 International Conference in Amsterdam.
. It will log on to the HP 3000, download files from a database, then
. call 1-2-3 by Lotus.  If you want to cancel this operation, please
. type ^C now, otherwise
PAUSE
CD AMSTERDM
ERASE ITEMS.DIF
ERASE SALESUM.PRN
CD ..
ADVLINK &CHAIN \AMSTERDM\AMSTERDM.ALK
CD AMSTERDM
PATH A:\
DIFWKS ITEMS ITEMS -c
ERASE AUTO123.WKS
RENAME AMSTERDM.WKS AUTO123.WKS
CD ..
ERASE SAVE123.CNF
RENAME 123.CNF SAVE123.CNF
COPY \AMSTERDM\AMSTERDM.CNF 123.CNF
123
CD AMSTERDM
ERASE AMSTERDM.WKS
RENAME AUTO123.WKS AMSTERDM.WKS
ERASE ITEMS.WKS
ERASE ITEMS.DIF
ERASE SALESUM.PRN
CD ..
ERASE 123.CNF
RENAME SAVE123.CNF 123.CNF
EXIT
```

Figure 12. AMSTERDM.IN$ P.A.M. Install File

```
Amsterdm Demo
A.01.00
CALL.COM
COMMAND.COM /C AMSTERDM.BAT
19000
COMMAND.COM
CALL.COM
AMSTERDM.BAT
```

TITLE: CONVERTING SPL PROGRAMS FROM THE HP3000 TO THE HP150

SPEAKER:

Jacques Van Damme
SYDES N.V.
BELGIUM

ABSTRACT:

This session tells the story of the conversion of fifty thousand lines of SPL code to assembly language for the HP150.

It deals with the (lack of) possibility of re-writing in PASCAL, and presents an alternate solution that is allmost generaly applicable.

Many interesting aspects about the internal structure of the HP150 and the difference whith the HP3000 are treated.

This session interests all people who want to know more about the HP150.

This text was prepared and printed using the MAKEUP word-processing system

1. Introduction.

What solutions do you have, when you must convert 50,000 lines of SPL code from the HP3000 to the HP150.

What solutions do you have when the programs concerned in this conversion call a large variety of MPE intrinsics.

What solutions do you have if you want to write programs for both the HP3000 and the HP 150, and you only want to maintain a single source program.

We have found a vary interesting solution to this problem. This solution is fast, safe and largely applicable. Even extensions towards programs written in other languages, such as COBOL and FORTRAN could be realised, using the same techniques.

2. The PASCAL solution.

The first, and most obviuos way to solve a conversion problem is to choose a programming language that is 'portable'.

In our case that would have been PASCAL. So we acquire a pascal compiler for the HP3000 and start doing some performance tests.

Problem number one is to translate the many SPL constructs into equivalent PASCAL structures. Try it out whith bit concatenations were each word on the stack may contain a variety of bit compositions and you will find that PASCAL is not very handy for this type of problem.

Problem number two is far more important. The CPU time taken by the PASCAL equivalent of the SPL programs is exactly doubled. The main reason for this is that PASCAL checks the validity of the source variable before storing it into the destination, while the SPL compiler assumes that this is the programmer's responsibility.

The net result is that we could not afford to run the PASCAL version of the programs on the HP3000.

3. Other available solutions.

The only other solutuion we could come up whith, was to re-write all the code in PASCAL for the HP150 and maintain two sources of the same set of programs.

Needless to say that re-writing in assembler was not considered a feasable way. Or was it?

Open parenthesis:

> Most compilers, before they start generating machine language, convert the source code into some form of intermediate code.
> This intermediate code is known as 'Inverse Polish Notation' and is extremely suited to generate code for a stack machine.
> So we can ask the question: 'Is machine language code generated for an HP3000 a general code that could be used as the source for generating any assembly language?'.

Close parenthesis.

4. Our solution.

So we decided to investigate if it would be possible to convert HP3000 machine language (this is a PREPped program file) into HP150 assembly language.

4.1 Emulating the code.

The first problem is to evaluate whether all needed registers and assembly language instructions are present in the HP150 to replace the HP3000 registers and machine instructions.

The registers on the HP3000 are:

DL, DB, Q, S, Z for the data-segment

PB, P , PL for the code segment

X the index register

ST the status register

The register DL, Z, PB and PL are only used for bounds checking and can be omitted from the emulation whithout to much danger. If the program runs on the HP3000 whithout bounds violation, and the conversion works ok, then no bounds violation should occur on the HP150.

The P register's place is taken by the program counter (PC). The problem hereby is that the HP3000 always increments the program counter by 1 (the double word instructions are not relevant in this conversion), while the HP150 has instructions of various sizes.

This problem was solved by translating all addressing into labels.
A special problem occurs here whith the SPL CASE and SWITCH constructs These will be covered later.

Now let's see how we can emulate the stack registers.

The HP150 has a stack mechanism in its assembly language. However there are only two (TWO!) instructions that handle the stack: PUSH and POP. Now this is not very funny.

So we had to emulate the stack by other registers:

```
S   is replaced by BX,
Q               by BP,
X               by SI
```

and DB is assumed to be zero all the time.

All offsets to registers except for byte addressing instructions must be doubled since the HP150 addresses byte all the time.

A first glance to the example in the appendix of this text may give you a better idea.

4.2 Emulating the instruction set.

For each instruction of the HP3000 we had to find the equivalent HP150 instruction or set of instructions. This is very simple for simple instructions such as INCM or DECM. But even the simple stack operations like LOAD and STOR require the adjustement of the BX register that replaces S.

Therefore an algorithm was designed that keeps the Top od Stack values into the registers AX,CX and DX. Only when these registers are needed for other purposes or when the Top of Stack must be saved into memory, extra code is generated.
This optimisation algorithm also recognises specific sequences of HP3000 instructions and converts them into one or more assembler instructions.
This is the case for the CASE and SWITCH constructs, but also for sequences like

```
LOAD DB+ ..
CMPI 0
```

The more complicated instructions like SCU or SCW are emulated by a set of instructions in a loop. The very specialised instructions like MTBA or MTBX are emulated by calling an assembly language routine.

Many examples of these techniques can be found in the appendix.

4.3 Emulating the MPE intrinsics.

To emulate the MPE intrinsics, we wrote a set of assembly language modules that have the same calling sequence as the corresponding MPE intrinsic, and are written in HP150 assembler.

It was amazing how simple some of the intrinsics were converted. These simple ones include

FOPEN, FCLOSE, FREAD, FWRITE, PRINT, READ

The more dificult ones are intrinsics like

FCONTROL, PRINTFILEINFO

but since we use only a small part of their possibilities, we only emulate the function we actualy use.

5. Different steps in the conversion.

The conversion is perfomed in different steps.

5.1 The programs are compiled whith the MAP option, and the compiler list output is saved into a disc file.

5.2 The programs are prepped whith the FPMAP option.

5.3 An adapted version of the contributed program DECOMP is run. This program generates machine language code from a prepped program.

The main changes made to this program are:

- use the FPMAP information in the program file for all pcal instructions
- flag the entry point and the start of code for each procedure
- flag all word move, byte move and byte compare sequences
- flag all instructions that are probably data
- flag all SPL SWITCH and CASE constructs.

Refer to the example in the appendix for more details.

5.4 Manualy include the SPL MAP information into the decompiled code.

5.5 Run the conversion program.

5.6 Transfer the converted code to the HP150 using dsn/link.

5.7 Assemble and link the program on the HP150 and zingo.

APPENDIX: a conversion example.

SPL CODE:

```
$control uslinit,MAP
BEGIN
INTEGER I;
DOUBLE D;
BYTE ARRAY BA(0:20);
ARRAY A(0:20);

PROCEDURE TEST(STRING,ARR,INT,DOUB);
VALUE INT;
INTEGER INT;
DOUBLE DOUB;
ARRAY ARR;
BYTE ARRAY STRING;

BEGIN
  INTEGER I,J,K;
  DOUBLE D;

  INTEGER SUBROUTINE SUB(A,B);
  VALUE A;
  INTEGER A,B;
  BEGIN
     SUB:=A+B;
     A:=B:=0;
     END;

  INT:=INTEGER(DOUB);

LOOP:
  DOUB:=DOUBLE(INT);
  CASE I OF BEGIN
     GO LOOP;
     J:=J+1;
     SUB(I,J);
     END;
  MOVE STRING:="ABCDEFGH";
  MOVE ARR:="IJKLMNO";
  STRING(7):=";";
  ARR(4):=";";
  SCAN STRING UNTIL°8/" ",8/";"§,1;
  SCAN * WHILE " ",1;
  K:=TOS-àSTRING;
  J:=I.(2:9);
  J:=J CAT I(2:3:4);
  END;

TEST(BA,A,I,D);
END.
```

DECOMPILED hp3000 MACHINE INSTRUCTIONS

```
000.00000 000000      s   SEG'

     A                        ARRAY          LOGICAL     DB+004
     BA                       ARRAY          BYTE        DB+003
     D                        SIMP. VAR.     DOUBLE      DB+001
     I                        SIMP. VAR.     INTEGER     DB+000
     TERMINATE'               PROCEDURE
     TEST                     PROCEDURE
note: the above information was copied from the compiler output.

   0.0      041003  B.    LOAD  DB+003        é= OB'
   0.1      041004  B.    LOAD  DB+004
   0.2      041000  B.    LOAD  DB+000
   0.3      171001  ..    LRA   DB+001
   0.4      031002  2.    PCAL  TEST
   0.5      031003  2.    PCAL  TERMINATE'

     ARR                      ARRAY     (R)  LOGICAL     Q -006
     D                        SIMP. VAR.     DOUBLE      Q +004
     DOUB                     SIMP. VAR.(R)  DOUBLE      Q -004
     I                        SIMP. VAR.     INTEGER     Q +001
     INT                      SIMP. VAR.     INTEGER     Q -005
     J                        SIMP. VAR.     INTEGER     Q +002
     K                        SIMP. VAR.     INTEGER     Q +003
     LOOP                     LABEL                      PB+013
     STRING                   ARRAY     (R)  BYTE        Q -007
     SUB                      SUBROUTINE     INTEGER     PB+000

     A                        SIMP. VAR.     INTEGER     S -002
     B                        SIMP. VAR.     INTEGER     S -001
   0.6      041702  C.    LOAD  S- 002        é. TEST
note: start of code for subroutine test is indicated by é. sign.
   0.7      073702  w.    ADDM  S- 002,I
   0.10     051704  S.    STOR  S- 004
   0.11     000645  ..    ZERO, DUP
   0.12     053703  W.    STOR  S- 003,I
   0.13     051703  S.    STOR  S- 003
   0.14     032002  4.    SXIT  %2
   0.15     035005  :.    ADDS  %5       é- TEST
note: entry point is indicated by é- sign.
   0.16     153604  ..    LDD   Q- 004,I
   0.17     002701  ..    DTST, DELB
   0.20     051605  S.    STOR  Q- 005
   0.21     041605  C.    LOAD  Q- 005
   0.22     000600  ..    ZERO, NOP
   0.23     012120  .P    DASR  16 BITS
   0.24     163604  ..    STD   Q- 004,I
   0.25     041401  C.    LOAD  Q+ 001
   0.26     021003  ".    LDI   3
   0.27     002657  ..    STBX, LCMP
   0.30     141621  ..    BGE   P+ 021      <<=51>>
   0.31     174015  ..    LRA   P+ 015,X    <<=46>>
   0.32     074014  x.    ADDM  P+ 014,X    <<=46,%177766,-10,'..'>>
   0.33     032000  4.    SXIT  %0
```

```
  0.34     140413  ..    BR    P- 013     <<=21>>
  0.35     120402  ..    INCM  Q+ 002
  0.36     140013  ..    BR    P+ 013     <<=51>>
  0.37     000600  ..    ZERO, NOP
  0.40     041401  C.    LOAD  Q+ 001
  0.41     171402  ..    LRA   Q+ 002
  0.42     170002  ..    LRA   P+ 002     <<=44>>
  0.43     140435  ..    BR    P- 035     <<=6>>
  0.44     004000  ..    DEL , NOP
  0.45     140004  ..    BR    P+ 004     <<=51>>
  0.46     177766  ..    LRA   S- 066,I,X
  0.47     177766  ..    LRA   S- 066,I,X
  0.50     177767  ..    LRA   S- 067,I,X
  0.51     173607  ..    LRA   Q- 007,I
  0.52     170003  ..    LRA   P+ 003     <<=55>>
  0.53     010201  ..    LSL   1 BIT
  0.54     140005  .. b  BR    P+ 005     <<=61>>
note: the above instruction is flagged as being part of a move.
  0.55     040502  AB d  DW 'AB'
  0.56     041504  CD d  DW 'CD'
  0.57     042506  EF d  DW 'EF'
  0.60     043510  GH d  DW 'GH'
note: The above instructions are flagged as data.
  0.61     021010  ".    LDI   8
  0.62     020043   £    MVB   PB-DB SDEC=3
  0.63     173606  ..    LRA   Q- 006,I
  0.64     170002  ..    LRA   P+ 002     <<=66>>
  0.65     140005  .. b  BR    P+ 005     <<=72>>
  0.66     044512  IJ d  DW 'IJ'
  0.67     045514  KL d  DW 'KL'
  0.70     046516  MN d  DW 'MN'
  0.71     047440  O  d  DW 'O '
  0.72     021004  ".    LDI   4
  0.73     020003   .    MOVE  PB-DB SDEC=3
  0.74     021073  ";    LDI   59
  0.75     021407  £.    LDXI  7
  0.76     166607  ..    STB   Q- 007,I,X
  0.77     021073  ";    LDI   59
  0.100    021404  £.    LDXI  4
  0.101    057606  _.    STOR  Q- 006,I,X
  0.102    173607  ..    LRA   Q- 007,I
  0.103    040017  à.    LOAD  P+ 017     <<=122,%020073,8251,' ;'>>
  0.104    020161   q    SCU   SDEC=1
  0.105    021040  "     LDI   32
  0.106    020121   Q    SCW   SDEC=1
  0.107    101607  ..    SUBM  Q- 007
  0.110    051403  S.    STOR  Q+ 003
  0.111    041401  C.    LOAD  Q+ 001
  0.112    026451  -)    EXF   (2:9)
  0.113    051402  S.    STOR  Q+ 002
  0.114    041402  C.    LOAD  Q+ 002
  0.115    041401  C.    LOAD  Q+ 001
  0.116    026464  -4    EXF   (3:4)
  0.117    027044  .$    DPF   (2:4)
  0.120    051402  S.    STOR  Q+ 002
  0.121    031404  3.    EXIT  %4
  0.122    020073   ;    MVB   DB-DB SDEC=3
  0.123    177777  ..    LRA   S- 077,I,X
```

```
  0.124   177777     e  TERMINATE'
  0.125   000015     p  TEST
  0.126   000000     p  OB'
note: The above lines are flagged as externals or procedures.
  0.127   040003     t  SEG'
```

HP150 assembly language code.
All the following code is generated 100% automaticaly.
HP3000 machine language is included as comments automaticaly.

```
 EXTRN DCMP:FAR
 EXTRN DMUL:FAR
 EXTRN DDIV:FAR
 EXTRN MVBW_A:FAR
 EXTRN MVBW_AN:FAR
 EXTRN MVBW_AS:FAR
 EXTRN MVBW_ANS:FAR
 EXTRN MVBW_N:FAR
 EXTRN TERMINATE_:FAR
 PUBLIC TEST
 PUBLIC OB_
SEG_  SEGMENT
       ASSUME CS:SEG_
                                      ; s   SEG'
OB_:
        MOV  AX,SS                    ;     LOAD  DB+003       é= OB'
        MOV  DS,AX
        MOV  ES,AX
        MOV  BP,86
        MOV  BX,BP
A0    : MOV  AX,DS:°6§                ;>    LOAD  DB+003       é= OB'
        MOV  DX,DS:°8§                ;     LOAD  DB+004
        MOV  CX,DS:°0§                ;     LOAD  DB+000
        INC  BX                       ;     LRA   DB+001
        INC  BX
        MOV  °BX§, AX
note: the BX register is used as the stack pointer.
        MOV  AX,1
        INC  BX                       ;     PCAL  TEST
        INC  BX
        MOV  °BX§, DX
        INC  BX
        INC  BX
        MOV  °BX§, CX
        INC  BX
        INC  BX
        MOV  °BX§, AX
        ADD  BX,8
        MOV  °BX§,BP
        MOV  BP,BX
note: the above 3 instructions build a stack marker.
        CALL FAR PTR TEST
        ADD  BX,8                     ;     PCAL  TERMINATE'
        MOV  °BX§,BP
        MOV  BP,BX
        CALL FAR PTR TERMINATE_
```

```
A6     : MOV  AX,°BX-4§          ;>    LOAD  S- 002        é. TEST
note: TOS is now in AX register.
         MOV  DI,°BX-2§          ;     ADDM  S- 002,I
         SHL  DI,1
         ADD  AX,°DI§
         MOV  °BX-6§,AX          ;     STOR  S- 004
         XOR  AX,AX              ;     ZERO, DUP
note:  the  DUP  instruction doesn't generate any code because we keep
      the TOS elements in registers.
         MOV  DI,°BX-2§          ;     STOR  S- 003,I
         SHL  DI,1
         MOV  °DI§,AX
         MOV  °BX-4§,AX          ;     STOR  S- 003
         MOV  AX,°BX§            ;     SXIT  %2
         SUB  BX,8
         JMP  AX
TEST:
A15    : ADD  BX,10              ;>    ADDS  %5        é- TEST
         MOV  DI,°BP-8§          ;     LDD   Q- 004,I
         SHL  DI,1
         MOV  DX,°DI§
         INC  DI
         INC  DI
         MOV  AX,°DI§
         INC  BX                 ;     DTST, DELB
         INC  BX
         MOV  °BX§, DX
         INC  BX
         INC  BX
         MOV  °BX§, AX
         XOR  CX,CX
         XOR  DI,DI
         CALL FAR PTR DCMP
         MOV  AX,°BX§
         DEC  BX
         DEC  BX
         DEC  BX
         DEC  BX
         MOV  °BP-10§,AX         ;     STOR  Q- 005
A21    : MOV  AX,°BP-10§         ;>    LOAD  Q- 005
         XOR  DX,DX              ;     ZERO, NOP
note: TOS is now in DX, TOS-1 is in AX.
         XCHG AX,DX              ;     DASR  16 BITS
         MOV  CX,16
         SAR  DX,1
         RCR  AX,1
         LOOP $-4
         MOV  DI,°BP-8§          ;     STD   Q- 004,I
         SHL  DI,1
         MOV  °DI§,DX
         INC  DI
         INC  DI
         MOV  °DI§,AX
         MOV  AX,°BP+2§          ;     LOAD  Q+ 001
         MOV  DX,3               ;     LDI   3
         MOV  SI,AX              ;     STBX, LCMP
         CMP  AX,DX
         JAE A51                 ;     BGE   P+ 021      <<=51>>
```

```
                           ; c   LRA    P+ 015,X    <<=46>>
        MOV  DI,OFFSET A46 ; c   ADDM   P+ 014,X    <<=46,%177766,-10,'..'>>
        ADD  DI,SI
        ADD  DI,SI
        MOV  AX,CS:°DI§ .
        JMP  AX            ; c   SXIT   %0
```
note: the above sequence is recognised as being an SPL case construct.

```
A34    : JMP  A21                 ;>    BR     P- 013      <<=21>>
A35    : INC WORD PTR °BP+4§      ;>    INCM   Q+ 002
         JMP  A51                 ;     BR     P+ 013      <<=51>>
A37    : INC  BX                  ;>    ZERO,  NOP
         INC  BX
         MOV WORD PTR °BX§,0
         MOV  AX,°BP+2§           ;     LOAD   Q+ 001
         LEA  DX,°BP+4§           ;     LRA    Q+ 002
         SHR  DX,1
         INC  BX                  ;     LRA    P+ 002      <<=44>>
         INC  BX
         MOV  °BX§, AX
         INC  BX
         INC  BX
         MOV  °BX§, DX
         INC  BX
         INC  BX
         MOV WORD PTR °BX§,OFFSET $+7
         JMP  A6                  ;     BR     P- 035      <<=6>>
         DEC  BX                  ;     DEL  , NOP
         DEC  BX
         JMP  A51                 ;     BR     P+ 004      <<=51>>
A46    : DW   A34                 ;>a   LRA    S- 066,I,X
         DW   A35                 ; a   LRA    S- 066,I,X
         DW   A37                 ; a   LRA    S- 067,I,X
note: the above 3 instructions complete the CASE construct.
A51    : MOV  AX,°BP-14§          ;>    LRA    Q- 007,I
         MOV  SI,OFFSET A55       ;     LRA    P+ 003      <<=55>>
                                  ;     LSL    1 BIT
         JMP  $+10                ; b   BR     P+ 005      <<=61>>
A55    :
         DB   'AB','CD','EF','GH'
         MOV  CX,8                ;     LDI    8
         MOV  DI, AX              ;     MVB    PB-DB SDEC=3
         CLD
         REP MOVS BYTE PTR ES:°DI§,BYTE PTR CS:°SI§
         MOV  AX,°BP-12§          ;     LRA    Q- 006,I
         MOV  SI,OFFSET A66       ;     LRA    P+ 002      <<=66>>
         JMP  $+10                ; b   BR     P+ 005      <<=72>>
A66    :
         DW   'IJ','KL','MN','O '
         MOV  CX,4                ;     LDI    4
         MOV  DI, AX              ;     MOVE   PB-DB SDEC=3
         SHL  DI,1
         CLD
         REP MOVS WORD PTR ES:°DI§,WORD PTR CS:°SI§
         MOV  AX,59               ;     LDI    59
         MOV  SI,7                ;     LDXI   7
         MOV  DI,°BP-14§          ;     STB    Q- 007,I,X
         ADD  DI,SI
         MOV  °DI§,AL
         MOV  AX,59               ;     LDI    59
         MOV  SI,4                ;     LDXI   4
         MOV  DI,°BP-12§          ;     STOR   Q- 006,I,X
         ADD  DI,SI
         SHL  DI,1
         MOV  °DI§,AX
```

```
        MOV   AX,°BP-14§    ;     LRA    Q- 007,I
        INC   BX            ;     LOAD   P+ 017      <<=122,%020073,8251,'  ;'>
        INC   BX
        MOV   °BX§, AX
        MOV   AX,8251
        MOV   DI,°BX§       ;     SCU    SDEC=1
        DEC   BX
        DEC   BX
        CMP   °DI§,AH
        STC
        JE    $+10
        CMP   °DI§,AL
        CLC
        JE    $+5
        INC   DI
        JMP   $-11
        MOV   AX,32         ;     LDI    32
        CMP   °DI§,AH       ;     SCW    SDEC=1
        STC
        JE    $+10
        CMP   °DI§,AL
        CLC
        JNE   $+5
        INC   DI
        JMP   $-11
        SUB   DI,°BP-14§    ;     SUBM   Q- 007
        MOV   °BP+6§,DI     ;     STOR   Q+ 003
        MOV   AX,°BP+2§     ;     LOAD   Q+ 001
        AND   AX,16352      ;     EXF    (2:9)
        MOV   CL,5
        ROR   AX,CL
        MOV   °BP+4§,AX     ;     STOR   Q+ 002
                            ;     LOAD   Q+ 002
        MOV   DX,°BP+2§     ;     LOAD   Q+ 001
        AND   DX,7680       ;     EXF    (3:4)
        MOV   CL,9
        ROR   DX,CL
        AND   DX,15         ;     DPF    (2:4)
        AND   AX,-15361
        MOV   CL,10
        ROL   DX,CL
        AND   AX,DX
        MOV   °BP+4§,AX     ;     STOR   Q+ 002
        MOV   BX,BP         ;     EXIT   %4
        MOV   BP,°BX§
        MOV   AX,°BX-2§
        SUB   BX,16
        AND   AH,3
        CMP   AH,2
        RCR   AH,1
        DB 0CBH
                            ;<    MVB    DB-DB SDEC=3
        MOV   AX,°BX-126§   ;     LRA    S- 077,I,X
        ADD   AX,SI
        INC   BX            ; t   SEG'
        INC   BX
        MOV   °BX§, AX
SEG     ENDS
```

```
STACK   SEGMENT STACK
        DW   0,0,0,10,16,0,0,0,0,0,0,0,0,0,0,0,0,0
        DW   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DW   0,0
        DW   1200 DUP (?)
STACK   ENDS
note: the stack segment is created from the PROG file.
        END OB_
```

GROWING NEED FOR HP3000 -TO- MICRO

David C. Dummer
IMX Systems Corporation
Los Angeles, California U.S.A.

Henny van Beek
IMX Systems Corporation (Nederland) BV
Nieuwegein, The Netherlands

This paper reviews the role of the HP3000 as a file server for micro-computersystems. Delivery of the paper at the Amsterdam conference will concentrate on details of the DataExpress product and will include practical experiences that the author has had with DataExpress. The author will also mention the latest enhancement of DataExpress as well as future directions.

DataExpress was introduced by IMACS Systems Corporation two years ago. One of the major disign goals for DataExpress was to have a general purpose data extraction facility which was making full use of major dictionary and directory facilities on the HP3000. When designing such a product we were aware of the fact that there was, and at that time still is a proliferation of microcomputers and spreadsheetsystems amongst the end-users. This made the support of multiple types of microcomputers an absolute nessecity. Because of supporting multiple types of microcomputers DataExpress had to be able to generate all standard interface file formats for major microcomputer applications like Visicalc, lotus 1-2-3, Dbase II as well as formats for HP3000 applications like HP3000 based spreadsheets and DSG. DataExpress is very easy to use for the end-user, and allows the DataExpress management to customize at system level and also at user level. Because DataExpress is using a message catalog for menues, messages and help facility it is possible to tailor DataExpress to the language and terminology needed by the end-user.

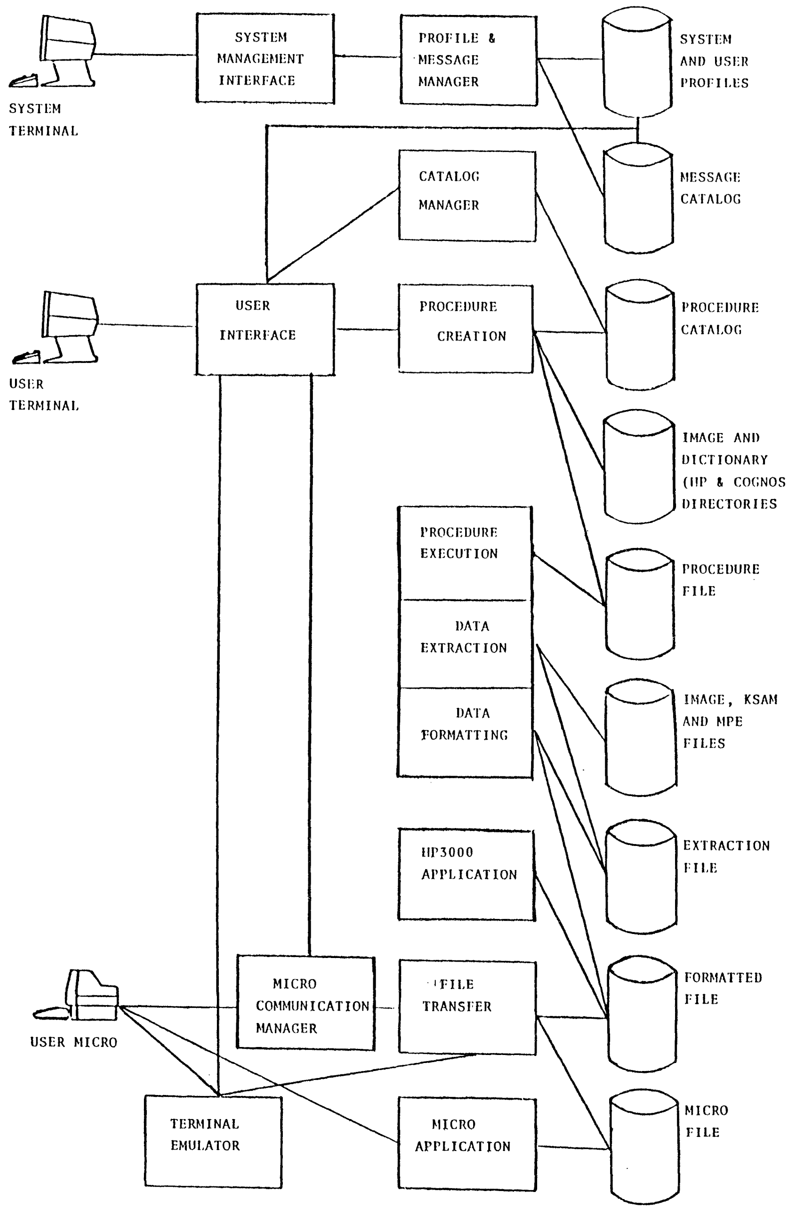
DataExpress is using a simple menu-driven command structure so the end-user can specify, through an intelligent dialogue, which HP3000 source files he wants to access. To assist the novice user of the DataExpress system, a two level "help" facility will provide a comperhensive step by step guidance. DataExpress uses information in Dictionary/3000, Cognos dictionary, INFORM groups or IMAGE root files to provide the user with menues from which he has to select data bases, sets, files and fields. If the user wants to access files on which the system also no information DataExpress prompts the user for input descriptions of the files and the desired fields. The selected data along with logical, textual, arithmetic operations and field combinations is used to generate the output file. Separating the specification of the desired data from the actual manipulation of the source files, the DataExpress manager can effectively control and limit access to files by end-users. This is done by making only the run time version of a DataExpress program available to the end-user. The DataExpress manager can also limit the capabilities of end-users in respect of access, features and volume of data accessed. The format of the DataExpress output file may be specified to conform to the requirements of a range of standard micro-computer applications or may be specified by the end-user or DataExpress manager to any desired format.

Author Profile

David Dummer is president of IMACS Systems Corporation which develops and markets software products, education and consulting services through offices in the United States, Canada, the United Kindom and the Netherlands. The company has specialized in productivity tools for the HP3000 computer system since 1977. Four of its software products were required by Hewlett-Packard in 1981, enhanced and now marketed as the RAPID product family of Dictionary, Transact, Report and Inform.

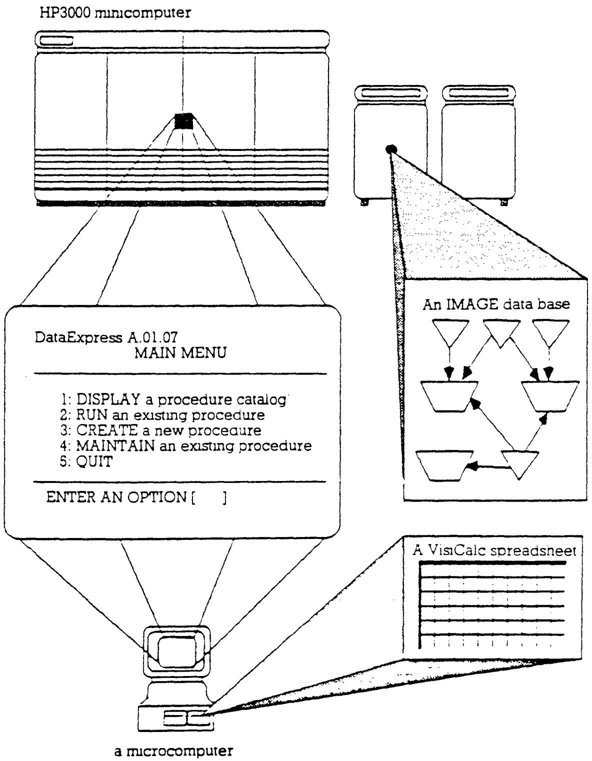
David has personally spent nineteen years in data processing most of which have been in the research, development and marketing of data base and dictionary driven systems for mainframe and mini-computer system environments. For several years as a data base administrator he gained considerable data dictionary driven systems. He has lectured extensively in North America and Europe on technologies and implementation methodologies associated with the data base approach.

DataExpress FUNCTIONAL OVERVIEW



A Pictorial Overview

To give you a better understanding of exactly what DataExpress will do for you, the following chart illustrates the basic concepts.



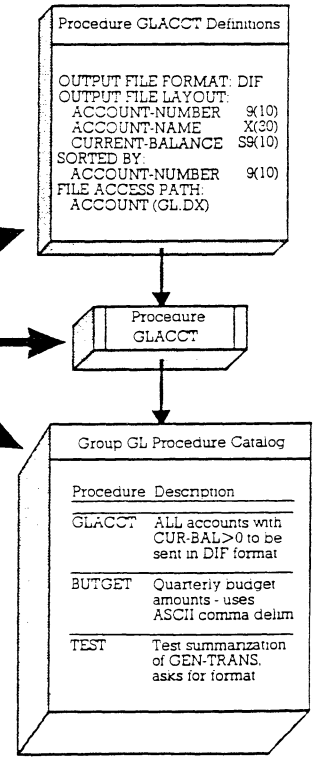
Data Express runs on the HP3000, but is run by the user on a microcomputer.

The user works by creating procedures which manipulate selected fields.

These procedures may be run, and then stored in a catalog for reuse.

Information in an IMAGE data base can be extracted, formatted and downloaded into a DIF file, which can then be loaded directly into a VisiCalc spreadsheet

The microcomputer must be connected to the HP3000 and run a terminal emulator program that can download ASCII text files.



THE PLACE OF THE MICRO IN AN H-P 3000 NETWORK

E. S. Fisher
Wellington Management Company
Boston, Massachusetts, USA

Summary

Data Processing managers throughout the industry are facing one of the most significant challenges to their authority and management skill ever to arrive on the computer scene. The microprocessor "revolution" is providing unprecedented numbers of office professionals with access to computer power undreamed of as recently as five years ago.

This access to and experience with computers has had several effects on the office environment:

- The general level of "computer literacy" has increased;
- Users are becoming less frightened of computer terminals;
- People are becoming aware of the MIS information available.

With the increase in awareness is coming an increasing demand to have management information from corporate data processing available to microcomputer users. Corporate data processing managers have several options available to them to deal with this demand:

- They can ignore the problem and hope it will disappear;
- They can absolutely forbid any interconnection;
- They can take the lead in assisting users and avoid problems.

This paper will deal with the problems and possibilities associated with DP management taking an active rôle in the selection, support and interconnection of microcomputers in an H-P 3000 environment. Some mention is made of the products we have used, namely VisiCalc, Lotus 1-2-3, IBM PC, MS-DOS, H-P 150, PC2622, DSN/Link, and PowerHouse. However, the intent of this paper is to emphasize the standardization process more than to advocate specific product use.

A BRIEF HISTORY OF THE MICROCOMPUTER IN THE OFFICE

Once upon a time, there was no such thing as a microcomputer. Amazingly enough, this time is a part of the very recent past. In those days, computers were sacred monsters residing in a holy temple attended by a priesthood of magicians spouting arcane gibberish. Once in a great while, something useful would come out of the temple. Maybe it would be a document (sometimes called a "report"); sometimes it would be an interesting display on a funny television-like device called a "CRT Terminal". What everybody knew about that computer was that no ordinary mortal could ever hope to understand what went on inside the room. Furthermore, no one could ever expect the magicians inside to understand the problems of ordinary mortals. People joked about "Murphy's Law" and continually complained about promises broken, delays in delivery, and programs that did not do what was needed. We are all too familiar with that world and with the mutual hostility engendered by the communication problems between DP professionals and other office workers.

Then one day, a strange looking beige box with a keyboard sticking out of the front and a TV sitting on top began to appear in certain electronic specialty stores. It was fairly expensive, and had a very funny name (What's an Apple?). It could be programmed in BASIC (ho hum). It could play games (so can much less expensive toys). Educators marveled at the potential to teach kids about computers (so what?). It could do word processing (The printer is more expensive than the computer, and my secretary does a better job).

Then came another day, and VisiCalc appeared. ***VisiCalc!!!*** Suddenly all of those special financial reports that higher management had been waiting for years to get from the DP department could be done in hours without ever having to talk to one of those obnoxious computer magicians! The rest is not really history, but present. Electronic spreadsheet programs have revolutionized data processing by making it possible for people unskilled in computer programming to use computer power very effectively. The effects of this revolution are being felt in DP departments around the world as DP professionals suddenly find themselves talking to users who know enough about computers so that they are no longer in awe of the DP department, who have become accustomed to half-second response times, who have found out that computer programs can indeed be easy to use, and who have become tired of keying numbers into their electronic spreadsheets from computer printouts produced by their departmental or corporate computer.

PREPARING FOR THE MICROCOMPUTER ONSLAUGHT

The appearance of microcomputers in any type of organization should come as no surprise. As sales soar, they are found in more and more businesses and offices. As they become more common, it will be only a matter of time before DP managers, especially those of H-P 3000 shops, are asked to connect them to the H-P 3000. Such a request will force these managers to take a hard look at the alternatives. The options seem to be clear:

- Ignore the fact that personal computers exist;
- Refuse to allow them to be connected to the H-P 3000;
- Set software and hardware standards for micros to be connected.

The first option is really not possible. Personal computers do indeed exist. There will only be more of them, not less. DP managers who ignore them will find that their users (including possibly their bosses) will be asking them very hard questions.

The second option is only slightly less possible. There is a lot of truth in the old myths about data processing. We all seem to be working on at least a two-year production backlog for new systems and programs. Fourth-generation languages solve many problems and greatly increase productivity, but one of Murphy's corollaries seems to be that no matter how successfully we work to increase our productivity and reduce our backlog, the only result is that our users find more things to ask us to do. This is wonderful for our job security, but frustrating for our users. Personal computers offer a golden opportunity to allow our users to participate with us in system development. This participation increases our productivity by delegating some of the programming to our users. At the same time, it reduces the frustration level of those same users by allowing them to create their own reports in their own ways without going through the agony of trying to explain their needs to someone else.

The third option, besides being the one most likely to make DP managers heroes in the eyes of their users, is also the one most likely to create a co-operative environment in the office. As has already been said, our users participate with us in the system development process. The questions most likely to slow down the neophyte in the personal computer field concern hardware and software selection. By answering these questions in advance, setting standards allows us to proceed more quickly to the really important questions concerning the ways a microcomputer will really benefit the user.

In short, the best preparation is acceptance of the inevitable plus some time spent becoming familiar with the machines and software available. No one in the firm is more familiar with its information needs than the data processing department. It is against the background of these needs that potential microcomputer applications should be evaluated. No one in the firm is more familiar with the capabilities and limitations of the firm's central computer equipment than the data processing department. It is in this environment that networked microcomputer applications must operate, and thus data processing personnel must be involved in microcomputer software and hardware selection if the network is to provide the benefits desired.

A BRIEF HISTORY OF MICRO DEVELOPMENT AT WELLINGTON MANAGEMENT

Wellington Management Company is a financial investment management firm providing investment management and counselling services to private funds and mutual funds. At Wellington Management, we really did not have a choice among the above options. It was company policy from the start that whatever microcomputers were obtained would be connected in some way to the H-P 3000 if desired by the microcomputer user. They arrived early at our firm, starting with Apple II and H-P 125 models in 1981. Virtually as soon as these machines arrived, attempts were made to interface them to the company's H-P 3000. Some successes were achieved. As powerful as these machines were, however, they could not easily handle the financial modelling needs of our security analysts and portfolio managers, and their ability to communicate effectively with the H-P 3000 was severely limited.

Because interconnection capability and network support were always company policy, the data processing department was involved from the start. Our early experiments with data communications, file transfer and terminal emulation provided invaluable information that allowed us to establish selection criteria and software standards for the next round of microcomputer experiments.

Our first criterion was for more native speed and processing capacity in the microcomputer itself. It was evident that a machine that could only address 128KB of memory in its most expanded form would not meet our needs for large spreadsheets, which had rapidly become our major application requirement. In addition, the time required to recalculate even a spreadsheet that could fit in such limited memory proved burdensome.

The next criterion was for better spreadsheet software. We needed a spreadsheet package that would handle more than 256 rows. We wanted one that was easier to use and had more powerful functions. It would have been nice if the new package was also easier to interface to the H-P 3000 and ran faster.

The third criterion was for better data communications support. We tried hard at this point, approximately the spring and summer of 1983, but had little success beyond finding packages that were slightly easier to use.

At that point in time, the confluence of our selection criteria could only lead to the IBM PC. It is hard today to imagine a microcomputer world without the IBM PC, but then it was a truly revolutionary machine. With its huge memory capacity and its 16-bit processor, it satisfied our requirements for speed and spreadsheet size. Along with this machine came an equally revolutionary spreadsheet package, Lotus 1-2-3. We in data processing were delighted to find out that we were not required to produce DIF (Data Interchange Format) files in order to send information to a Lotus spreadsheet. More about DIF files later.

It took some time after that for the data communications software to catch up to the rest of the environment, but we are now using the PC2622 software package from Walker, Richer and Quinn for our IBM PC's and COMPAQ's to cause them to emulate H-P terminals and provide sophisticated file transfer capabilities, along with DSN/Link for our H-P 150's.

Some numbers attest to the success of the IBM PC for our application environment: We now have over 60 microcomputers installed in our offices. Of these, 12 are directly cabled to the H-P 3000. We also have dial-up lines for the occasional use of the other microcomputers. Of our installed microcomputer base, some are COMPAQ's, some are H-P 150's, and most are IBM PC's or PC/XT's. All are equipped with large memories, communications capability, Lotus `1-2-3, and MS-DOS or PC-DOS operating systems. Microcomputer usage ranges from two hours per week to nearly full-time.

LESSONS LEARNED IN THE PROCESS

1. Microcomputer networking is a DP Department function.

This concept may seem intuitively obvious. However, it avoids trouble if there is a clear company policy statement to this effect. Responsibility for effective computer usage should always rest with the DP department, no matter what form the computer takes. The DP department, in turn, must have a supportive policy toward microcomputer users.

2. Know your microcomputer software and hardware.

DP management cannot have a supportive policy toward microcomputer users if there has been no effort to learn about the machines. Someone on the DP staff must take the time (i.e. resources, even if scarce, must be allocated) to learn the basics of available microcomputer software, hardware, and communications environments. If resources cannot be made available, and if microcomputer usage is a company policy, higher management must be informed of the support cost involved. Once DP department personnel have become familiar with the microcomputer environment, they can determine the most appropriate mixture of hardware and software for their network.

3. Standardize, Standardize, Standardize!!!

With the knowledge obtained in Lesson 2, one or more company standard environments can be designed. Consult with the present and potential microcomputer users and find out what they need. Then define standards that you know you can live with and let the users know what will be supported. This statement of policy should not be taken as a company prohibition of non-standard equipment; rather, it is a statement of support policy: your department will support this equipment. Microcomputer users who do not need DP department support generally are sufficiently sophisticated that they also do not need it. One caveat, however: Because of the complicated nature of the H-P communications environment, interconnection support should not be extended even to the sophisticated microcomputer users unless their equipment runs good terminal emulation software. The definition of "good" will vary from one installation to another; whatever it is, DP management must be very rigid about it.

4. Avoid microcomputer software that requires DIF files.

DIF, or Data Interchange Format, was developed as a means of transferring information from one microcomputer software environment to another. It is very simple in concept and extraordinarily difficult to create without a special-purpose program. Most fourth-generation report writers on the H-P 3000 can easily create disc files corresponding to the input requirements of any microcomputer software that will accept a one-record-means-one-row concept. DIF does not conform to that concept. There is software available that will reformat IMAGE and KSAM files into DIF format, but most of the advanced microcomputer software no longer requires it. The selection criteria for microcomputer software should include the ability to accept information from ordinary ASCII files.

LESSONS AS APPLIED AT WELLINGTON MANAGEMENT

We took a conservative approach to hardware and software selection. The IBM PC was chosen in the beginning because of the size of the vendor and the expected availability of Lotus and other innovative business software on the machine. When we made the choice, we could not have predicted the success of that machine. It is because of its incredible success that the software market has followed, to the extent that most new software offered for the microcomputer market is targeted to the IBM PC first, then to other machines if demand warrants. Unfortunately, this very success can lead to reduced innovation in the future. However, for the present, we have continued to build on our base of IBM PC compatible machines.

The selection of Lotus 1-2-3 has had numerous beneficial side effects. For the DP department, the biggest benefit is the ease with which information is extracted from a Lotus spreadsheet into a disc file which can then be transmitted to the H-P 3000 (uploaded), and the ease with which information can be extracted from data files on the H-P 3000 and then transmitted to a micro disc file (downloaded) which can then be read into a Lotus spreadsheet (imported).

Because of the importance of reliable and simple file transfer techniques to our microcomputer applications, finding good communication software was absolutely essential. We had to wait for it, but when it finally arrived, PC2622 from Walker, Richer and Quinn, along with DSN/Link for the H-P 150, proved to be everything we wanted.

We have found that standardization of software and hardware must extend to revision levels and machine configurations. Even with control over software and hardware distribution, it is hard to maintain compatibility from one machine to the next. We were especially hard hit when a new version of PC2622 was released that required a new version of the file transfer support program on the H-P 3000. Another time, a new version of Lotus 1-2-3 arrived that, in combination with the latest version of PC-DOS, could no longer could run spreadsheets of any useful size in 256KB of memory.

CONCLUSION: PRESENT AND FUTURE CONSIDERATIONS

What is the place of the microcomputer in an H-P 3000 network? In a word, welcome! With proper support, the information management resources of a company can be multiplied greatly by the distribution of computing power. The kinds of applications that run standalone on the micro tend to be those that consume great amounts of CPU cycles. It is a great relief not to have the H-P 3000 burdened by financial modelling.

At the same time, we must retain our perspectives on the continuing development of computer hardware and the significant cost reductions coming. A friend of mine was recently asked to prepare a cost estimate for a local area network consisting of 5 PC's with one file server. After calculating all of the costs, he came to the conclusion that his user would be better served at $20,000 less cost by purchasing an H-P 3000 Series 37 "Mighty Mouse" with five terminals! He had more available disc space, more CPU horsepower, and the known benefits of the MPE timesharing system.

I believe that the future will bring a more closely integrated environment where large computers will become more and more oriented toward database and communications network management and microprocessors will be used for "front-end" applications. I have heard the term "co-processing" used to describe such a situation. The software exists today that will allow an application to run on a microcomputer, completely independent of the mainframe, and check information on the mainframe data base only when it is unavailable on the local machine. Currently, applications must be written in third-generation languages on the microcomputer in order to support such applications. However, with the advent of system-wide dictionaries and fourth-generation languages for microcomputer systems, it is only a matter of time before such applications can be easily and quickly developed.

Word processing and graphics are fruitful areas for microcomputer application. We do not use our microcomputers this way, generally, at Wellington because our graphics and word processing requirements are beyond the capabilities of the present generation of microcomputers. However, for more limited requirements, microcomputers can be very helpful. Once again, intelligent analysis is required by knowledgeable DP personnel to fit the computer resources to the application.

Biography

Eric S. Fisher
is Director of Systems Development and a Vice President of Wellington Management Company. While working with Wellington Management, he has connected everything from old Apple II's to the latest H-P 150's to their H-P 3000 Series 68. He is responsible for systems integration and support, an area which includes interfacing with outside data network services as well as the in-house microcomputer network. He has been working with the H-P 3000 for nine of his eighteen years in data processing. Many of his projects over the past nine years have involved interfacing strange devices to H-P 3000 data communications facilities.

USING YOUR HP PC's AS HP3000 WORKSTATIONS

DALE FOLKINS

CAROLIAN SYSTEMS INTERNATIONAL

INTRODUCTION

The number one need in Data Processing is to provide users with the most benefits from the software and hardware solutions available to them. As you manage your HP3000 you have the challenge of equipping your users with the most flexible tools possible. Hewlett Packard Personal Computers offer a very good solution to this need for flexible tools. Since HP has gone to extremes to make their PC's easy to use and easy to connect to HP3000's, it seems wise to take advantage of their efforts.

The first step in deciding how to use these powerful PC's is to define what a "workstation" means to your users. Work is definately the key, that is, what work can be accomplished by someone using the terminal or PC. The other important factor is the concept of a "station", that is, a place where work is done, or, where a worker is assigned. In many business environments the station is the work area of the user, be it a manufacturing, office automation, or other type of work area. In any of these environments, the user wants to get as much work done at one station as possible. Also, for productivity reasons, a user shouldn't have to move from one station to another in order to perform the variety of tasks required to complete their work. In summary, an HP3000 workstation should be a place where an individual can accomplish a variety of business tasks.

In this presentation, I will describe how the HP150 (or the Touchscreen) Personal Computer and the HP110 (or the Portable) can be used as multi-function workstations on your HP3000. The majority of the information contained in this presentation is found in the various manuals provided by HP, I've chosen this avenue to consolidate these workstation topics. Most of the information presented will be for the HP150 but I will add HP110 related tips in pertinent areas. I will also be handing out an up-to-date addendum for those who attend my presentation in Amsterdam.

SET UP

There are three set-up configuration menus on your HP150 that help in using it as a workstation for the HP3000. They are, (1) the Global Configuration, (2) the Terminal Configuration, and (3) one or both of the Data Comm Port Configurations. The first selection of the Global Configuration menu asks you to choose whether you want the HP-150 to power on as a terminal or a computer. If you choose computer but actually want to have it function strictly as a terminal, you will have to wait until the P.A.M. menu screen has completed before you can change to terminal. On the other hand, if you choose the terminal, the HP150 will "come up" rather quickly and a simple "shift/stop" will get you to the P.A.M. menu. Also, if you want to start an HP150 application automatically at power on, you will want to set the Power On value to computer. The other important selection on the Global Configuration menu is the Remote/Serial Dev option. In this option you decide which of the Data Comm Ports to assign to the key RS232 device designations Remote and Serial. The Remote device is used in MS DOS Config to assign a Data Comm Port (ie; PORT1) for use with DSN/LINK using the device name COM1. The Serial device is used to assign a Data Comm Port for use as the LST device and is used by many software packages as the destination for printed output. If you will not be connecting an RS232 printer to your HP150, but possibly another Host computer or modem, then you would assign PORT2 to the COM2 device. Note that on the HP150B or "Touchscreen", this configuration step is simplified by using Easy Config. Many of you may also be changing the character set selection in the KEYBOARD option to your native language.

The Terminal Configuration menu has many options, but for our consideration we are only concerned with the Terminal ID selection. By default 150A will be the value, this is fine for a growing number of HP3000 software application programs but for many other applications to recognize the functions on the HP150, you'll need to set this value to 2623A. Since the HP150 emulates an HP2623A terminal, most HP3000 applications will run when the Terminal ID is set to 2623A when using HPDRAW (version II) though, you should set Terminal ID to 150A in order to get the full capability of the HP150 (especially to allow the use of an HP-IB plotter).

The HP110 has a built-in terminal emulator to facilitate connection to Host computers, especially the HP3000. This connection can either be accomplished through the 300 Baud modem or the 9 pin RS232 connection. Either way it is very simple to set up the terminal configuration. The comment, logon string and phone number fields will be set to values that make sense in your environment. Next, fill in the appropriate device (that is, Modem or Serial), then you set the following:

Terminal Configuration

Comment: TEST LOGON

Logon String:	E(:)W1"HELLO DEMO15Ø.HPACCT;TERM=1Ø"E				
Phone Number:	9,1234567			Device:	Modem
Handshake:	Enq/Ack	Parity:	Even	Baud Rate:	3ØØ
Xmit Pacing:	None	Recv Pacing:	None	Data Bits:	7
EOL Sequence:	CR	Parity Check:	On	Echo:	Off

The final set-up configuration step is to set the Data Comm Ports for your host computer or modem. The main factor here is the Baud rate, the other values may be assigned by choosing the default settings for the type of connection you will be making. Either the Full Duplex Hardwired or Full Duplex Modem menus should be selected for each of the Data Comm Ports you are using. If you choose the Full Duplex Modem for a port, the Default value for the asterisk field if RR, which displays an * on the screen when the modem signal is present. This asterisk appears near the time, either to the left for Port 1 or the right for Port 2. An ideal set up is to have a high Baud rate set for Port 1 hardwired and the correct setting for an internal or external modem attached to Port 2.

OPERATION

Although your user's applications will decide a major part of their operational concerns, here are some helpful general operation hints:

1. To reset the HP150 use the SHIFT/CTRL/RESET keys simultaneously, which causes a reload of MS DOS if set to Power On-Computer. Otherwise, it will just hard re-set the HP150 as a terminal.

2. When a user is in the P.A.M. menu, using the Terminal soft key will exit P.A.M.

3. When a user is in Terminal mode, a SHIFT/STOP will get the HP150 into the P.A.M. menu.

4. These HP150 applications use special commands to return to the P.A.M. menu:

MS DOS COMMANDS	-	PAM or EXIT
CONDOR & BASIC	-	SYSTEM
MOST OTHER APPLICATIONS	-	EXIT SOFTKEY

5. The Terminal Configuration menu soft key is accessible from all of these applications:

MS DOS COMMANDS	BASIC
MS DOS CONFIG	DSN/LINK
TERMINAL MODE	

6. HP Touch features are built into many applications and can also be added to your local applications. If for any reason you want to turn off HP Touch, press the SHIFT/CTRL/MENU keys simultaneously. The same function turns HP Touch back on again. When HP Touch is turned off, "TOUCH OFF" appears on the screen just under the seventh function key label.

INTERFACING

When interfacing HP150 applications to HP3000 applications, the main transport vehicle is DSN/LINK. The standard method is to use MPE files on the HP3000 side and MS DOS files on the HP150 side as interface media. DSN/LINK is designed to allow HP150's to talk to other HP150's, other PC's and host computers. Use with the HP3000 requires two steps, (1) Uploading the file transfer program to the HP3000 and (2) using the DSN/LINK function to do the actual file transfers. Upload transfers the latest version of a file called LINK100 to the PUB group of the SYS account. If your HP3000 doesn't have this file, you will need to obtain it via the UPLOAD program on the HP150. It is provided with your copy of DSN/LINK and you should use 2400 Baud or less for the transfer. The upload process takes a good deal of time since it transfers the file twice and does a verification of the number of records in the program. If your system manager knows that LINK100.PUB.SYS already exists, you may skip the upload step, but if you run into any problems with DSN/LINK, your HP support representative may ask you to purge the old copy of LINK100 and UPLOAD your new copy. Even if you don't plan on transferring files using DSN/LINK, you should keep the latest copy of LINK100.PUB.SYS so that HP support personnel can use their HP150's to send you new files or patches (depending on your local support office's procedures).

DSN/LINK

DSN/LINK is an easy to use file transfer program specifically written for transfers between the HP150 and the HP3000. Handling file transfers both to and from the Host 3000 is but one (or two) of it's facets. Along with a logging feature, DSN/LINK also has command file capabilities that allow a set of commands to be performed automatically whenever the program is started. In fact, when combined with P.A.M.'s autostart capability, a user need only have access to the correct diskette in order to utilize all the functions of DSN/LINK. This adds ease of use benefits beyond MPE's user defined commands since a command file can pass pre-set user responses into the application itself. (UDC's can only allow a user to start an application or provide for a single user response via the Info=parameter.) The following is a partial list of DSN/LINK command file features, for more information on these and other capabilities of the product, see the DSN/LINK reference manual.

1. Auto Logon with or without a password. (A set of commands will only be evoked when an auto logon password is correctly entered.)
2. Up to 15 commands can be included in the Auto Logon.
3. Commands can be MPE commands or DSN/LINK commands (which are preceded by an &).

SOME OF THE & COMMANDS ARE:

(a) ASSIGN - for assignment of text to a variable to be used later

(b) CHAIN - to move execution to another command file.

(c) DSCOPY - a from/to type of format for file transfers (just like the transfer menus).

(d) IF/ELSE,ENDIF - for logical sequences, alternative sequences and the conclusion of ranges of logical sequences.

(e) FASTKEY - to load function keys with labels and associated execution steps.

(f) INPUT - allows the user to input the value of an assigned variable.

(g) KEY - to load function keys with labels and an executable command file.

(h) OFFKEY - turns off any function key assignments.

(i) SEND - transmits text to the host, handles special text such as control characters or variables.

(j) WAITCLOCK - tells your HP150 to wait until a certain time before preceding with the rest of a command file.

In order to demonstrate the use of these HP150 features, I've devised a short "ease of use" example. The goal of this example is to transfer the data from a

CONDOR database on the HP150 to an IMAGE database on the HP3000. The first step was to use the WRITE command in CONDOR to make an MS DOS file. Then P.A.M. was set to auto-start DSN/LINK with the following auto logon commands:

&SEND (to get a : prompt from MPE) HELLO DSNLINK,DALE.FOLKINS

&KEY1,"CUSTIN XFER",CUST.XFR (labels softkey 1 to start the local command file named CUST.XFR)

When DSN/LINK starts, this autologon command sets up softkey 1 for execution of the command file when the softkey is pressed.

The CUST.XFR command file was actually created and written using EDITOR then the file was transferred to the HP150 via the Transfer "FROM HOST" Menu of DSN/LINK. The complete command file follows:

&INPUT "NAME OF 3000 FILE?",P1

Asks user to input the HP3000 file name and assigns it to variable &P1.

&SEND "PURGE",&P1

Purges any old copy of the file on the HP3000.

&SEND "BUILD",&P1,";REC=-174,1,F,ASCII;DISC=100"

Builds a new copy of the file.

&DSCOPY CUST.DLF TO &P1,REMOTE

Copies the file written from CONDOR to the HP3000 file.

&SEND "TRANSACT"

Activates a UDC for TRANSACT.PUB.SYS.

&SEND "CUSTER"

Runs the TRANSACT program (system)called CUSTER to load the IMAGE database.

&SEND "E"

To exit TRANSACT.

&TERMINATOR OFF

BYE

& TERMINATOR Q'

To release the HP150 from the HP3000 session.

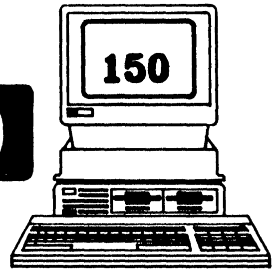
Although this is a rather simple example of using command files, it does show how easy it is to use DSN/LINK in order to improve the friendliness of your applications for your HP150 users. This also allows users to capitalize on the multi-functions of the HP150 workstation by touching only one function key (and even that step can be eliminated). These flexible command files increase the range of use for an HP150 workstation.

In conclusion, I've begun a table of HP150 application to HP3000 application interfacing guidelines. This table is not meant to be all inclusive but as a starting point for your users in their use of the HP150 as an HP3000 workstation. Please refer to the copy of my overhead slides for this compatibility table.

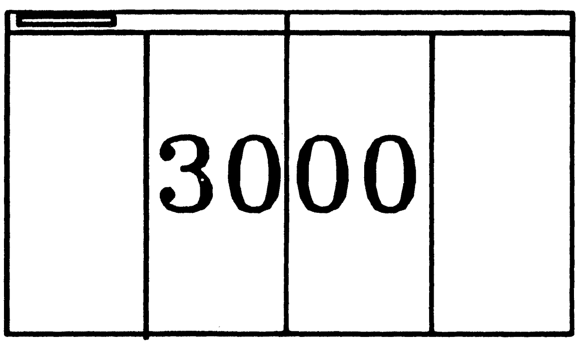
BIOGRAPHY

Dale Folkins started the Southern California Sales and Support office for Carolian Systems International last year. Dale had been a HP3000 Systems Engineer with Hewlett Packard Company in San Diego, California for over five years. Prior to HP, he was a programmer/analyst with Burroughs Corporation designing both engineering and business systems. He is a graduate of San Diego State University.

USING YOUR hp



AS AN hp



WORKSTATION

INTERFACING

HP150 APPLICATIONS

MS-DOS FILE

DSN/LINK

MPE FILE

HP3000 APPLICATIONS

IUG-11

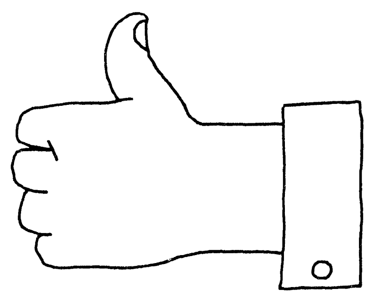
LINK

TRANSFER FROM/TO HOST

COMMAND FILES
AUTO LOGON
(P.A.M. AUTOSTART)

UDC's
+

FILE MANAGER
(MS-DOS FILES)



IUG-14

EASE OF USE FEATURES

==========================

P.A.M. Allows Autostart–
use with LINK

LINK has Auto Logon–
with or w/o PASSWORD
up to 15 Commands
Commands may be &Commands
Load Softkeys
IF/ELSE,ENDIF
DSCOPY
ASSIGN Parms
SEND (w/Parms)
INPUT Parms
WAITCLOCK
and more ... COMMAND FILES too!

IUG–16

Ease of Use Example:

1st – Condor Db on 150
Used the Write cmd
to make an MS–DOS file.

2nd – Autostarted LINK on P.A.M.

3rd – Set up the following Autologon:

&SEND (to get :prompt)
HELLO DSNLINK,DALE.FOLKINS
&KEY 1,"CUSTIN XFER",CUST.XFR
(labels softkey 1 to
start local CMD file
CUST.XFR)

(this CMD file was made in EDIT/3000, then transferred via LINK)

IUG–17

Ease of Use Example:

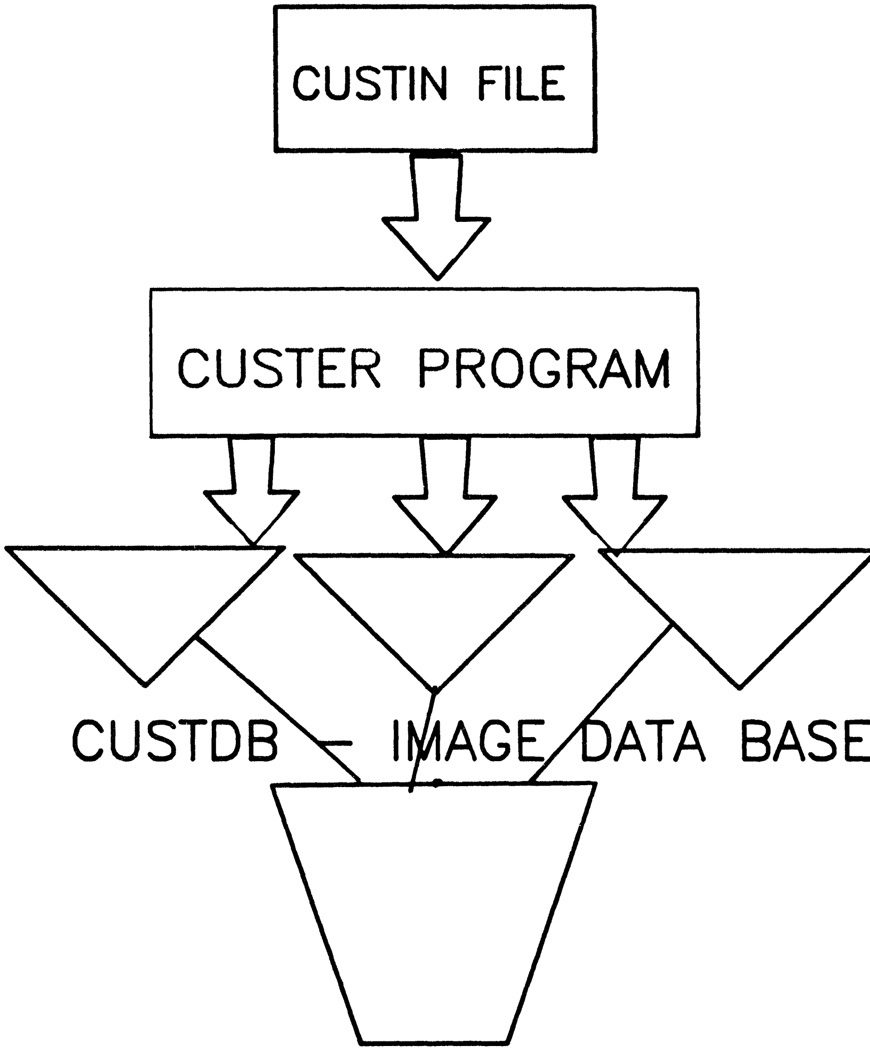
The complete CMD file –

```
&INPUT "NAME OF 3000 FILE?";&P1
&SEND "PURGE ",&P1
&SEND "BUILD ",P1,";REC=-174,1,F,ASCII;DISC=100"
&DSCOPY CUST.DLF TO &P1,REMOTE
&SEND "TRANSACT"
&SEND "CUSTER"
&SEND "E"
&TERMINATOR OFF
BYE
&TERMINATOR Q'
```

NOTE:

- Parms
  - into Subsystem
    - DSCOPY
    - Auto BYE

IUG-18



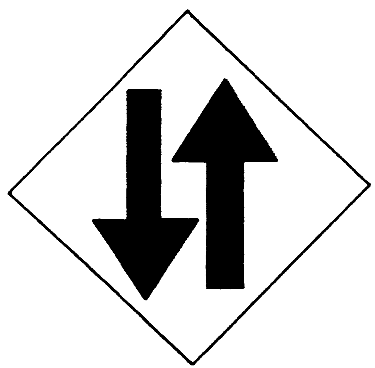
IUG-18A

HP150 to HP3000 FILE COMPATABILITY

150 APPL	INTERFACE FILE	3000 APPL	NOTES
WORDSTAR	EDITFILE	HPWORD	* for ptr cntl
"	"	HPSLATE	
MEMOMAKER	"	"	look for extra CRs
CONDOR	MPE FILE	IMAGE	use even byte fields – saves time
VISICALC	<DIRECT>	VISICALC	
"	DIF FILE	"	
MS–DOS FILES	MPE FILES	File Storage	if prog file, use file= code=prog & rec= –128,,F Binary
PROGRAMS	MPE FILES	PROGRAMS	use Binary in "To Host" Menu
DATA	"	DATA	3000 to 3000 xfer by way of 150

HP3000 to HP150 FILE COMPATABILITY

3000 APPL	INTERFACE FILE	150 APPL	NOTES
*** REVERSE OF HP150 to HP3000 APPLICATIONS ***			
GRAPHICS	SDTODIF	GRAPHICS	SDTODIF prog uses SD file on 3000
"	"	VISICALC	and DIF file on 150



IUG-21

To one accustomed to the genteel world of the HP3000 and MPE, the microcomputer world is, in a word, chaotic. Like the carnival fun house, it is fun, scary, even educational. But the fun house is no place to hold a board meeting. Likewise, the microcomputer does not seem to be the place to do true business data processing.

The micros, however, are already an important part of corporate DP. This, despite the fact that the DP department has done little to promote them. They are important to corporate DP because the most frequent users of micros are managers. Managers acquire micros for reasons ranging from a not so laudable quest for status, to the undeniable need to get some work done.

Frequently, managers are able to get results with a micro through their own efforts, under their own control, on their own schedule. Most large businesses have had a manager who, after failing to have his data processing request satisfied by the DP department, concocted his own solution with VisiCalc or 1-2-3. Usually he winds up with a more attractive, presentable result than he could have gotten from DP anyway.

DP professionals view the invasion of micros with dismay and a sense of foreboding. Manager-users, flushed with a justifiable sense of accomplishment, view the introduction of micros as a liberation. Neither group really knows how to incorporate the micros into a complete corporate information systems strategy. Thus the two groups, data processing and managers, drift apart; each implementing a part of the total business solution, each imcompatible with the other.

DP managers do not have a strategy for micros because they know very little about them. The DP budget does not include either time or money for investigating micros. The traditional demands on DP resources have not abated despite the introduction of micros to do some of the managers' work.

The DataSoft strategy is to integrate both mini and microcomputer development using the existing HP3000 expert DP development team. To implement this strategy requires that DataSoft make the micro look very much like an HP 3000. At the same time, compatible tools must be provided for the users. The user still needs his independence.

DataSoft guarantees the user's independence in such a way that his work can be integrated with corporate-wide development of mini-micro applications. In fact, compatibility, both between the HP 3000 and the micro and between the expert and the manager, is the key to DataSoft's strategy. To implement this strategy, DataSoft has developed microcomputer products for the end user and the DP professional.

The central part of DataSoft's strategy is MIRAGE, a micro database that looks like IMAGE. That means that the creation of the database and the programmatic access to the database are the same as for IMAGE. In fact, the programmatic interface is identical to IMAGE; intrinsic names are the same; parameters are the same; return conditions are the same; even the bugs are the same. In addition, MIRAGE includes many of the ADAGER functions, such as PATHADD, MASTCAP, and SETMOVE.

MIRAGE is for data processing and data management professionals. All of the tools necessary for the data manager to create or restructure a database are found in MIRAGE. Through the external interface intrinsics, the data processing professional can access the database programmatically.

DataSoft also gives the manager-user the ability to create a MIRAGE database using MIRAGER. MIRAGER uses a tutorial Question and

Answer technique to help the end user design and create a data base to meet his immediate needs. MIRAGER then creates formatted data entry screens that give the user complete access to his database. At the same time, it creates a screen for extracting data from the database into DIF files, for later use with spreadsheet or graphics programs.

Together, these products present the two aspects of compatibility that are at the heart of the DataSoft mini-micro strategy. First, MIRAGE on the micro is compatible with IMAGE on the HP 3000. Experts who are knowledgeable with IMAGE can apply their expertise immediately to MIRAGE, even though they have moved to a radically different machine and operating system environment.

Second, the manager using MIRAGER creates a MIRAGE database. He is drafting an application or specifications for an application that can be edited and expanded by the DP expert. The manager's work is valuable not only for its immediate usefulness to him, but also as the first step in a more widely applicable system for others. Furthermore, the data that he gathers is in a form that is readily accessible and transferrable.

Other DataSoft products reflect one or both aspects of the strategy of compatibility. For the HP 3000 experts, DataSoft distributes Pascal and FORTRAN languages that interface to MIRAGE. The Pascal is a complete ISO standard implementation with extensions. It has been certified by the British National Standards Institute, currently the only Pascal for micros that is certified. The FORTRAN is FORTRAN 66 with many FORTRAN 77 extensions. Both compilers are closely compatible with their counterparts on the HP 3000.

MISTRAL is VPLUS for the micro. Forms definition is very similar to FORMSPEC. The programmatic interface uses the same intrinsics as for VPLUS. And the ENTRY capabilities of VPLUS are duplicated with the MISTRY utility of MISTRAL.

DSMIRAGE is DataSoft's most advanced tool for experts. As its name implies, DSMIRAGE gives a program running on the micro distributed systems access to databases on remote HP 3000s. It consists of two parts, a request server that runs on the HP 3000, and a set of intrinsics that are linked into the application program on the micro. Thus, access to remote databases via DSMIRAGE is very similar to the HP 3000-to-HP 3000 remote access using DS/3000.

In the area shared by experts and end users, DataSoft offers several products. REVEAL is a non-procedural specifications language for more complex interactive transactions against a MIRAGE database. Where MIRAGER creates add/show/change/delete transactions for single data sets automatically, REVEAL will let either a programmer or end user write transactions that require information from, and/or make additions or updates to, many data sets simultaneously.

MENTOR is QUERY and a little bit more. With the exception of some of the abbreviated command formats, MENTOR has all of the functions of QUERY on the HP 3000. It even executes XEQ files. In addition, MENTOR will save SELECT files and report from multiple data sets. MENTOR is for reporting, data entry and ad hoc inquiry against MIRAGE data bases. End users can be instructed in the use of MENTOR just as they use QUERY on the HP 3000.

DataSoft products implement the mini-micro strategy of compatibility using methods familiar to the HP user. In conjunction with the HP 3000, they represent an advanced, and perhaps unique strategy for fully integrated mini-micro software development.

DATA WHEN YOU NEED IT

Spreadsheets. Databases. Distributed systems. Remote processing. Microcomputers and mainframes have a lot to talk about. What they have to say varies in volume and in the need for immediacy. The volume requirements range from very large, such as moving whole files or databases, to very small, such as the need to verify that a single data item is valid. The time requirements vary similarly from days or weeks to "right now."

In the environment defined by the HP 3000, tools already exist to satisfy some of these needs. First, there are products like DSN/LINK or PC2622 that transfer files back and forth between the HP 3000 and the HP 150 or the IBM PC. Such products facilitate off-line development of source programs or other documents. Ultimately the files wind up being used on one system or the other almost exclusively.

Another type of product moves data for one specific application, such as accounting or spreadsheets. These products are already common in the IBM mainframe environment. Usually the products are offered by the application solely for the purpose of making data available to versions of the application that run on both the mainframe and the micro.

Finally, there are products that both transfer and convert data for a specific use. Examples of this type of product are programs that move data between data files on the mainframe, and a spreadsheet on the micro or vice versa.

All of these tools work at terminal speeds (2400 to 19,200 bps) so volumes are limited to tens of thousands of characters to a few hundred thousand characters. The effective speed may be increased by as much as a factor of three if the communication program uses data compaction. In order to improve the accuracy of transferred data over asynchronous lines, the program may also use a synchronous-like protocol such as the CCITT protocol.

There are currently no good methods for transferring very large volumes of data between micros and mainframes. This is not entirely unreasonable since micros are seldom used for applications that require or generate megabytes of data.

DSMIRAGE

DataSoft's DSMIRAGE is a general purpose tool for low volume, real-time data transfer between IMAGE databases on the HP 3000 and user-written programs on the micro. As its name implies, DSMIRAGE works much like DS on the HP 3000.

With DSMIRAGE, a program running on the micro can read from and write to remote databases on an HP 3000. Access to the remote database is made possible by a request server running on the HP 3000 and a set of 'DS' intrinsics. The request server is activated by logging onto a port on the HP 3000 and running a program. The micro can then run in local mode, completely self-contained, except for possible DS calls.

With file equations set up correctly, the DSMIRAGE-generated remote request may be passed down to an HP 3000 DS network.

All of the 'DS' intrinsics in DSMIRAGE behave like their 'DB' intrinsic counterparts in IMAGE on the HP 3000, or MIRAGE on the micro. A user program running on a remote micro can open and close an HP 3000 database, and get, put, delete, and update entries in the database.

Access to single records in real-time is most valuable when the host database can be partitioned so that a single user can work almost exclusively in his own partition. For example, if an order entry operator gets a request for a part that he doesn't usually handle, the program would look similar to this:

```
DBGET(localbase,status,mode,set,buffer,argument) IF COND-CODE <> 0
THEN DSGET(remotebase,status,mode,set,buffer,argument)...
```

This record-level remote access is one of the fundamental capabilites required for a complete, coherent mini-micro software development strategy. With this capability, an application can spend more than 95 percent of its time running locally on the micro, while still being able to finish each transaction based on complete information from the host database. Using this capability in an application increases the productivity of the HP 3000 host computer. It introduces the option of programming a network application using the HP 3000 as the central node. And it turns a micro into a request-only node on an HP 3000 DS network.

DSMIRAGE is the common language that lets the HP 3000 and micros talk about almost anything, right now, while its' still important.

THE PORTABLE AND THE SALES REP'S OFFICE OF TOMORROW

> DataSoft International Inc. is looking for a senior sales representative with good HP3000/HP150 background to sell MIRAGE products to major corporations in North America. Based at home, and using a portable to write letters, calculate sales forecasts and projections, store leads in micro database, send or receive US and overseas telexes, send mail to be printed on our main office LaserJet in Austin, or access HP3000 files, you will travel and visit customers all over the country. Moreover, using MIRAGE on a portable will give you practical experience in using the product and support your selling effort.

No, you aren't reading another newspaper. This is a copy of an ad we will publish in a few weeks, and it may very well be a model for future ads.

Having a sales representative on the road is not a new idea. At the beginning of civilization, vendors hopped from one city to another to sell liquors, drugs, or garments. In the past few years, they have travelled all over the world to sell or promote all types of products. Yet communication and follow-up can be complicated. Some hotels and airports offer traditional facilities to businessmen such as telephones, telex equipment, and typing services.

These traditional facilities, however, are often limited. It is unreasonable to expect a hotel secretary to type your ten-page contract at eleven o'clock at night—no matter how urgent. In different time zones it can be especially inconvenient to reach the office and collect your mail and messages. You may have an answering system, use a telephone answering service, or have a beeper in some cities, but you have no way to get your mail or lengthy messages.

The need for field support is why most sales forces are grouped by regions, cities, or districts and share a sales office adequately staffed with secretaries, phones, copiers, computers, telex equipment and coffee machines. One secretary typically supports several sales representatives. One consequence of such a strategy is the size of the economic base required to support the system. You need many representatives and lots of customers for such an office to make good economic sense. And not making good economic sense may eventually squeeze you out of the marketplace.

Today's technology offers better alternatives both in terms of growth path and productivity. A recent technological breakthrough is the portable computer with the HP110 being the first truly professional system in the marketplace.

But just what is a portable computer? In short, it is a computer that can be carried with one hand and used anywhere. Transportable computers, like COMPAQ, represent the first attempt at making it possible for people to conveniently move their systems to different locations. Unless you are preparing for the next Olympics, however, you will find it difficult to take your transportable to and from home and the office every day. Moreover, it is not possible to operate many transportables without a main electricity supply, making you a slave to electrical outlets everywhere you go. You can rule out working on a plane, in the airport, or at your customer's office.

The first portable to be freed of the need for Olympian muscles and a direct electrical supply was the TRS-100 from Tandy. It was perfect for mail, and access to a remote computer. You could have a basic program to calculate sales. However, there was no disc storage, no database possibility, and no spreadsheet. I used it six months. As I became more proficient at using my portable, the limitations described above seemed insurmountable. Then, the HP110 portable came along. It has a word processing program (MEMOMAKER) so simple that even managers can use it! There is LOTUS 1-2-3 as well, giving the system much-needed spreadsheet and graphics programs. You can also use it as a database. If you don't know what a true database is, you cannot be disappointed. You will find a terminal emulator program which enables you to send or receive telexes trough Easylink, directly from the PAM menu, or to log on to an HP3000, or to The Source. Moreover you can use MIRAGE database without having an external disc drive. Voila ... your portable office!

One of Bob Green's article on office problems (The Chronicle, October 1984) mentions a few problems you can solve in a step-by-step way. I shall mention a few more answers.

PROBLEM	SOLUTION
Too many letters to be typed.	Use MEMOMAKER.
Month-end problems, need check-list; lost critical notes on scraps of paper.	Use MEMOMAKER.
Sometimes I need to find a phone number by the individual's name and sometimes by the company.	Do both by using the MIRAGE database.
Telephone tag, too many interruptions; in basket overflowing with memos; out of touch when on the road; no time to create and brainstorm.	Electronic mail using terminal with Easylink or equivalent type of mailbox.
Forget meetings.	Use the alarm system of the HP110.
Forget to do follow-up.	Use MEMOMAKER to type a letter. Send it to the LaserJet using either the HP3000, or Easylink or a direct connection. This takes advantage of the HP110 internal modem.
You have your own ideas on sales trends.	Support your ideas using LOTUS 1-2-3 to extract your MIRAGE database where you have actual records stored.
Too many numbers, no answers.	Use LOTUS-1-2-3 graphics from the same MIRAGE file.
Buried in computer reports.	Do away with many printed reports by having instant on-

screen access.

Because the average user, manager, or sales representative cannot type his own letters perfectly, a draft can be prepared on the portable then sent to the HP3000 at the main office. Once there, it is collected by the main office secretary who will edit it before printing on the LaserJet. Providing user-oriented database creation programs and database enquiries programs like MIRAGER or OLYMPIC gives managers and sales representatives access to their own data definition and control. Moreover, since we are creating an IMAGE compatible environment, assistance from their EDP team is available when necessary. Simply stated, they talk the same language.

Using an HP110 and Easylink, our sales representative will share the secretarial facilities with our headquarters staff. We have a LaserJet to ensure that our correspondence looks perfect. We need only one secretary for the whole team because she edits or polishes letters rather than typing them from scratch, and she does not send or receive telexes for individuals on the staff. The only pitfall for our sales representative on the road is that he has no access to the coffee machine...

Setting up the HP110 to send telexes automatically from PAM is simple. The HP110 PAM (Personal Application Manager), unlike its counterpart in the HP150, is extremely easy to set up. Call MEMOMAKER and type two lines for each new program you want to access directly from PAM. The first line is the program title as it will appear in the window. The second line is either a program, a batch file to be executed, or an MS-DOS command. To access Easylink the second line should look like : TERMINAL /T easylink. Actually "easylink" is the name of a file that we will now create. You can use any name. The first line can be something like : EASYLINK TELEX. Store your memo into a file call PAM.MNU. If you already have a PAM file, append or insert these two lines.

Once your PAM file is ready, you need to create the file called "easylink". From PAM you call terminal. Terminal prompts you for a Configuration File Name. Since you are about to create it, press RETURN to escape that question. The screen should clear and a set of new function key labels will appear at the bottom of the screen. Press F5 to set up a terminal configuration. The cursor is now on "comment". You may enter any free text, or skip it. Using the TAB key go to the "logon String" prompt. You have the choice of entering a string there or referencing a file containing the string. For Easylink, it is easier to reference a file. Enter easylink.log . Instead of "easylink.log" you can enter any valid file name. The file name must be preceeded by " " and followed by " ". Press TAB to go to the "Phone Number" prompt. Enter the easylink number. It may be a local number or an 800 number. If you have a switchboard you need to indicate the figure to get access to the telephone network. Set "device" on Modem, Handshake on "None," Parity "even," Baud Rate" on 300, Xmit Pace on "None," Recv Pace on "None," Data Bits on 7, EOL Sequence on "CR," Parity check on "On," and Echo "Off." Press F2 to store your configuration. At the question "file name" enter "easylink" or whatever name you have used in the PAM.MNU file. Then exit the configuration with F8. Press F8 again to exit Terminal and get back to PAM. Once again, call MEMOMAKER. Now, we set up the EASYLINK.LOG file.

The Logon string enables the HP110 to logon the easylink computer without any manual intervention. When the connection is established with Easylink, you enter the terminal type, your identification number and your password. We will do all that automatically. The easylink.log file needs only one line with : ID? WO"01 EID999999 YOUR.PASSWORD"E PTS "/scan 10"E. Let's first explain the syntax. The XX indicates that the HP110 will wait XX in order to continue. Since EasyLink is supposed to send the prompt ID?, we better wait for it. The WO indicates that after the receiving of "ID" we will wait 10 seconds. The 0 means 10, 1 means 1, 2 means 2, etc. You may try several possibilities. Waiting 10 seconds before sending the ID answer is more than safe. The "01 EID999999 YOUR.PASSWORD" is the ID answer. The 01 indicates the terminal type. The HP110 is 01 for Easylink. EID is a mandatory prefix for Easylink. The 999999 is your ID number and YOUR.PASSWORD is whatever is designated for you. Both ID and password are provided by Western Union in the contract you have signed to access Easylink. The E before PTS indicates that we will enter an End Of Line to transmit the ID and Password to Easylink. PTS means that we will wait for the PTS message from Easylink before processing further. PTS means Proceed To Select in Easylink jargon.

In order to automatically list the contents of our mailbox, we have placed"/scan 10"E at the end. The 10 indicates to Easylink that we want a pause after each 10 lines. Save your one line file as easylink.log or with the name you gave when you prepared the configuration file. Get out of MEMOMAKER. If you start the "EASYLINK" application which should now appear in one of the windows before MEMOMAKER you will directly logon Easylink. One more detail. The HP110 must be connected to a telephone jack with a regular extension. Without it, it won't work.

I have used this system, not only from the office or my home, but also from several hotels from East to West coasts. Some hotels have strange switchboards which cannot accept the HP110 call. However, if you buy a telephone extension from RADIOSHACK with one input for the wall, and two outputs (phone and HP110) you can use the phone to dial the switchboard and the HP110 to dial the Easylink number. I recommend the Marriot Hotels, and some Best Westerns since each room has regular jacks. You can also use the closest HP sales office or an HP dealer store as an office. They are generally very cooperative, and sometimes you may even have access to the coffee machine!

Our sales representative will not have an office. However, he will work from the most convenient place for him and he will use the facilities of one of the DataSoft Support Offices, either in Austin (Texas), London (England), Paris (France), and very soon, Toronto (Canada) to print and mail letters. There is no reason why London cannot type an urgent RFP answer overnight, and send it to the Austin LaserJet when the Texans wake up. No reason at all. The non-office is an economic reality. Productivity and cost effectivness are the two driving factors why DataSoft sales representatives will not have a fixed desk. What about YOUR organization's sales representatives?

This text was typed using Wordstar on a COMPAQ (The HP150 was not available). The text was read by MEMOMAKER on the HP110 using the COMPAQ drives as regular HP110 drives (HP sells a board you can plug in your IBM/PC or compatible) MEMOMAKER got rid of all Wordstar special characters. Then I called directly from PAM the UPLOAD program to load the text from the COMPAQ disc, thru the HP110 into an HP3000. HP3000 needs to run EDITOR. The upload configuration must dictate the FROM file, and the HPPROMPT protocol. After direct editing on the HP3000 by my secretary, the text was sent to this publication which somehow or another got it printed in their magazine.

The non-office. It works!

Michel Kohon
DataSoft International, Inc.

TO RELATE OR NOT TO RELATE...THAT IS THE QUESTION.

THE SOURCE...

The best source of inspiration in the HP3000 world is our international guru, Alfredo Rego. Each time I listen to him, speak with him, or read something by him, I get carried away. The first time was in 1978. When reading the Denver International users group meeting proceedings from somewhere in Holland, I discovered ADAGER. Can you believe I wrote to him in Guatemala! However, this time the source of great inspiration is the article published in The Chronicle (October 1984) called "Reflections." "Reflections" on Image, pure Alfredo style.

Somewhere in this great piece of 'bon sens' Alfredo talks about relational databases in the very way I would like to talk about it if my English were not so miserable. Let me remind you of the text since you might not have The Chronicle on your desk.

"A casual cruise through literature of the times might leave us thinking that only the so-called "relational" systems can relate. Nothing could be more unrelated to the truth. Relating is the whole idea behind any DBMS."...and few lines after..."We may relate when we enter the information, when we report the information, or any time between. We may use techniques such as hashing and indexing to go (more or less directly) to the locations of parcels containing key pieces of information."

I will stop citing, although it would be worth reprinting the whole article and, like a gourmet, enjoy it word by word, or letter by letter, depending on your taste.

NO TOLL FREE...

The fact is that IMAGE as a network relates as does all the DBMS in the world. It depends on where and when you want to pay the price. Since IMAGE and MIRAGE maintain a set of pointers up-to-date at the writing stage (DBPUT or DBDELETE) the price is paid at entrance. If you are lucky enough to have access to your data sets with a known and complete key, the price of obtaining the information is minimum, and the response time (the measurement of the price) is good. (For the definition of a good response time, refer to Bob Green's article in the same Chronicle issue.) If you want to access your database with keywords, you may use IMSAM as a complement of IMAGE. Most existing relational databases on both minis and micros do not charge at entrance. You pay at enquiry time. When you want to obtain some data from your third or fouth generation language, the relational system will start indexing or merging data sets in order to later select the requested entries. You may even end up with an intermediary new file which can be saved, or with a virtual file which contains addresses of records, like a QUERY select-file. Relational databases are not like the Boston Tunnel, or the Golden Gate Bridge in San Francisco where you pay when you enter town, not when you leave. But you do pay somewhere and somehow.

However, if RELATIONAL is now the best marketable word in the EDP industry, there are some good reasons. There are also some good reasons to believe that IMAGE or MIRAGE, or any network database structure is potentially a relational database. We will now examine all these good reasons, and if you are still with us in few minutes, learn how we can one day use MIRAGE in either a network or a relational way.

NO NEED TO SHOP AROUND...

Relational databases have existed for a long time. One good example is RELATE on the HP3000. However they have not been very popular for several reasons. First of all, in the HP3000 world, IMAGE is so easy to use, so reliable, so flexible with ADAGER, so easy to access with QUERY, ASK3000, RAPID, etc., that there was no need to shop around. Only a few too smart users were rapidly stuck with IMAGE's limitations in very specific situations, but most of us were happy. And when you are happy, you enjoy yourself. So we did. Another reason is that the word was spread that relational data bases were slow. It again depends on what you call slow, and where you measure it (ENTRANCE or EXIT). The last reason that I am aware of is the difficulty to set up a correct database structure--that is to say a third normal form one.

There we go. You start wondering what a third normal form structure is, and moreover you think that if this paper is going to be theoretical, you won't continue reading it. First of all, let me tell you that if you think I am a theoretical man, you are wrong. Just read my Step-By-Step article and you will find out how practical I am. Secondly, I have listed in the appendix a number of references of excellent papers about normalization, and third I will explain in one sentence what a third normal form is.

A third normal form database consists of masters where all fields of one individual data set refers to the key, only to the key, and to nothing other than the key. Purists will add many comments to that. Let's not listen to them.

LET'S BE ESOTERIC...

You start to realize that an IMAGE or MIRAGE database can be organized in a third normal form and therefore can be used as a relational database. You also start to appreciate the difference between the logical structure of a database, and the way you want to access it. A database can have a network structure, hence you may want to access it in a relational way with relational operators. Relational operators? Another esoteric term. I have seen so many poor explanations of those that I am really ready to give one more. It can only be better.

The relational database concept was created by Codd from both experiences and theories. Codd was able to explain by a number of mathematical formulae the relational concept. As in any mathematical environment, he needed operators to play around with those concepts. The three basic ones he defined are SELECT, PROJECT and JOIN. The mathematical background certainly did not add anything towards the understanding and explanation of relational databases. I will not try to explain it now. Later in the paper, you will understand what it is.

EVERY ONE IS BUYING THEM...

As we said, relational databases were not popular on minis like the HP3000. However they are on micros. Data bases like DBASE II or CONDOR call themselves relational and everyone is buying them for obvious reasons. First, a relational database is easy to build when you create a set of MASTERS. These databases have no mechanisms to enforce any third normal form structure and therefore the response time at inquiry time can be very, very long. However, most of the user community accesses only one data set at a time, and with a specific key, so the response time is good. But you don't need a relational database to do that. Any network structure will do the same. Relational databases are popular on micros because people use them as a network. Second, they generally have a fourth generation language attached to them which means that users can easily draft an application. The problem is that when it becomes too complex, or when response time is too poor, there is no language interface for any fine tuning. And this introduces another misconception about relational databases. The fact that you must have a fourth generation language to access them is not true at all.

What is true is that relational databases offer mechanisms to make retrieval easy whatever the data you want to access. Because the database knows its own structure, it is able to navigate automatically. It avoids programming lengthy and untested series of DBFINDs and DBGETs in a user's program. So complex, sometimes with IMAGE (which unlike MIRAGE is only a two level network), that products like RAPID, or POWERHOUSE do it for you. But it does not mean that the code (the MOVE, IF, SET, etc.) which is in these fourth generation languages needs to be there. It can be in some DBREAD which eventually will decide for you which DBFINDs and DBGETs to execute.

1 + 0 = ... 2 !

Both relational databases and network databases have their advantages. What is good about a network is that you can retrieve data quickly when you know the key, and especially when you have only master data sets. What is good about relational databases is that the retrieval of data is handled by the database itself, not by any Query language, or by your program. We can now take the best of the two worlds and rather than taking sides, we may add possibilities.

What do we need to access a network in a relational way, with only masters or mixed data sets for MIRAGE? We need to indicate what we want to select. We may want to have a DBSELECT which will indicate for a specific data set the criteria of selection like "ORDER VALUE >=100000" or "NOT CUSTOMER = JOHN".

We need a DBREAD which will accept a multi-set list. The DBREAD will go in the different data sets to retrieve the items of the list. The active DBSELECTs will help DBREAD to find the most appropriate route between the different data sets. The DBREAD will JOIN several parts from several data sets into one buffer.

A DBREAD on one master with only a selection with the key value is a DBGET mode 7 in the IMAGE/MIRAGE world. We are only expanding concepts, not changing them.

We also need a DBWRITE with the same multi-set possibility. It will decide what to PUT and what to UPDATE from the very same DBSELECTs. It will PROJECT the buffer into several data sets like an exploding bomb projects its shell.

By now, you know what the three basic relational operators, SELECT, PROJECT and JOIN, are. As you can see, you don't need them to program. The relational mechanisms of the database will use them for you. These operators are transparent throughout the use of DBSELECT, DBREAD and DBWRITE. Your programs can access these intrinsics directly like fourth generation languages will be able to access them.

PRACTICING...

Let's imagine that we have created a MIRAGE database with a third normal form structure. We will study how to build such a data base in a future article. Our example consists of 2 masters and one mixed data set. A mixed data set is a feature of MIRAGE. It is a data set which is both a MASTER for one of its field and a DETAIL for one or several other fields. It enables access either with DBGET mode 7 or DBFIND/DBGET 5/6. This feature implies that MIRAGE is a true multi-level database.

```
        PARTS                           WAREHOUSES

    PART-NO (MASTER KEY)            WARE-NO (MASTER KEY)
    PART-COST                       WARE-LOCATION
    PART-ON-HAND                    WARE-CAPACITY

        PART-WAREHOUSE

    PART-NO
    WARE-NO    ---. PART-NO + WARE-NO  (MASTER KEY)
    WARE-PART-QTY
```

With MIRAGE you can define concatenated keys like this PART-NO + WARE-NO master key. PART-NO and WARE-NO are also, individually, detail keys to make sure that consistant information is recorded in PART-WAREHOUSE.

Our program wants to retrieve all PART-NO and PART-COST for warehouses with WARE-CAPA greater than 1000 but less than 2000.

Written in Pascal, the code will look like the following :

```
PROCEDURE example;
BEGIN
open-database(basename);
setname  := 'WAREHOUSES;';
operator := '>=';
mode     := 1; (*initialise*)
argument-name := 'WARE-CAPA;';
argument-value := 1000;
DBSELECT(basename,setname,mode,status,
         argument-name,operator,argument-value);
argument-value := 2000;
operator := '<=';
mode := 2;          (* AND *)

DBSELECT(basename,setname,mode,status,
         argument-name,operator,argument-value);

(* DBSELECTs will not do any I/O.  They will prepare a table from
which the next DBREAD will compile and execute a navigation route.
*)
mode := 5;    (* use only DBSELECT criteria and set names, not the
DBREAD ones *)
list := 'PART-NO,PART-COST;';
REPEAT
DBREAD(basename, setname,mode,status,buffer,list,argument-value);
IF (status.Cond-code = 0)
    THEN BEGIN
         edit-proc(buffer,buffout);   (*  to prepare buffer for
                                          print*)
         written(buffout);            (* print buffout on screen
                                      *)
         END;
UNTIL (status.cond-code    0);
END;  (*of example*)
```

WHAT THEY ARE COOKING ...

It is DataSoft's plan to introduce a set of intrinsics to access a MIRAGE database in a relational way. However, today's MIRAGE works 'only' like IMAGE, with operators to access basic records. The example shows you how your program will look when using the ECLIPSE relational operators. MIRAGE is already set for it. MIRAGE is a multi-level network (i.e IMAGE is limited to 2 levels). What is a potential problem (complex structure means complex programming to retieve or post data) is a premium for relational operators. MIRAGE accepts concatenated and overlapping keys which avoid data redundancy that would normally be created by the third normal form. MIRAGE works from a set of flat "vanilla" sequential files which means that your data can be accessed either through MIRAGE or through ECLIPSE.

What does it add to your shop to use relational operators in programs. Productivity is the key word.

First, programming will be simplified. You will take better care of the user interface. Both productivity and quality will improve. Second, your programs are not completely tied to the data base structure. Less commitment to a database structure means more flexibility and therefore productivity increases again. Moreover you will not lose your existing programs investment, and your existing databases. This is at least the objective of our future ECLIPSE relational database. Let's hope that Hewlett-Packard will follow that direction. It is now time for all of us to know what they are cooking up there. Isn't it?

Michel Kohon
DataSoft International Inc.

BIBLIOGRAPHY

Codd, E.F "A Relational Model of Data For Large Shared Data Banks". Communications of the ACM. 13,No 6(June 1970) 377-387.

Kent, William. "A Simple Guide to Five Normal Forms in Relational Data Base Theory." Communications of the ACM. 26, No 2(February 1983) 120-125.

Rego,Alfredo. "Reflections on Image" The Chronicle, October 1984.

Kohon,M.D. "Introduction To Step By Step" SuperGroup Association.Vol 3, Issue 3,July/August 1983.

Seltzer,R.L. "An Introduction To Data Base Normalization" .Proceedings HP3000 IUG 1984 Anaheim.

Complex documentation applications with laser-print-output

IMPLEMENT IT JUST BY MOULDING HP-SUPPLIED SUBSYSTEMS FOR INDIVIDUAL NEEDS

by David Padanyi-Gulyas
and Hans Ulrich Benz
Benz & Benz Consultants AG, Zuerich, Switzerland

Summary

Documentation is more and more important for the users of all types of products. It is a new field of applications for the EDP-Departement. There are Standard-Software-Packages of HP available and they give a very useful help for the creation of document applications. The combination of textual and graphical information-representation is a very powerful instrument in the hands of somebod knowing how to handle it.

Introduction

THE PROBLEM

Hearing the word "DOCUMENTATION" you mostly get a bad feeling in your stomach. Everybody knows what should be documented and everbody knows that the documentation is uncomplete, not up to date or just not existing.

There are some reasons for this fact:

In the past documentation was unimportant, you "just know how to do it". Today it's more and more essential because our technical world is going to get more and more complex. In the past you have written the documentation after your product was finished, just before it was thrown on the market. Today you need the documentation more and more even before you begin the developement of the product. In the past you understood by a documentation a pile of papers with endless texts, hard to understand. Today you see pictures, colors, indices and drawings summing up an easily understandable and handy documentation.

No need to say: today, documentation is needed by everybody

DEFINITIONS

A "DOCUMENT" is an ordered collection of texts, drawings and pictures of more than one page (normally 10 - 10'000 pages).

A "DRAWING" is something sketched by hand (or HPDRAW), consisting of lines and textures.

A "PICTURE" is a more complex representation of something that can not be easily drawn by hand, it is normally photographed or scanned.

A "DIAGRAM" is a graphic representation of a statistical row of results, e.g. a pie-chart.

A "LOGO" is a rasterized graphic symbol which is repeatedly used in a printed document.

We want to make a clear difference between text- and documentprocessing. TEXTPROCESSING deals only with texts, it has letter quality as printing standard and normally a size of 1 - 10 pages. Documentprocessing is concerned with the above mentioned elements and has lower print-quality standards than letters.

The normal output device for a document is a (laser-)page-printer. For special purposes you can use a matrix-printer along with a plotter to produce a document.

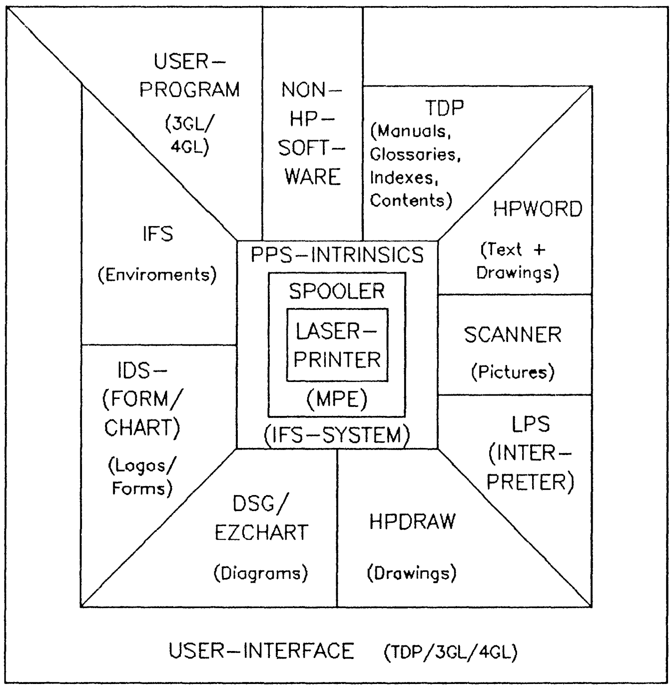
ABBREVIATIONS:

```
DA  --> Documentation Application
HP  --> Hewlett Packard
PP  --> (Laser-)Page-Printer
3GL --> 3d Generation (Programmers) Language:
        COBOL, BASIC, PASCAL, SPL etc.
4GL --> 4th Generation (Programmers) Language:
        TRANSACT, GENASYS, QUICK etc.
PGM --> Program
DB  --> Database
HLL --> High Level Language, 3GL or 4GL
STD --> Standard
```

Background

The technology of our DA environment:

On HP 3000 with some HP-Standart-Software-Packages:



SEMIU02.HB.TEST

You don't need all software-packages to produce a document on a laser printer, you can use a normal programming language and PPS-intrinsics to do it. But it is a hard work to implement it that way! You can have almost the same results in a shorter time by using HP-standard-packages and a "USER INTERFACE".

Although HP tends to sell those packages as "ready to use", we made the experience that the end-user is not happy with them when you do not present him a more user-friendly interface.

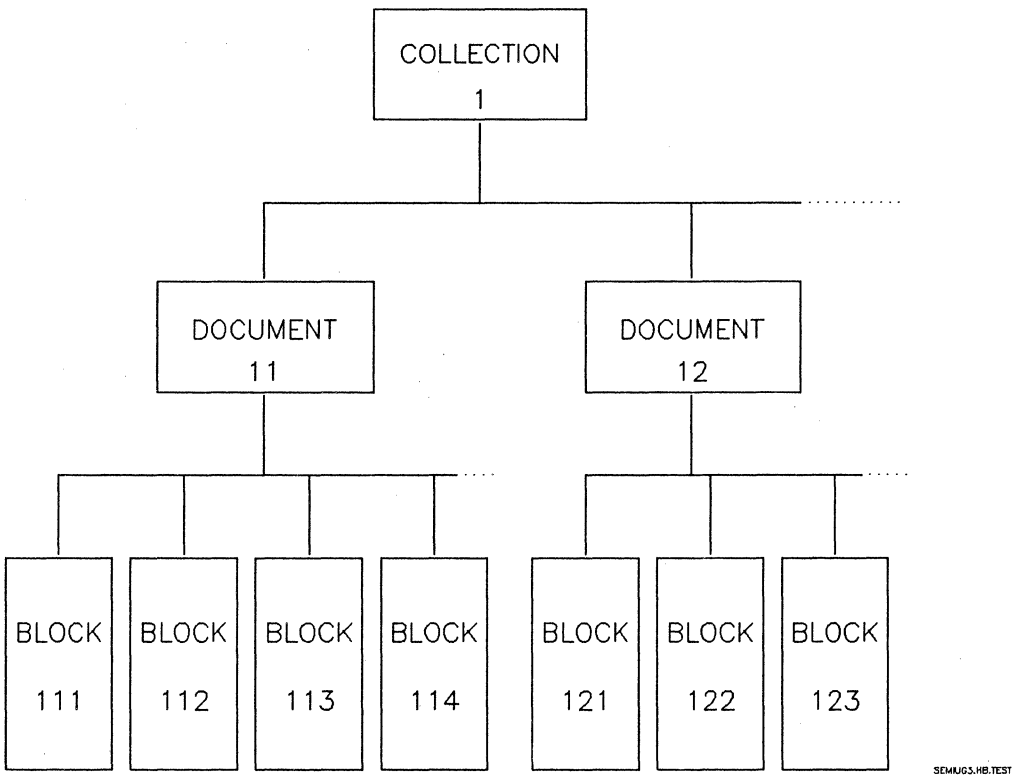
One of the main disadvantages is (talking especially about TDP) that there are too many possibilities in the packages.

Topology of the Database

(BLOCK, DOCUMENT, COLLECTION, DESCRIPTION)

We have analized several different DA's and we found always the same kind of <u>data structure</u>

The Data Structure



The related **DESCRIPTIONAL DATA BASE** with the essentials:

NAME OF THE ELEMENT		TYPE	SHORT DESCRIPTION OF THE CONTENT
COLLECTION	1	REFERENCE	
DOCUMENT	11	REFERENCE	
BLOCK	111	TEXT	
BLOCK	112	DIAGRAM	
BLOCK	113	TEXT	
BLOCK	114	PICTURE	
DOCUMENT	12	REFERENCE	
BLOCK	121	TEXT	
BLOCK	122	TEXT	
BLOCK	123	DRAWING	

SEMIUG4.HB.TEST

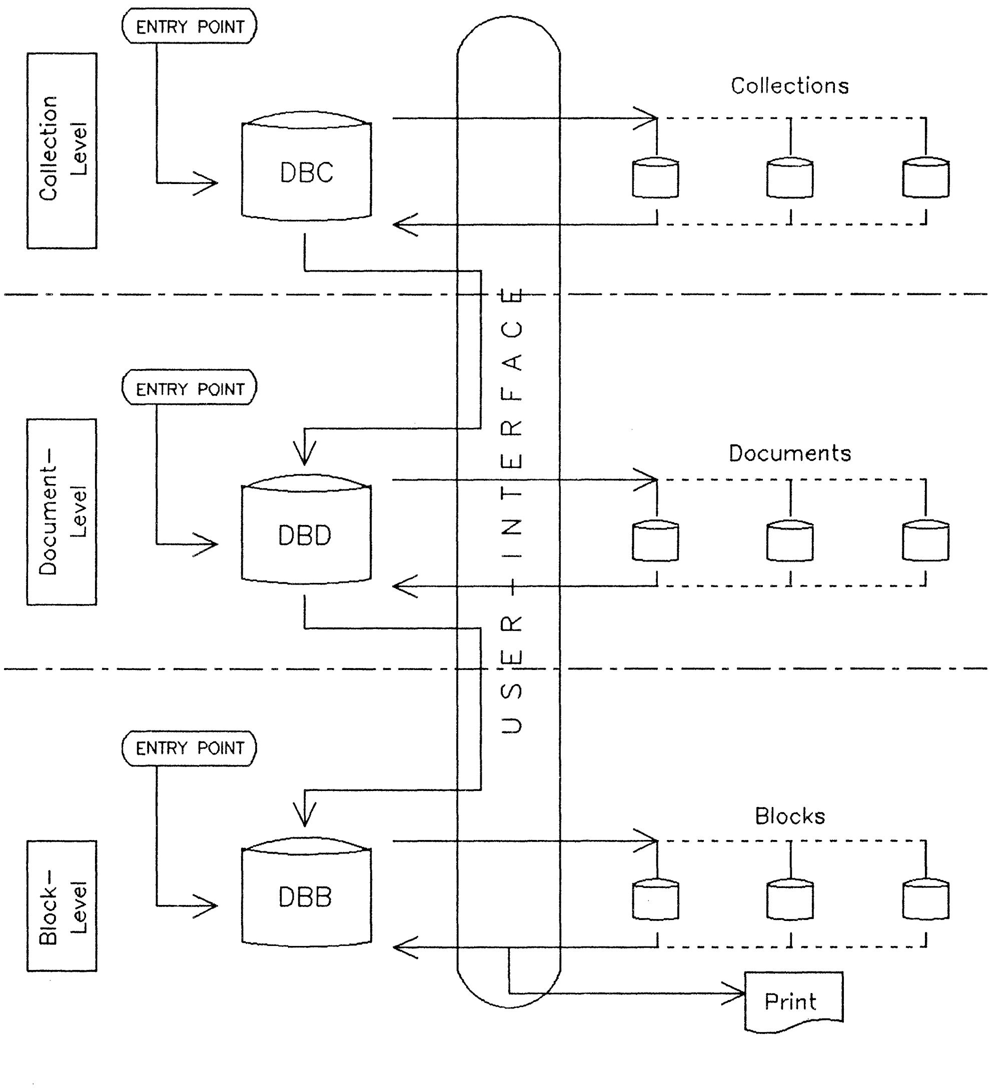
The BLOCK contains your data which will be in your DA. Each block can contain any type of data of a DA: text, picture, drawing etc. But each block may contain only one type of data.

The DOCUMENT contains no senseful data at all but only references to all types of blocks. Each block can be part of several documents.

The COLLECTION is a document that contains no senseful data but only references to several documents. The differnce to a document is that a collection also can have INDICES and CONTENTS.

In the DESCRIBTIONAL DATA BASE you find all.informations about each BLOCK, DOCUMENT or COLLECTION. This database can be on a sheet of paper if you make your maintenance work manually or an IMAGE-data-base if you manage your DA by a computer.

The user's view of the datastructure must be very simple. It must guide the user from all possible ENTRY-POINTS, to the descriptive database (DBC, DBD, DBB) and the final printing-EXIT.



User's view on the User Interface

The USER-INTERFACE must represent the user's needs to create, maintain and print the blocks, documents and collections.

We must support the following steps of creating a collection of data (example):

* The user creates a block of information,
  then

* he prints a preliminary version of the block,
  then

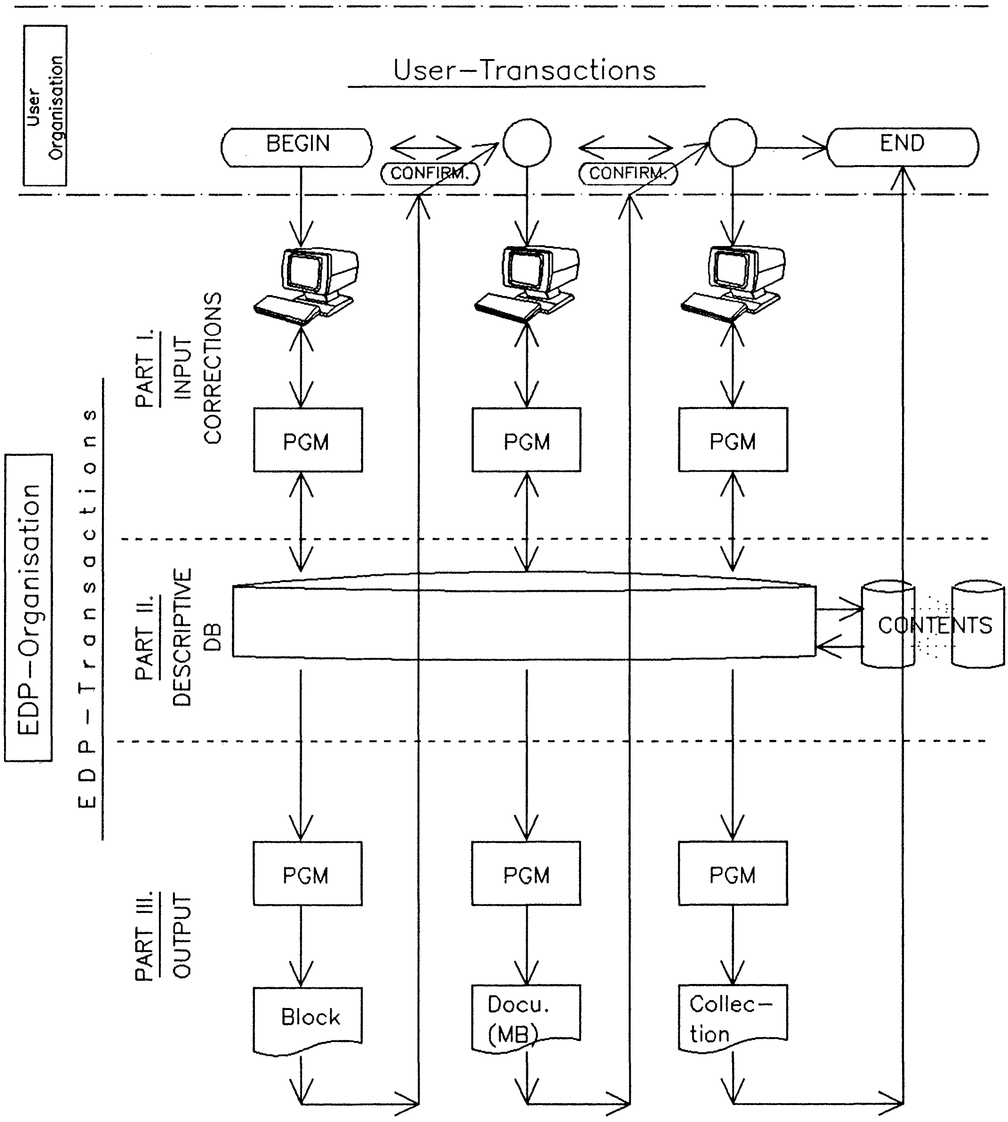
* he corrects the block,
  then

* he reviews the block and confirms it.
  After the confirmation he can use the block in a document but he cannot correct it any longer by normal means, because it's locked.
  Then

* he creates a document containing several blocks and confirms it.
  Then

* he creates a collection of several documents and confirms it.
  Then

* he unlocks or deletes a single block of information: the DA must tell him where the block is used and what effects it can have on several related documents.

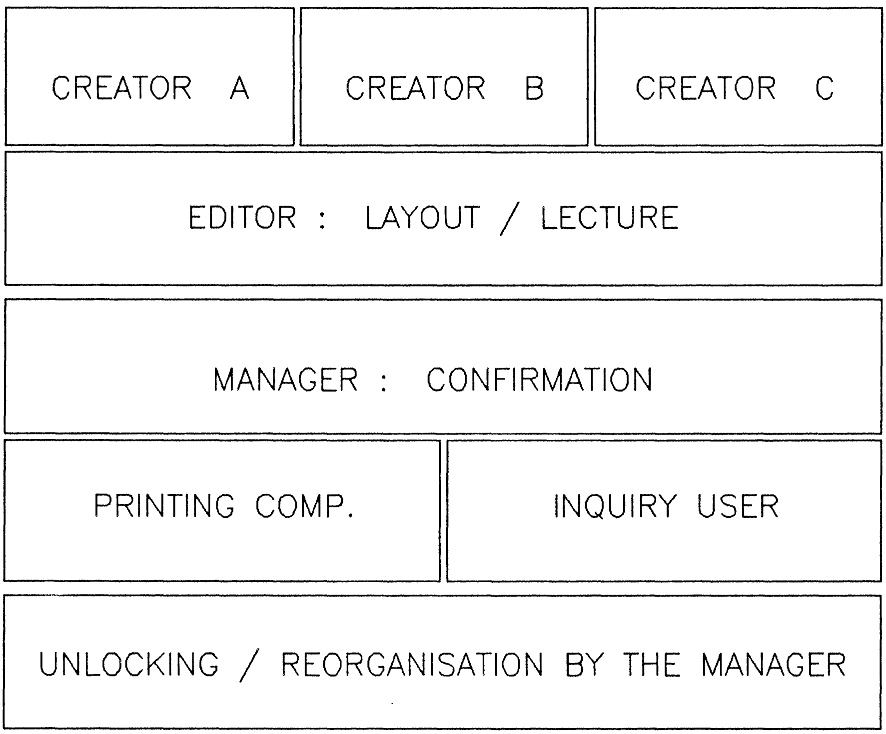


SEMIUG6.HB.TEST

The structure of the User-Organisation

Implementing a DA, you soon will have several categories of users:

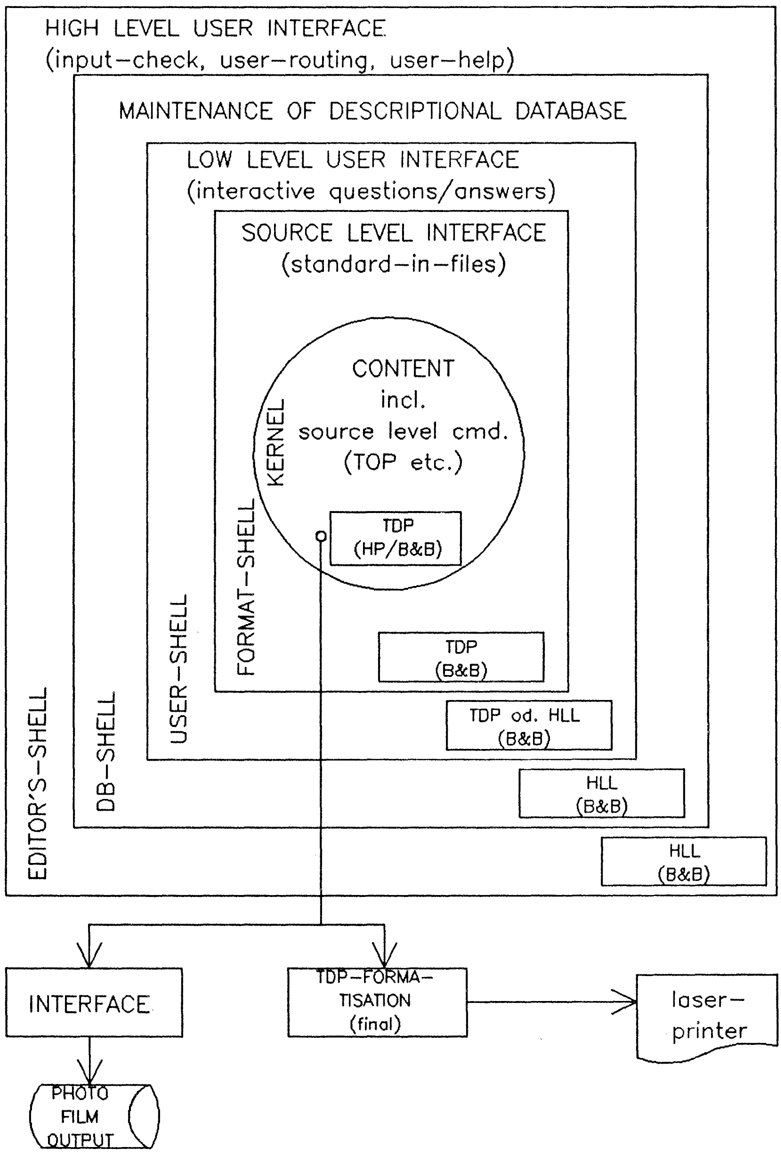
--> responsible DA manager
--> his data capture and maintance departement
--> the printing and publishing people
--> the normal user:
  - wants to see certain documents on his screen
  - wants to create documents for his own use etc.



It is necessary for each created block that it is reviewed and brought to a common layout. This must be a centralised task near to the management of a DA.

Methods of Implementation

SOFTWARE STRUCTURE



SEMIUG8.HB.TEST

You can use several shells to implement a DA. Depending on the know-how of the user and on the comfort he wishes, you can implement it in different ways.

On the KERNEL-level you must know very well the TDP-commands to get a good document. The user is responsible for everything and he must take a notice of the name of every document he produces. This shell means that you use TDP as it is without any help of a supplementary software piece.

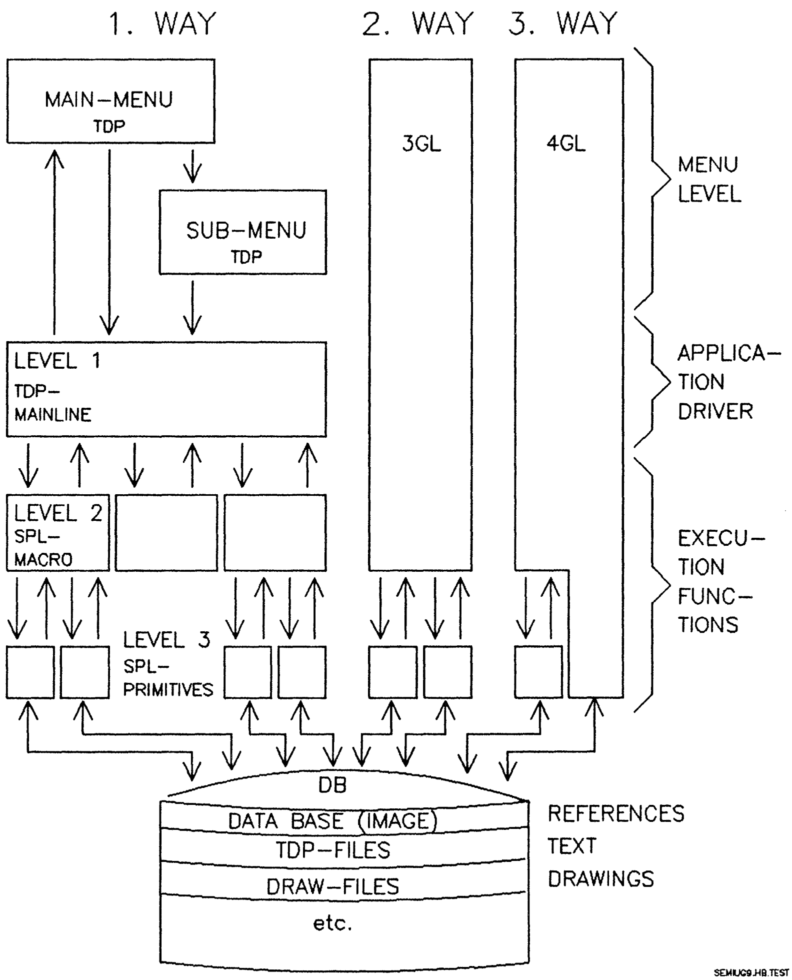
You can create a LOW LEVEL USER INTERFACE if you use standardized IN-files (user-written procedures, written in TDP) and an application driver (user written interactive procedure, written in TDP or some HLL-codes). On this level the user must still note manually what pieces of data and documents he has on the system.

Through the HIGH LEVEL USER INTERFACE you have comfort and functions realized as in a normal interactive application: every input is tested, full menu-procedures are provided and the documentation is available interactively on the screen by means of a help-function. The documentation of each block is automatically maintained by the DA.

It is important for a DA to check which level you want to realize on. It can be a very frustrating experience for a user if he has to begin on a much too low level.

The Choice of Programming Language

There are several ways of doing it:



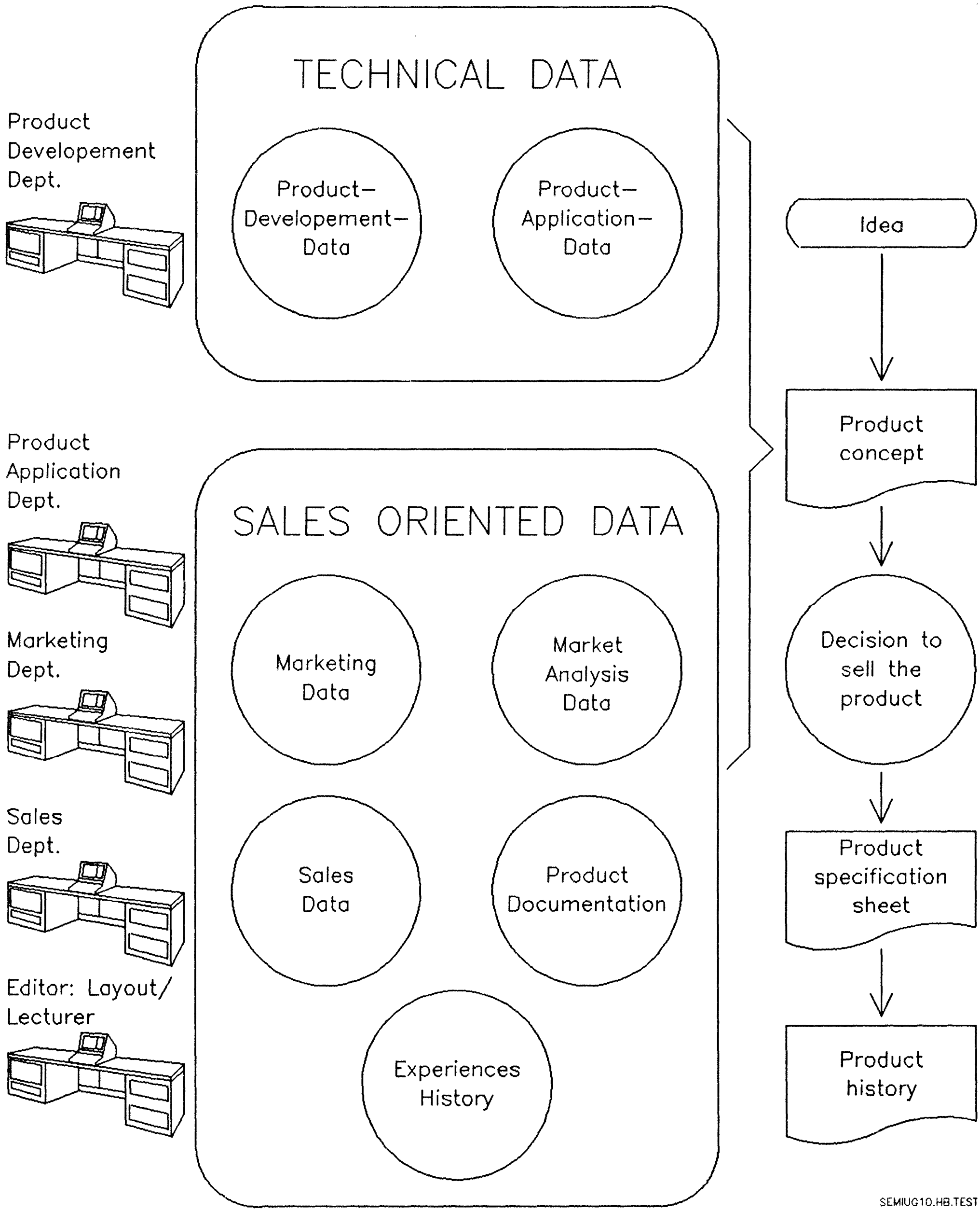
At the time being we have implemented a DA in the first way. If you have a properly structured approach, it is possible to write a good part of the DA in TDP, in a 4GL-like-manner and it will be working! To do so you need experience and knowledge, both is very hard to find at HP. We found that around 30 % of the codes are menu- and application driving instructions. This can be done in a more elegant way in a HLL or 4GL.

Two Examples

PRODUCT DOCUMENTATION

In this example the customer has around 40'000 different products. He wants to describe the type, the application and the use of his products for his customers in PRODUCT SPECIFICATION SHEETS. Each product must have his specification sheet which consits texts, pictures and drawings. Each product group has common texts and a common logo in the head of the product sheet. Per product group the customer wants to have a collection of all related product sheets.

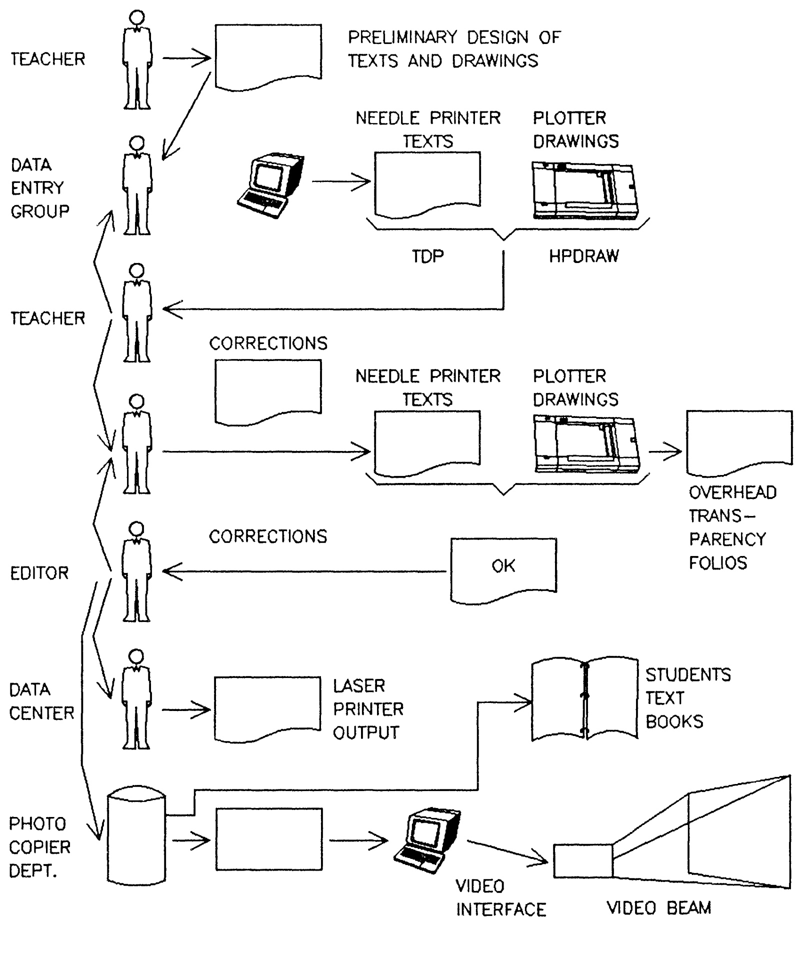
There are 5 departements which collect, enter and view product-data. The final layout and lecture is done by the editor. On the other hand you can have documents on all stages of a product-life-cycle on the system: at the beginning it is only a note of the product idea, then the user develops the product concept. After the decision to sell the product it is really developed and produced. During the same period the user produces the definitive product specification sheet.



TECHNICAL DATA
Product Developement Dept.
Product-Developement-Data
Product-Application-Data
Idea
Product concept
Product Application Dept.
SALES ORIENTED DATA
Marketing Dept.
Marketing Data
Market Analysis Data
Decision to sell the product
Sales Dept.
Sales Data
Product Documentation
Product specification sheet
Editor: Layout/ Lecturer
Experiences History
Product history
SEMIUG10.HB.TEST

SWISS SCHOOL OF BUSINESS INFORMATICS

This is one of the first swiss HP-DA that has been working for 2 years now. In the first stage of the project of that new school, a team of around 10 people have developed and produced the course material with simple use of TDP and HPDRAW. The teacher-team has produced over 2500 pages of text and about 1000 drawings. In the ultimate time period before the first course started, 4 people were working for the data-entry team each at a screen conected to a HP data-center. 3 of them were students hired temporarily for this purpose.



Appendix

TDP USE-file example

This is an example of a TDP-written routine that creates a job for rastering every HPDRAW-FIGURE as preparation run for the laser-output.

```
Q "******************************************************"
Q "* THIS IS A TDP-USE-FILE FOR RASTER-FILE GENERATING  *"
Q "******************************************************"
DQ ALL
NAME RSTJOB
@L0
ZP::="WELCHE GRUPPE  (A od. B od. C) ?..... "
@IF NOT FILE "Z::" THEN GO TO L0
ALQ,!JOB BRASTER,JZ::1/WISS.WISS,KURSZ::;INPRI=3
ALQ,!RUN TDP/BB.PUB.SYS
ZP::="FILE-NAME OF THE TDP-TEXT-FILE ? .... "
ALQ,T Z::
@IF "RASTERING FROM THE BEGINNING (Y/N) .. " THEN GO TO L1
ZP::="START RASTERING AT LINE NUMBER ? .... "
ALQ,DQ FIRST/Z::-1
@L1
@IF "RAST. TO THE END OF THE FILE (Y/N) .. " THEN GO TO L2
ZP::="LAST LINE-NUMBER TO RASTER ? ........ "
ALQ,DQ Z::+1/LAST
@L2
ALQ,ALQ,************ LAST LINE *************
ALQ,SET MISSOK
ALQ,FINDNUMBER FIRST(1)
ALQ,FN "ILLUST",LIT;HQ "ILLUST"/*,APPEND;FNQ *+1
ALQ,@999
ALQ,DQ ALL
ALQ,A
ALQ,\ENVIRONMENT WISSENV.TDP
ALQ,\LPTOP 1;TOP 9;BOTTOM 3;PAGELENGTH 72
ALQ,\HEADLINE 1;PAGENOLINE -2;LFT 4;RHT 84
ZP::="FILE-NAME OF THE FIRST RASTER-FILE ?  "
ALQ,\NAME RASTER Z::
ALQ,//
ALQ,AQ,HOLD
ALQ,CQ "ILLUST" TO "\NEW;ILLUST" IN ALL
ALQ,L ALL
ALQ,FINAL TO *PP
ALQ,EXIT
ALQ,!EOJ
SET YES
K
SET PERMYES=OFF
STREAM RSTJOB
Q "========> END  for stop,  @  for continue"
END
```

INTERNAL

This paper has been produced with a DA that uses the HP-Standard-Software products TDP, HPDRAW, IFS, IDS and LPS. It was printed on a HP 2680 Page-Printer.

BIOGRAPHY

David Padanyi-Gulyas is the responsible System Programmer and Consultant in a private Data-Center in Zuerich, using an HP-3000 and serving about a hundred remote terminals spread over the city. Formerly he worked at HP-Switzerland as an HP-3000 Systems Engineer for over 2 years, following another 2 years-period of sales consultant activity on HP's scientific market. He holds a degree of fine mechanical engineering and optical tooling of the University of Budapest.

Hans U. Benz is chief executive manager of "Benz & Benz Consultants AG, Zuerich" and is responsible for all consultant activities in several application areas. After a period of being a Systems Engineer and Applications Consultant at IBM-Switzerland, he was the co-founder of his recent company 10 years ago. He is a holder of a degree and license of Economic and Business Management of the University of St. Gallen.

NEW IDEAS IN SPREADSHEET PROGRAMS

Ted Rypma

CAROLIAN SYSTEMS INC.

Toronto, Ontario, Canada

INTRODUCTION

The accountant's and financial manager's balance sheet has been with us for a long time. The ledger, with its columns of numbers to be totalled, is quite familiar. The original Electronic Spreadsheet was developed to do away with the tedious and error prone arithmetic operations which had to be performed to total columns and rows of numbers. The computer's memory replaces the paper, the terminal and keyboard replaces pen and pencil and the CPU itself replaces the hand calculator (or user's brain) in doing arithmetic operations. In addition to the simple replacement of these capabilities, the electronic spreadsheet also provides easy formatting of values and titles, simple access to complex functions, the ability to provide iterative solutions to some problems and easy re-organization of the rows and columns making up the balance or ledger sheet.

The end result and object of most spreadsheet applications is the printed report generated on a hard copy device - display of information on a terminal or microcomputer screen is seldom adequate. Any number of input techniques and calculations may be used with a series of formulae in the body of a spreadsheet itself, but the end result is almost always generation of a report or graphics based on a report. It is this capability of a spreadsheet program to generate a report that will be discussed in the remainder of this paper particularly as it relates to data processing and system management in modern businesses.

TRADITIONAL SPREADSHEET APPLICATIONS

The most common applications for the electronic spreadsheet program have been and still are the preparation of budgets, forecasts and financial analyses. The work to set up a model is done once - setting up headings and titles, deciding what calculations need to be done and where and setting up areas for input data. After an initial model is developed, decisions regarding page layout and numeric formatting simply require addition of display format specifications and the movement of entire rows or columns complete with associated calculations. Referenced values in other rows and columns are automatically updated. Extra space for more data or to improve readability of final output is easily created with spreadsheet editing commands to insert rows or columns or to change the display widths of columns.

Many books have been written on the subject of creating spreadsheet models of various financial applications. Most have been written specifically for use with the microcomputer spreadsheet programs on the market today. These books describe implementations of many accounting and financial management and analysis techniques used in business in such a way that specific knowledge of the particular spreadsheet is not even necessary. A useful report may be generated in a few minutes.

DEFICIENCIES

Two almost universal assumptions made by the users of most microcomputer spreadsheet programs and the authors of books written about them are that

1. the volume of data to be input is small and

2. the input will be done manually from the computer or terminal keyboard.

That which sets most modern spreadsheet programs apart from the original, older programs is the ability (and, indeed, requirement) to manage and input large volumes of data on a regular basis. The larger memory of current microcomputers and the advent of minicomputer spreadsheets allows storage and manipulation of much larger quantities of data. The newer spreadsheet programs are also either integrated with data input facilities (ie. direct availability of data from a datacomm port) or have integrated into them the ability to read directly from computer-generated data or report files. Most often the input data comes from another application or program running on the same or another computer facility.

THE SPREADSHEET AS DATA PROCESSING TOOL

The ultimate purpose of most computer programs is to manage data input, do some processing on that data and generate one or more reports based on the data to a hard copy device. The traditional and most common way to create programs is to hire experts in the area of computer program design to consult with users on their requirements, design a system or application to meet the needs, code the application in some computer language and then do further consulting with users on the degree to which the application as written meets the requirements. This entire process may become very lengthy, especially for relatively complex requirements.

Great effort has been made in recent years to reduce the time and effort in moving from end users' requirements to finished, accepted programs. The ideal situation is one in which the user actually completes all or most of the steps him- or herself. The next best situation is one in which the computer professional still consults with the user, but applications are written using more powerful tools than a language compiler. Typically, a program which is capable of general data input and manipulation and report processing is used. The requirements now are to create control directives for this general purpose program rather than actually writing a program, with the result that more time is spent on designing and satisfying user data processing needs instead of coding instructions in a computer language.

Many of the spreadsheet programs, when properly used, fall into one of the last two categories described above: complete design and implementation of a program by the user; design and implementation assisted by a computer professional.

Examples of applications capable of being entirely handled by a spreadsheet program are small parts inventories, simple bills of material, accounts receivable and payrolls and any application in which limited groupings of related information need to be maintained, processed and printed.

The greatest problem in maintaining these types of applications is in getting new data into the model on a frequent, often daily, basis from other more traditional applications and data bases. If the spreadsheet program has a means to allow controlled input of data from system files, this need is made simple and routine.

The following are some examples of spreadsheet use for typical data processing tasks.

1. Internal phone directory

   As an example of a data manipulation application, consider keeping track of a large internal telephone system. There is a need to report costs of telephone services by department for internal billing purposes. Such things as name of the individual at an extension, department responsible, billing account number, hunt grouping, directory listing status, call forwarding capability, number of bells, cord length, connections for data or tape recorders and any charges associated with the above are maintained and reports created for the purpose of charging individual departments for services and options used. Each option listed for an extension has a cost associated with it and it is used to calculate total cost for a particular option or cost of a particular extension. Summaries of many types are then available by cross-tabulating and may be printed by selecting the appropriate section and deleting unwanted details.

2. Multiple departmental budgets

Another example illustrates use of consolidation of data from multiple departmental budgets into a final report based on all departments. Each department keeps a private budget spreadsheet for its own purposes, broken down by month and category. The actual physical format is created by each department head with only the requirement that certain minimum information be kept available. Information is then collected in a non-linear (ie. by name, not by cell co-ordinate) manner from each budget to create a final complete report. This facility was used successfully to predict the impact of salary negotiations on departmental budgets.

3. Energy management

Yet another example of successful use of a spreadsheet application is control and monitoring of energy use for a pulp mill on a daily basis. Month-to-date and year-to-date figures are kept by adding in daily figures through consolidation. An accurate, up-to-date report of current energy consumption is constantly available for management use.

The major point to be kept in mind for the examples discussed above is that not one line of code needed to be written or compiled by the various organizations described. In fact, the end user departments did the design and setup of the spreadsheet models on their own with only some assistance from data processing department personnel. The problem definition was done by the people with the problem, the calculations required were created and entered with the help of systems personnel and report creation and generation is in the hands of the users. This is exactly the aim of most so-called "fourth generation" computer tools.

THE SPREADSHEET AS A SYSTEM MANAGEMENT TOOL

The MPE operating system makes available to the user a fairly large quantity of system related information with a number of commands and system utility programs provided by Hewlett-Packard and by user contribution. These tools do a good job of providing snapshot or instantaneous information, but most provide little or no summary or cumulative reporting capability and others, such as the REPORT command give only summary information. A list of these tools includes HP supported LISTF, REPORT, FREE2, LISTDIR, LISTLOG, and OPT, as well as others such as SYSVIEW, SOO, TUNER, SURVEYOR and SOO2101.

A spreadsheet program may be used to create numerous additional useful reports from these existing tools and utilities by combining and summarizing the data available. Information may be reformatted and grouped for convenient graphing, daily, weekly, monthly or yearly summaries may be done, averages and totals may be computed, data may be sorted various ways before being printed. In short, a spreadsheet may be used to post-process the data available with existing tools to provide more useful reports to help manage a system on a day-to-day, month-to-month or year-to-year basis.

Any spreadsheet program which will directly input data from a disc file will be much easier to use as a system tool, as all utilities may be forced to create disc files using :FILE commands for their list files or $STDLIST. What is even more useful is an input facility which is capable of selectively inputing data from a disc file, picking up for instance, only account related data from a full REPORT command listing or only program file information from a LISTDIR or LISTF lising. Following are some examples of suggested spreadsheet use for system management information.

1. Daily incremental file space usage by account

   By selecting an "accounts only" REPORT, total disc space usage for each system account is available on a cumulative basis. By using the cumulative total from the day before, the last day's incremental usage may be computed for file space, CPU usage and connect time. This information may then be retained on a weekly or monthly basis for graphing or other analysis.

2. Disc free space reporting

   Disc free space can be a critical resource which decreases and increases on a gradual basis. More important, often, than the total disc free space is the distribution of free space sizes. Average number of free space areas may be tracked on a per disc basis by pooling distributions from all disc drives and then keeping a cumulative and day-to-day or week-to-week record for trend analysis. Graphing of the reports produced is most useful.

3. LISTLOG analysis

To do effective LISTLOG analysis, the ability to selectively extract information from a disc file is required. If this is available, information summarizing almost any event is possible, including I/O errors by specific device, logon logical devices by user, files opened and their disc usage statistics. Depending on worksheet organization, it is easy to run out of room with a spreadsheet application of this type. Summary type information about a restricted set of events should be kept.

4. LISTDIR analysis

Account, user, group or file information can be quite conveniently summarized, including more information than is available in a LISTF or REPORT command. Again, the requirement is for the spreadsheet program to be able to selectively input data from a disc file. File selection could be made by file code for reporting of file information organized by program , data base, and KSAM files, for example, for eventual printing of a report.

In all cases where daily, weekly or monthly reports are generated, having the data in spreadsheet format facilitates the creation of consolidated summary reports and extraction of information for graphing or other uses. Only the user's imagination limits the capability and usefulness of a tool like the spreadsheet.

SPREADSHEET CAPABILITIES

To be of real use in today's computing world, a spreadsheet must have the ability to input information from disc files and hardware ports as well as from the keyboard so that massive amounts of data may be handled without tieing up the user unnecessarily. Such bulk input facilities must also allow automatic selection and rejecting of data in order to avoid swamping the program with irrelevant information.

A spreadsheet program needs the ability to perform conditional processing if it is to fulfil its potential as a real "fourth generation language".

Creation of useful reports depends directly on the existence of a flexible print command which allows output of selected sections of a spreadsheet. The addition of a sorting facility for data, both primary and derived, further enhances the ability to produce various reports from the same set of data.

A consolidation feature is desirable to allow easy combination of spreadsheets produced by different departments in an organization as well as to allow the program to handle spreadsheets produced by other spreadsheet programs whose data formats may be different.

Graphics output, though not as important as the above features, is a useful addition to a spreadsheet program and provides greater clarity and visual variety to reports.

SUMMARY

We can fully expect in future that more tools will become available to reduce programming and programming effort. The spreadsheet is such a tool whose usefulness has yet to be appreciated by some. As more spreadsheet programs become available with features such as consolidation, data input and selection, flexible report printing and graphic interface combined with higher performance and larger data capacity, the computer user and professional alike will find less need to write programs for relatively simple applications and tasks.

PERSONAL OR POWERFUL:

CAN WE COMPUTE BOTH WAYS?

Kathy Setian
Hewlett Packard
Cupertino, CA, USA

I. SYNOPSIS

The rapid proliferation of personal computers in business is proof of the growing demand for computing power that is available quickly and easily. Basic human needs for intuitive, flexible tools are being met by manufacturers of personal computers with varying levels of success.

But the personal computer alone cannot satisfy all requirements. Powerful processors, standardization and control are some of the strengths of mainframes and minicomputers. Many companies have sizeable investments in these data resources which users must tap in order to be effective.

The personal computer, ergonomically designed as an extension of the user, can provide an optimum human interface to the network of corporate data resources.

The combined approach of attention to human factors and development of integrated capabilities for personal computers results in an information network that is personal AND powerful.

II. PERSONAL COMPUTING: A QUANTUM LEAP IN DISTRIBUTED PROCESSING

By any measurement, the tremendous growth of personal computers has been an undeniable phenomena, one that data processing professionals in corporations today cannot ignore. Consider some of the statistics:

- From less than 200,000 personal computers in use in 1977, the installed base will grow to 80 million units by 1987, worldwide (1). We are witnessing exponential growth.

- Today, the unit shipment of personal computers roughly equals that of minicomputers worldwide; by 1987, shipments of personal computers will equal that of mainframes (1).

These statistics are impressive when we consider that in 1975, personal computers were toys for hobbyists. But the numbers are even more awesome when we focus on the penetration of personal computers into the business arena:

- Although only 1/3 of the units being shipped go into business as opposed to homes, 3/4 of the dollar value go to businesses (1). This means that much of the serious computing power is being directed into businesses.

This trend toward business applications comes as no surprise when we consider the history of personal computing. With the advent of electronic spreadsheets which offered timely budgeting and "what-if" analysis, cost-justification of personal computing in the business environment became practical. Personal computers expanded their influence from the home into the corporate environment.

But are personal computers here to stay or are they just a passing fad? Compare the penetration of personal computers to other popular inventions with lasting impact:

- The rapid acceptance of personal computers is similar to the pattern of penetration of televisions in the post-war period (1).

- The degree of penetration of telephones into businesses over a 75 year period will take personal computers only 10 years to accomplish (1).

Personal computers are here to stay; and they will change the way we do business in as fundamental a way as telephones transformed corporate life.

What are the needs that personal computing answers in such an undeniable way? Simply put, personal computing addresses a fundamental need to know and to communicate. While distributed data processing has gone a long way toward addressing this need, personal computing represents a quantum leap forward. Historically, distributed data processing meant moving mainframe computing from the corporate data center to local departments with minicomputers. The next step, which is really a leap in order of magnitude, is to move the computing power onto the desktop.

The attraction of this step to end-users is undeniable. The personal computer is personally manageable. Long procurement cycles, high level sign-offs, and lengthy software development cycles can be circumvented. And the personal computer offers the promise of "ease of use". This term used to apply to programming ease, but now it suggests that one need not be a programmer or computer specialist at all. Personal computing is, or should be a "people's" tool.

How successful have personal computers been at fulfilling their promise as friendly, easy, personal tools?

III. COMPUTING CAN BE PERSONAL: ERGONOMICS

To be accessible to the legions of potential users, personal computer design must overcome some formidable barriers. The amount of time spent learning how to use a new tool, and the amount of frustration provoked in the process must be severely lowered. This is largely a matter of software design, and the most successful software vendors in the long run will be those who decrease learning curve constraints creatively. Higher processing capacities and artificial intelligence are also parts of the solution, but will not be covered in this paper.

The goal of a system designer taking human factors into account should be to relate the user's goal to use of the system in a natural or intuitive way. The user should be able to figure out what to do and how to do it without obstruction from the system. Menus as software interfaces presented a partial solution to this design goal, but menus also gave rise to a new question: what is the easiest way to make selections from a menu?

This brings us to a second barrier, one independent of software design. It is the problem of keyboard ineffectiveness. Many alternate input and control devices have been developed over the past years, each with its own strengths and application niches. What follows in this section is a discussion of keyboard limitations and a survey of the most popular input alternatives. These devices are all part of the continuing microprocessor revolution with the goal of personalizing the human/computer interface and tailoring it to the task.

KEYBOARD LIMITATIONS

The following are some of the reasons which have been advanced to explain user discomfort with the keyboard as the main interface to the computer:

- Many new computer users lack typing skills, making the keyboard an ineffective tool and a deterrent to personal computing.
- Unspoken yet powerful subjective factors include keyboard avoidance due to its "non-executive" connotation.
- Environmental factors such as "hands-free" or low light environments.
- The data entry bottleneck caused by the fact that computers can process data faster than humans can enter data.
- Associated problems of data accuracy and reliability.
- The profitability problem caused by the labor intensive attribute of keyboard data entry.
- The productivity drawbacks of long learning cycles.

For these reasons and many others, several alternatives to keyboards have been developed and explored, and ergonomic factors have influenced the success of these new devices.

The goals of ergonomic design are to enhance the comfort and efficiency of the human interface. Unfortunately, since efficiency is the more measureable of the two in terms of ROI (Return on Investment), the "comfort" goal tends to be overlooked by hard-line bottom-liners. However, it must be remembered that a tool which is more comfortable, intuitive, natural and less frustrating is more likely to be used in the first place.

There are more than 20 types of devices used as alternatives to keyboards, and more are being developed. We will briefly survey those devices which are most likely to be of benefit to personal computer users in a business environment: touch sensitive screens, mice, joysticks, track balls, graphics tablets and digitizers, bar code readers, video imaging, and voice recognition.

TOUCH SENSITIVE SCREEN

The most natural, intuitive method of making a selection is by pointing. For this reason, the touch sensitive screen is perhaps the most ergonomic interface available today. It experiences the highest level of user acceptance since it requires no training. It is superior even to the popular mouse, trackball, or joystick since the latter devices require the user to translate information from a vertical plane (video display) to a horizontal plane. Further proof of high user acceptance is the fact that touch screens are being used in public videotex systems and information kiosks where novices walk off the street and interrogate the system without any assistance or instruction.

There are several touch screen technologies in use. Hewlett Packard has pioneered the use of infra red LED's and phototransistors in an array around the screen. A finger (or other pointer) breaks the infra red beams registering the xy coordinates of the intersection. Every line and every other character can be accessed giving a 27x40 matrix.

Other touch technologies include resistive membranes which create a voltage when touched, capacitive sensing which requires an overlay, and acoustic surface waves. Most of these technologies have shortcomings related to reliability, durability, clarity or resistance to environmental factors.

The best niche for touch screen interaction is among executives who must retrieve data but not necessarily enter data. HP's electronic rolodex application uses the touch screen to its best, most intuitive advantage. Vertical application areas where the touch screen is often used include, as an example, stock and commodity traders who are not inclined to take time to use the keyboard.

THE MOUSE

Perhaps the most well-known alternative to the keyboard is the mouse. Sometimes described as an "upside down track ball", it allows users to point to commands or objects on the keyboard with just a little practice. Developed by Xerox in 1968, the mouse became associated with Apple computers in 1983 when it was adopted as their standard interface.

The best applications for the mouse include those that rely on high resolution graphics such as CAD, CAM, and graphics design. For freehand graphics applications, nothing is more akin to the stroke of a brush than the mouse. A disadvantage is the relatively large amount of open work space it requires to be moved across. This contradicts the reality of most people's desks in a business environment. Another limitation is that it is clearly not suitable for portable personal computing.

A new variation is a stationary touch pad located on a keyboard that functions like a solid state mouse.

JOYSTICKS AND TRACKBALLS

In the mid seventies, the burgeoning video game industry borrowed joysticks and track balls from the Apollo space program for home entertainment. However, it is surprising how many of these devices are sold on personal computers. It has been estimated that at least 25% of personal computers have one of these devices, probably due to the large number of business computers that make their way home and double as entertainment devices.

Track balls provide 360 degree motion control, while the joystick offers a more limited number of angles of motion. However, the joystick is much less expensive.

Niches for joysticks and track balls include CAD, CAM, robotics, aerospace and military applications. Because it takes less desk space than the mouse, the track ball may be preferred in some offices.

GRAPHICS TABLETS AND DIGITIZERS

With an electric stylus that picks up current as it passes over a pad, tablets and digitizers compete with OCR (Optical Character Recognition) for scanning and storing graphics. Application niches are high resolution graphics oriented such as CAD, CAM, graphics design, or industrial training.

BAR CODE READERS

In the mid seventies the first standard emerged for bar codes. The Universal Product Code (UPC) is now widely used in retail stores and supermarkets. A predefined set of black magnetic ink bars with varying widths are read by a scanner. During the past decade, several other standards have been established for different industries.

The applications for bar code readers have expanded from point-of-sale to inventory and process control. In the early eighties the development of laser-based hand-held scanners contributed to their use in industry since operators would not have to touch the unit to read its number. The advantages of bar code systems include speed, accuracy, low cost and convenience as an input device.

VIDEO IMAGE INPUT

This is a young, small but growing area of technology. A video camera is used to capture and store images (graphics and data) in digital form. Educational applications and training companies are likely to benefit from this technology. Today, the chief disadvantage is cost, especially due to the high storage requirement.

VOICE RECOGNITION

Voice recognition, along with touch screens, offer the most natural form of human/computer interaction, although voice recognition technology is in its infancy and needs much more development before it can be regarded as a practical keyboard alternative. Today, most voice recognition systems are speaker dependent, meaning that the system must learn and store a vocabulary for each user. By the 1990's, it is hoped that voice recognition systems will be speaker independent and will accomodate continuous speech.

Even today, there are niches for voice driven data entry especially in "hands busy" environments such as assembly lines. As the technology improves and the costs come down, voice input will play a major role in the office.

WHITHER THE KEYBOARD?

With all of the development and excitement of alternate input devices, what will be the role of the keyboard? Will it be replaced entirely?

Studies show that the one class of users that consistently chooses the keyboard over other devices is skilled typists. For these people, it is a disadvantage to take the hands off the keyboard to use another device.

Other indications are that while the alternate devices are much preferred during the learning period, they can reach a point of diminishing return, at which time the user will frequently revert to keyboard use. Although in these cases the alternative device can be regarded as transitional, in the long run, the new devices will be seen as a complement to the keyboard. The keyboard will be used as a general purpose device, while the alternative devices have partiicular application niches as has been indicated.

STANDARDIZATION

Not only is there a wide variety of types of input devices, but within each category there are different implementations. For example, there are both mechanical and optical laser mice, and there exists no standard for connections to computers. Some use RS232 interfaces, others use plug in interface cards, and each device transmits different signals.

Beyond hardware incompatibility, there is an associated cost in software development. Often a software product must be modified to accomodate drivers for different devices.

Hewlett Packard has pioneered a solution to this problem through its HP HIL (Human Input Loop). This is an interface board that connects up to 8 devices. Future HP software products will include an HP HIL driver so that alternate input devices can be added with no subsequent modification to the software. Our recently introduced Graphics Gallery software incorporates the HP-HIL driver. Perhaps independent hardware vendors will choose to take advantage of the HP HIL concept as well.

IV. COMPUTING CAN BE POWERFUL: THE WORKGROUP CONNECTION

Once an interface device is selected which is natural for the user and appropriate to the task, the next challenge for the personal computer user is to integrate the data gathered through a variety of methods and work in tandem with all of the other computer resources located in the company. The essence of powerful computing is this ability to assimilate information from diverse, often traditional sources.

Historically, the information flow within a company was greatly dependent upon the topology of the data processing network controlled by the corporate mainframe computer. The limitations of the mainframe solution included inflexibility, lengthy procurement and development cycles, lack of remote access, and a division between data processing professionals and the business professionals they proported to serve. As we have seen, end users are bringing personal computers into this environment in record numbers, and this will be true whether or not the company has a masterplan for integrating them into the workplace. Furthermore, farsighted business professionals know that they can not stand on the sidelines until such a masterplan is developed before they partake of the advantages of PC technology. To do so would be to concede the productivity and profitability benefits to their competitors.

With the evolution of distributed data processing, many companies find themselves with a dual environment today. Traditional corporate data processing resources and personal computers exist as two separate realities in information management. This situation presents problems. It often indicates a duplication of effort within the company as each department or individual sets out to resolve their needs, many of which are common to other users. Often times leveraged solutions are overlooked as companies rake up the costs of peripherals and software that are not shared among users. Furthermore, the traditional benefits of centralized data processing have been overlooked by PC enthusiasts: data integrity, on-line accuracy, security, back-up, and auditability. All of these issues are seldom if ever addressed by the isolated personal computer user.

Vendors of microcomputers made an initial attempt to bridge the gap between personal computers and corporate data resources by offering terminal emulation capabilities. Clearly this is a limited first step offering little more than the ability to crudely transfer files. It also uses little if any of the power of the personal computer which may be relegated to the status of a dumb terminal.

To fundamentally address the challenge of integrating personal computing with our corporate data resources, we must take a radically different approach, one that might be characterized as "bottoms-up". This means that we must start from the premise that each professional within the organization wants to use the personal computer as an entry point into the network of data processing resources.

As we survey the capabilities needed in an integrated business environment, we see that the propagation of personal computers has if anything increased the importance of departmental minicomputers. However, their application mix has changed to support more multi-user data base oriented applications, and communication functions. Productivity and decision support software is moving off of the mini-computer and onto the personal computer.

Let us look at the kinds of capabilities a truly integrated personal computing and data processing environment would offer, and note how many of these capabilities are within reach with the appropriate planning and development.

INFORMATION ACCESS AND MANAGEMENT

The issues most critical to powerful personal computing revolve around the ability to access data that is stored on one or more departmental level computers, and to manipulate that data in a meaningful fashion as input to a variety of PC applications. There is no justification for a return to the "dark ages" of pre-data base management systems, when each application "owned" its own copy of the data. Rather, we must take the next step forward in distributed data processing by allowing the PC user to extract those portions of the corporate data bases that are relevant and to which he or she has authorized access.

There can be no doubt that such a solution requires extensive sophisticated network level software in order to become a reality. On the other hand, the theory and direction are already in place and have already been subscribed to by those who have preached the benefits of distributed data processing since the late 1970's. What is required is an extension of that commitment and an extension of those software tools.

Some of the software which lends itself most readily to an integrated PC and departmental computer solution include:

PC-based inquiry facilities with a menu driven interface that allow the PC user to easily access data.

- Relational data bases so that data can be extracted (projected) and combined (joined) without predefined limitations.

- Data dictionaries which locate the requested data on the user's behalf.

- Network directories that handle the routing of the data transparently for the user.

The above software facilitates data access; equal attention must be paid to the other side of the coin: data control. A complete solution addresses the concerns of security, data integrity, back-up and restoration of data, and auditability, among others. The integrated PC and departmental computer solution is the most advantageous method of offering these services to PC users. They are simply an extension of the control mechanisms that have already been designed for workgroup computers alone.

INTEGRATED APPLICATIONS AND COMPUTING ENVIRONMENTS

For software solutions to be effective as well as powerful, there must be some integration of PC applications. For example, sophsticated users will not put up with a situation in which data must be rekeyed from a spreadsheet application to a data base application. The popularity of integrated software products such as 1-2-3 and Symphony from Lotus or Framework from Ashton-Tate attests to the importance of integration.

Integration across computing environments is another important goal within the context of combined workgroup and personal computing. PC users want to be able to move their data and applications from the PC to the workgroup computer and vice versa without consciously initiating a data conversion or switching from one interface to another.

COMMUNICATIONS

Although users like the autonomy and control that personal computers give them, one of the first capabilities they request beyond the bounds of a stand-alone workstation is to communicate with their co-workers. This may mean that they want to broadcast a request for information, send out memos and reports, schedule meetings, share data, request feedback on a draft proposal, etc. These communications requirements can be predicted by numerous studies which show that people communicate mainly within the same workgroup or organization. Only 14% of written communication is addressed to people outside of the company, a surprisingly small percentage.

An electronic mail system running on the workgroup computer or network offers PC users an appropiate solution.
The challenge is to tightly integrate personal computers into the

electronic mail network. For example, a user might create a spreadsheet or a memo using a popular PC software package, and then want to mail it to a coworker sitting at the next desk or half way around the world. This process of moving from a local processing environment on to an extensive distributed processing network must be performed smoothly and invisibly to the user.

SHARED RESOURCES

Personal computer users may or may not have ther own printers, plotters, and other peripherals, but it is unlikely that each user would be willing to bear the expense of higher performance and/or higher quality equipment. The best solution to shared use of such peripherals is one that enables the PC user to redirect output to equipment that is attached to the workgroup computer. This provides a method of leveraging the cost of peripherals and still giving PC users convenient access to their output which is located within the immediate workgroup or department.

All of these capabilities constitute the goals and strategic directions for Hewlett Packard's Personal Productivity Center. Hewlett Packard has a unique advantage in its capability to integrate personal computers with office automation, data management and communications. With the personal computer as the professional workstation of choice, and with the minicomputer processing departmental level data, services can be provided which automate and control all of the above functions.

V. CONCLUSIONS AND A LOOK TO THE FUTURE

The personal computer has made an enormous contribution by making computing power available to business professionals. This advance could not have been accomplished without solving some of the human factors problems which stood in the way of wide-spread acceptance, especially among non-typing, non-computer trained executives.

In order to avert the potential disaster of dual, separate computing environments (PC's and traditional resources with no connection), hardware vendors are challenged to produce an integrated strategy. Mini-computer vendors with a long history of commitment to distributed data processing (such as Hewlett Packard) are in the most advantageous position to make a contribution by extending the concept of distributed data processing down to the desktop. This strategy provides a framework within which we can address the need for information access and control, integrated applications and environments, communications, and resource sharing. The total solution requires the planning and execution of software both on the workgroup computer and on the PC. Properly implemented, users can have it all: computing that is both personal and powerful.

What new needs will surface as the current needs are being addressed? Foremost is the need for portable workstations with the same capabilities that have been described for stationary

PC's. The current revolution has gone a long way toward addressing the needs of relatively sedentary office workers, but there are many professionals who are not office-bound. People working in sales and service and executives who travel a great deal are first on the list, followed by a wide range of specific workers from medical ambulence personnel to musicians, writers, and other free spirits who find their inspiration on remote mountain tops.

Another need which already exists but which will become more urgent is the need to integrate solutions among multiple hardware and software vendors. Despite the "shake-outs" which periodically occur in new industries, no single vendor could possibly satisfy all market requirements. Therefore strategies for co-existence, compatibility and integration will increase in importance.

Last but not least is the continued demand for ergonomic design in both hardware and software. We have barely begun to create systems that can truly be labeled "easy to use" or "easy to learn." As microcomputers become more powerful, more functionality will become transparent. One by one, the barriers to use will be removed and the extent of training necessary to get started will decrease. The coming years will push today's limits in both the personal and powerful dimensions.

FOOTNOTES AND REFERENCES

1. International Data Corporation: 1984 Industry Briefing

2. International Resource Development Inc.: Non-Keyboard Data Entry, 1984.

3. Chester, Michael: "The mouse and the trackball face off in PC marketplace", Systems & Software, December 1984, pp. 48-50.

4. Roman, David: "Building up Your Personal Computers: Data Input Devices", Computer Decisions, pp. 111 - 128.

<u>Biography</u>
Kathy Setian is a Product Specialist in the Personal Computer Group at the Hewlett Packard computer facility in Cupertino, CA.

AN INTELLIGENT PROCESSOR FOR 'TEXT-CREATING' USERS

Hans Vanstappen
Sydes N.V., Mechelen, Belgium

Summary

In any text-processing environment, either administrative or EDP-oriented, often a need for standardisation and efficient reproduction of earlier composed texts arises.

This lecture gives a practical solution to this problem introducing an intelligent processor based on a macrohandling concept.

1. Problem description.

1.1 A waste of time in administration.

User manuals, functional and technical drafts, circulars, contracts, reports ... all these documents often contain an important number of standard (or almost standard) paragraphs.
Manuals and functional drafts might have a section with definitions of elementary concepts such as terminal, screen, function keys, menu, ...
Contracts and offers often contain warranty formulations, and a description of the sold products.
Circulars can be built up from a limited number of standard formulations.
Longer reports should have standard title and subtitle layouts (in some cases depending on the printer on which they will be printed) ...

A lot of time is wasted looking up the needed paragraphs in earlier documents and copying them into the new one.
Besides, certain data can change in time. (Prices for instance often do not tend to be constant over a long period. Also warranty conditions can change from a certain moment on.) From the date of change on, it is necessary to avoid copying the old paragraphs.

1.2. Why programmers need to work so late.

Imagine a devoted programmer who has to implement, say ten, more or less analogous interactive programs. The technical draft requires a persistent standardisation in the use of function keys, screen layouts and the handling of database inconsistencies. These requirements demand a great deal of attention.

Our programmer might develop the first program and then start copying and adapting this source, which will yield him all the others. This of course seems a reasonable and efficient way of working. Though when testing the seventh program, our programmer finds a bug in one of the copied parts. Is this a newly introduced bug, due to some error in copying ? Or does this bug also exist in some or all of the previously written programs ? In the worst case, all programs have to be retested.

Some time after delivery, he gets a list of requests from the end-users representative, with questions as :
"Please replace the function key label 'PRINT' by 'PRINT SCREEN'."
"Could you extend the error routine with console logging ?"
Ten sources that have to be adapted and retested.

Moreover, can you ever be sure that our devoted programmer has never forgotten to test the Image condition code, after performing a call to the database ?

2. An outline of the solution.

For the reasons explained above, it might be useful to compile a library made up from standard paragraphs and/or standard coding that can be referenced. This library can be used for drawing up manuals, circulars, contracts, reports and programs. Any standard paragraph, we call it a macro, can be included in the text, simply by referring to its name.

The final text is obtained through expansion of all referenced macronames by means of a 'substitution machine', an intelligent processor with features such as conditional substitution, the use of parameters and nesting of macros. Its userfriendliness can even be improved by an automatic 'forward reference'. This implies that any macro call can precede its definition.

The supplementary processing step may not significantly slow down development time. So a high performance of the substitution machine also is an important requirement.

In the next section, an extensive functional description of the solution is given. Section 4 gives an overview of some results obtained with METRO/3000, a Sydes software product answerring to the description given in section 3.

3. A functional description of the required functions.

The processor matches a library of standard macros (standard library) and a library of application-related macros (specific textfile) to generate a final text or source. Both libraries are ordinary textfiles and can be composed by the user with any editor, according to some very simple grammatical rules for distinguishing macro definitions, macro calls, parameter settings and conditional substitutions.

Schema :

```
 ____________________
| standard library   |______
|____________________|      |
                            |       ____________        ______________
                            |______/ processor  \______| final text   |
 ____________________       |      \____________/      |   or source  |
| specific textfile  |______|                          |______________|
|____________________|
```

3.1. Basic notions.

We have to agree on some special symbols indicating the beginning and the end of a macroname :

starting symbol : <

ending symbol : >

macronames : any characterstring in between a starting symbol and an ending symbol
e.g. <string>
<unlockdatabase>
<checkaccountnr>

3.2. Definition of macros.

```
<macro>::=
|///////////////////////////////////////////////////|
|///////////////////////////////////////////////////|
|//////////// macro body ///////////////////////////|
|///////////////////////////////////////////////////|
|///////////////////////////////////////////////////|
<nextmacro>::=
|///////////////////////////////////////////////////|
|//////////// nextmacro body ///////////////////////|
|///////////////////////////////////////////////////|
```

The end of a macro is given by the beginning of the next one or by the end of the file.

An example :

```
<STROPHE>::=
Old Mac Donald had a farm, Ia Ia OOh
And on his farm he had some cows
With a mooh, mooh here
and a mooh, mooh there
here a mooh, there a mooh
everywhere a mooh, mooh
Old Mac Donald had a farm.
```

3.3. Macro calls.

To call a macro, one simply inserts the macroname in the text.

An example :

Here is one strophe of a popular song :

<STROPHE>

This song has been written during ...

After expansion the macro call <STROPHE> has been replaced by the text from the definition of strophe in paragraph 3.2.

```
Here is one strophe of a popular song :

Old Mac Donald had a farm, Ia Ia OOh
And on his farm he had some cows
With a mooh, mooh here
and a mooh, mooh there
here a mooh, there a mooh
everywhere a mooh, mooh
Old Mac Donald had a farm.

This song has been written during ...
```

3.4. The use of parameters.

The definition of a macro can contain a number of parameters. A parameter consists of a starting symbol, followed by a number of alphabetical characters.
In this text the exclamation point (!) is considered as the starting symbol.

Examples : !PARM
!DATASET
!NAME
!DAY!MONTH!YEAR

When calling a macro, values for all parameters can be supplied. Parameter values stand in between special characters like " " or ' '.

Consider again the example of paragraph 3.2.

```
<STROPHE>::=
Old Mac !NAME had a farm, Ia Ia OOh
And on his farm he had some !ANIMALS
With a !NOISE, !NOISE here
and a !NOISE, !NOISE there
here a !NOISE, there a !NOISE
everywhere a !NOISE, !NOISE
Old Mac !NAME had a farm.
```

We can define a new macro <SONG>.

```
<SONG>::=
***********************************************
*  VARIATIONS ON OLD MAC DONALD'S THEME       *
***********************************************
<STROPHE,NAME='Donald',ANIMALS='cows',NOISE='mooh'>
-----------------------------------------------
<STROPHE,NAME='Peter',ANIMALS='pigs',NOISE='oink'>
-----------------------------------------------
<STROPHE,NAME='Johnny',ANIMALS='ducks',NOISE='quaok'>
***********************************************
```

After expansion of <SONG>, we get a song with three strophes in which every parameter has been replaced by the given value.

```
***********************************************
* VARIATIONS ON OLD MAC DONALD'S THEME        *
***********************************************
 Old Mac Donald had a farm, Ia Ia OOh
 And on his farm he had some cows
 With a mooh, mooh here
 and a mooh, mooh there
 here a mooh, there a mooh
 everywhere a mooh, mooh
 Old Mac Donald had a farm.
----------------------------------------------
 Old Mac Peter had a farm, Ia Ia OOh
 And on his farm he had some pigs
 With a oink, oink here
 and a oink, oink there
 here a oink, there a oink
 everywhere a oink, oink
 Old Mac Peter had a farm.
----------------------------------------------
 Old Mac Johnny had a farm, Ia Ia OOh
 And on his farm he had some ducks
 With a quaok, quaok here
 and a quaok, quaok there
 here a quaok, there a quaok
 everywhere a quaok, quaok
 Old Mac Johnny had a farm.
***********************************************
```

Notice that in the example above, two macros have been nested.

3.5. Validity of parameters.

A parameter value is only valid in the macro for which it has been assigned and in all the macros of underlying levels.

```
<SONG2>::=
<STROPHE,ANIMALS='cows',NOISE='mooh'>
<STROPHE,ANIMALS='pigs'>
<STROPHE,ANIMALS='ducks',NOISE='quaok'>
```

After expansion of the macro call :

```
<SONG2,NAME='Donald'>
```

in each of the three strophes !NAME will be replaced by 'Donald'. The second strophe will yield a problem, since no parameter value has been assigned to !NOISE for that strophe nor for any higher level.

3.6. Conditional substitution.

In many cases a conditional substitution of some macro parts, depending on the value of of a certain parameter is necessary.

For instance, when making offers, prices can be given in the currency of the applicant. Or, depending on the printer on which a report needs to be printed, adapted title and subtitle layouts have to be printed. Or, in a programmer's environment, the error handling routine depends on whether a program has to run on-line or in batch.

Structure :

```
!IF condition !
...
!ELSE !
...
!ENDIF !
```

An example :

```
!IF '!COUNTRY'='B' !
  This product costs 1800 BF.
!ELSE !
  This product costs 200 HFl.
!ENDIF !
```

3.7. Miscellaneous features.

Globally defined macros from the standard library can be overruled by local macros in the textfile.

For COBOL programmers, an automatic line-renumbering feature is indispensible.

Userfriendliness can be improved by encoding file equations for input and output files in a separate commandfile.

Syntax errors such as 'macroname not found' need to be reported.

4. Experiences and results.

A processor as described in the previous section has been used internally since april 1983, mainly for the development of application programs in Sydaid (a fourth generation language) and for composing functional and technical drafts and user manuals.

4.1. General remarks.

As far as programming is concerned, it proved its usefulness on several different domains, such as :

- the standardisation of screen layouts
- the standardisation of function key handling
- the uniformisation of database calls and KSAMfile calls with condition code checks
- special routines such as checks on bankaccount numbers or VAT numbers or other project related control routines.

Although the starting-up period of a project is longer, (all standards have to be implemented first), the use of the processor leads to an important increase in the productivity during the other phases.

For elementary programs, such as a maintenance program on one dataset with read, write, update and delete capacities, the amount of coding can be reduced to less than one third of its original length.

There is not only a gain of time, due to the reduction of the total number of lines to be typed. But, as macros are in fact higher-level commands, also the programmer's way of thinking is simplified and this again implies a higher productivity.

Debugging time decreases significantly, as enormous blocks of coding need to be tested only once, or have already been tested during previous projects.

Maintenance also takes less time, as a lot of modifications can be made on the standard library instead of on each individual program.

4.2. A case-study.

For a project consisting of 44 interactive screens and 5 batch programs, a Sydaid library with 2824 lines of standard coding has been set up, covering the domains mentionned in paragraph 4.1.

The length of specific coding totals up to 31082 lines. The total length of all final sources after processing came to 81053, which is a factor 2.6 greater.

In Table 1, some results for different kinds of programs and groups of programs.

Table 1.

Programs or groups of programs	number of lines before (B)	after (A)	expansion factor A/B
The whole project	31 082	81 053	2.6
An interactive maintenance program on one dataset	513	1 813	3.5
A small batch program without database calls	142	220	1.5
A simple interactive program without database calls	387	1 155	3.0
All interactive programs with database calls	22 575	63 471	2.8
All batch programs with database calls	5 849	14 054	2.4

As mentionned before, performance of the processor was one of the most important requirements.

Table 2 contains some performance results for various types of programs. The standard library contained 175 macro definitions, one third of it with one or more nested macro calls.

All tests have been performed on a HP3000 series 42, with 2 Mb central memory and a 400 Mb disk with disk caching.

Table 2.

programs	length before	macro calls	length after	CPU time(sec)	elapsed time (sec)
interactive progr with DB calls . .	7493	778	21149	85	118
interactive maintenance progr on one dataset . .	513	74	1813	13	17
batch program without DB calls .	666	9	1017	9	12

5. Bibliography.

[1] R. Huysmans; Preprocessing, Het middel voor standaardisatie en productiviteitsverhoging; Sydes 3000 Newsletter 1:3 ; November 1984.

[2] METRO/3000 reference manual, version A02.00; Sydes NV, Mechelen, Belgium; July 1984.

6. Biography.

Hans Vanstappen was born in 1959. He studied mathematics at the University of Antwerp. After working some time as a programmer and an analist for applications software, he became a project leader at Sydes NV. He is also active as a teacher in computer science. A great deal of his free time, he spends on astronomy.

Man-Machine Interface in Technical Publications Simplified with Laser Printing

Steve Wilk, Marketing Support Manager
Sam Boles, Member Technical Staff

HEWLETT PACKARD

In Perfecting the Man-Machine Interface you usually find a high correlation to simplicity: the simpler the interface, the closer to perfection.

In the traditional approach to technical publications you find a degree of non-simplicity: interface with the typesetter, interface with the printer, interface with the graphics artist and perhaps the labeler -- "machines," if you will, from the user viewpoint. Each interface has its own queueing time, its own variety of entropy, its own set of weak links, its own set of surprises.

At the HP Systems Productivity Center in Cupertino, a team has developed a methodology for producing technical publications using the laser printer and computer graphics in a network of HP3000's and HP9000's. It simplifies the Man-Machine Interface -- eliminating the typesetter, the printer and others in the typical publication loop.

And while it's not ultimately perfect, it's cut time and dollar costs in half.

[Note that we used the methodology described in this paper to typeset it.]

. . . in the technical publication business . . . we must have quality, speed, economy, flexibility . . .

At the Hewlett-Packard Productivity Center in Cupertino, our charter is to ***integrate HP productivity tools*** into our day-to-day operation in a way that can be ***examined by customers***. It's a sort of do-it-yourself course ***in vivo***. And we do it in a way that can be ***exported*** to our field support personnel around the world. We perform a ***clearing house*** function. This is where we get our ***leverage***. And this is how we're ***in the technical publication business***, as a sideline.

To get this part of the job done right, we must have ***quality, speed, economy, flexibility***.

Our ***quality*** target is first-class content in middle-class packaging. What ***middle-class packaging*** means here, basically, is that we ***don't have to have color on slick paper***. But we do have to have ***type font crispness and range*** in the neighborhood of phototypesetting. Our ***graphics*** requirements include ***high resolution line drawings*** like the one below.

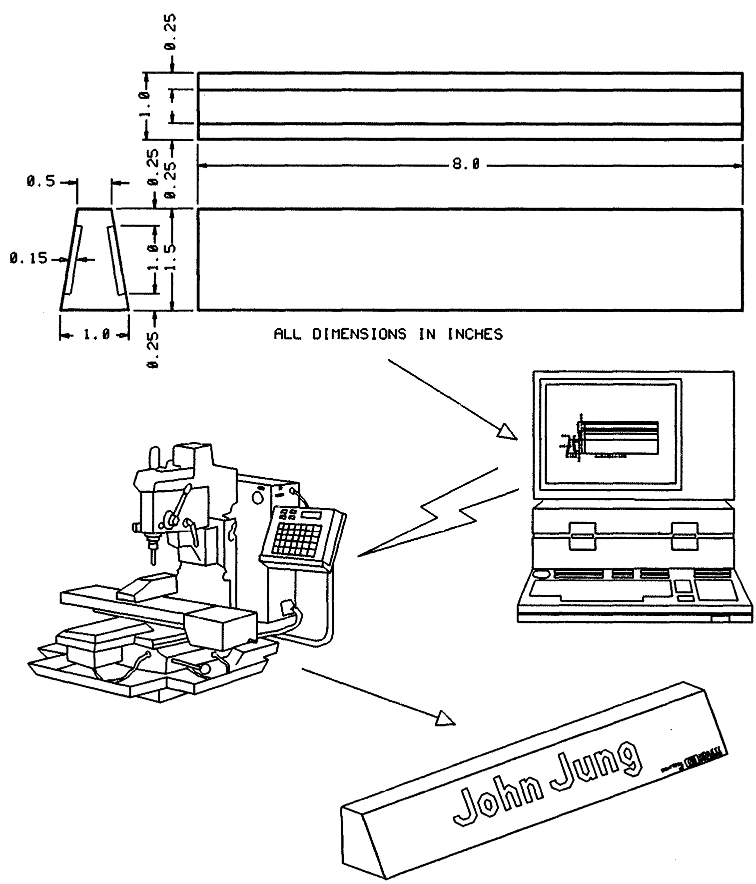
Our ***speed*** requirement does not allow a week or two for iterations with the ***typesetter***, plus a week or two for iterations with the ***printer***, plus the usual turn-around queues of the ***commercial artist, labeling operation*** and other components of the typical publication loop.

We've got the leverage of ***thousands*** of Hewlett-Packard ***field support personnel*** around the world and thru them ***tens of thousands*** of Hewlett-Packard ***users*** around the world. ***Depriving these people of a***

powerful productivity idea for an unnecessary month is not something our charter allows.

Our ***economy*** target is the ***cut-rate alternative*** that typically compromises quality and speed. Our objective is to ***match the cost of the cut-rate alternative. But without the compromise of quality and speed.***

Our ***flexibility*** requirement is what you'd expect for a leading-edge operation. A technology break-thru Friday afternoon preempts our feature article, but the Monday publication target ***can't slip***. We get a new measurement device set up with an RS232 output, and we want to feed that directly into our text for "typesetting." For a special issue we want to include a special interest group in our automatic mailing. The methodology has to have enough flexibility to accommodate a ***very dynamic environment***.



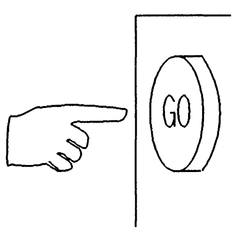
A drawing done with high-resolution high-performance CAD software on the HP9000, then ported to the HP3000 for transformation and integration with text for downloading to the HP2688 laser printer.

In Perfecting the Man-Machine Interface . . . the simpler the interface, the closer to perfection.

One day at the Hewlett-Packard Productivity Center in Cupertino we were brainstorming a user interface problem. As you might imagine, we do this a lot. Typically it's the ***Man-Machine Interface*** that ***determines whether*** a really powerful piece of ***new technology ever becomes a useful productivity tool***.

One of the team said he thought the ideal interface is ***a single big button*** that says "Go." All you have to do is push the Go Button, then all manner of magic comes billowing out while you just stand watching, amazed, with your hands in your pocket.

No decisions. No keying. No effort beyond pushing the button.



Then another member of the team -- she's the one who always takes our ***great ideas the next step*** -- said why not eliminate the button pushing and just have an ***actuator pad*** on the floor in front of the box. Then when you walk up and stand in front of the box all manner of magic comes billowing out while you just stand watching, amazed, without ever taking your hands out of your pocket.

Then old Analytic Andy, who hadn't said anything so far, looked as tho he wanted to say something. We always listen when Andy has something to say; he watches a lot of ***Star Trek*** reruns. He said why not rig a ***brain wave sensor*** and save having to walk up to the box and step on the actuator pad. You wouldn't even have to open your eyes.

Well, we ended up just porting a softkey module we'd gotten from somebody at HP Labs. It doesn't ***Perfect the Man-Machine Interface*** but it's good enough for the prototype.

The conceptual key here is ***Simplicity***. Let's look at this concept in our approach to our technical publication problem.

. . . the simplest interface: none at all . . . that's how we interface with the typesetter . . .

Our first interface was with the ***typesetter***. Sometimes the ***stickiest***, too. The typesetter was the whipping boy, the one at the end of our pop-the-whip line. The typesetter was the one who got all our typo's, all our illegible marked-up copy, all our last minute changes, all our venom when he didn't make a silk purse out of the pig's ear we gave him. And we really leaned on him to make up that 2-week overrun we'd perpetrated in getting our stuff "ready" for him.

How'd we handle this interface? With the ***simplest interface: none at all***. That's how we interface with the typesetter: we don't.

We do our forms, logos and character set-up with IDS/3000. We use IDSCHAR ("IDS" means Interactive Design System) software and an HP2647 graphics terminal to design our logos and special characters. There's a fairly rich set of fonts available on the HP2688 (you're looking at some of them on this page) so you don't need to do very much of this tedious work, especially the tweaking that you're always tempted to do -- even though ***you're the only one*** who'll ever notice that one dot out of the 300 per inch that's a little bit out of line.

For forms design we use IDSFORM software and the 2647. This gives us layout and shading capabilities within our quality and flexibility requirements.

IDSCHAR and IDSFORM generate dot matrices and vectors that are ***device-independent***. This data is then compiled by IFS ("IFS" means Interactive Formatting System), into an ENVIRONMENT file that is compatible with a ***specific I/O device***, eg the HP2688 laser printer. The HP3000 then downloads the ENVIRONMENT file to the laser printer.

The ENVIRONMENT file loads character sets (up to 32 different character sets at a time) and forms (up to 32 different forms at a time) into the memory on board the laser printer. This is used by the processor in the laser printer to control the laser so you get an 8-point Helvetica Bold ***when you ask for it*** instead of a 10-point Roman Italic.

This process can consume a few hours or a few days. Some of this time is ***learning curve*** and some is that "one last touch" of the ***amateur artist***, but most is the time necessary to design the form or logo and enter the points and other specs.

So your basic set-up time is a function of the complexity of the target product and what kind of bit-chaser (or dot-chaser) you are.

The real advantage is the fact that you can do it all yourself without the start-stop iterations, queues and communications entropy of a separate department or company.

The initial set up can be done in ***one continuous uninterrupted block*** of time. There are no meetings or key resources to schedule. There is no "now let me see where was I two days ago when I last worked on this" re-think time.

. . . TDP . . .
the gateway to doing your own typesetting and layout . . .

We use ***TDP*** (Text and Document Processor) for this because we need 2-column capabilities (to enable speed readers to bounce only vertically) and right justification of proportional character sets (to get more balanced "Linotyping"). Also, we need the capability of including ***graphics*** from DSG (Decision Support Graphics) and other graphics sources.

We specify the appropriate FONTID's to TDP to couple with the ENVIRONMENT file, give it margins, columns, page length, and the like and proceed to do our typesetting and layout.

For the layout of charts (bar/line/pie) from DSG and illustrations from HPDRAW and other graphics sources, we do a lot of trial and error placement, so we use a template like this:

```
/* ILLUSTRATE rasterfile
/NAME RASTER rasterfile
/ILLUSTRATE figurefile:figure #lines
```

We do the NAME raster file (to keep the temporary file) each time we change the size or the figure (be sure to explicitly PURGE the old raster file as needed) and therefore have to do another vector-to-raster conversion via the ILLUSTRATE command. Then for the next pass we comment out the NAME and second ILLUSTRATE, and de-comment the first ILLUSTRATE so we pull in the raster form ***without having to do a vector-to-raster conversion when not needed***:

```
/ILLUSTRATE rasterfile
/* NAME RASTER rasterfile
/* ILLUSTRATE figurefile:figure #lines
```

This ***template technique*** we extend to the issue level, so for the subsequent issues the engineer who has responsibility for a given issue can "cookbook" from the archive file of the previous issue without re-inventing the TDP commands. This saves time and gives consistency to the appearance of the publication.

We run through a number of iterations on our publications, balancing artwork, policing "widows" (TDP can do a lot of this automatically), cleaning up typo's and again falling into the "one last touch" syndrome of the amateur artist. In these iterations we get a significant ***productivity surge in cutting out the middleman*** -- eg the typesetter. We do a lot of the local typesetting on the fly as we write the articles at the terminal. The global typesetting and layout commands are largely templated from a previous publication. The cut-and-paste we do by computer. And the inevitable last-minute "stop-the-presses" newsflash can be accommodated with only a little pain in a matter of minutes rather than days via the typesetter's queue.

. . . simplify the printer interface . . . by reducing it to collating and stapling . . .

We simplify the ***printer interface*** by reducing it to ***collating and stapling***. We print multiple copies of each page using that parameter of the environment file, so we get performance on the order of 10 ppm. We label the cover page on the fly with a build rate on the HP3000/4x about 1.5x the print rate of 10 ppm on the HP2688. For volume we run multiple 2688's, dedicated to a single page. On weekends this can handle 500 copies of a 10 page publication with labels printed on the fly.

We set OUTFENCE high to prevent interleaving by random print-outs and to enable us to drain the queue as needed but in a controlled fashion. We do a HEADOFF to save paper and the labor to separate. We do a TDP FINALQ of our text file to suppress the TDP message at the end. We set COPIES=n on the SPOOL file to get multiple copies if the count in the ENVIRONMENT file doesn't cover it. Note however that this approach downloads the ENVIRONMENT file for each "copy" and this has a material performance penalty for heavy-duty environments.

. . . simplify the distribution interface with labels printed on the fly, presorted . . .

For labels on the fly we originally tried the TDP MAILER facility. The MAILER is good for multiple addressees for text that ***does not have a heavy FINAL formatting load***, since its formatting performance is about O(n). With 10 pages of 2 column right-justified proportional text plus graphics inserts, we may take several minutes for a FINAL even with the graphics already in raster format. An O(n) performance with hundreds of FINAL's at several minutes each is ***not operationally feasible*** for us so we use a SPOOK APPEND splice technique.

The redundant APPEND's (you're APPENDing a text set for each addressee and an ENVIRONMENT set for each 10-25 addressees) builds a new SPOOL file that prints at close to the nominal rate.

This isn't elegant, but a ***viable prototype*** that's functionally and performance-wise feasible for the volumes we deal with, until we engineer a little finesse to replace this brute-force method.

So there you have it. Our approach to the classic problem of technical publications. A subset of the global Man-Machine Interface problem. We simplify the interface. Reduce it to the degenerate case if we can, and it goes away. We currently have a working methodology that handles our volumes within the constraints of our quality, speed, economy and flexibility parameters. Our working prototype has some primitive elements as you would expect with first-generation software, but it beats our next best alternative in both time and dollar cost. This paper that you're reading is a living example of the methodology: without it we could not have had this level of quality by the proceedings publication deadline.

About the Authors ....

Steve Wilk *is the Marketing Support Manager at the Hewlett-Packard computer facility in Cupertino, California, with responsibility for the Productivity Center and the Marketing Response Center. His 19 years in the industry cover a wide range, including programming, systems analysis and program management. With an undergraduate degree in mathematics, Steve received his MBA from the University of San Francisco.*

Sam Boles *is a Member Technical Staff in the Productivity Center at the Hewlett-Packard computer facility in Cupertino, California. With HP since 1976, his computer experience started back in the AUTOCODER days of the 1401/1410, migrated thru the 360/370 era, and now focuses on networking HP productivity technology. Sam received his MS at UCLA in Information Systems.*

sebiug13 1500 27dec84

EXCHANGING ELECTRONIC MAIL BETWEEN HPDESK, PROFS AND OTHERS

Paul Wilson

British Columbia Systems Corporation, Victoria, British Columbia, Canada.

SUMMARY

Electronic mail systems are in vogue, they have proven to be an office automation tool almost as important as word processing. The concept transcends specific implementations and has created an immediate requirement for inter-system message exchange.

This paper outlines how mail is exchanged between HP3000 systems using HPDESKMANAGER and IBM 4300/VM systems using the Professional Office System (PROFS). Some of the concerns and problems that have arisen in the course of the exercise are also noted.

1.0 ELECTRONIC MAIL EXCHANGE - HISTORY

1.1 THE USER'S VIEW OF ELECTRONIC MAIL EXCHANGE

The advantages of electronic mail are now known within many functional areas of the Government of British Columbia. Although these departments are separate entities, there are many instances of regular inter-departmental communication.

The executive levels, purchasing, accounting, personnel and computing services constantly cross these boundaries with information exchange. The pressure for a government-wide electronic mail system increases every day.

Since all departments do not use the same computer architecture, the different electronic mail systems do not automatically link into a useful network. However, the technical differences between systems and the costs of linking them are no match for the perceived advantages. In fact, if two groups that have mail facilities wish to communicate, the technical barriers are seen to be very artificial. The dialogue between the user groups and their technical support staff may proceed as follows:

USER:	I want to send mail to J. Doe
TECH:	Sorry, you cannot
USER:	Why not?
TECH:	Because you use HPDESK and they use PROFS!
USER:	So what?
TECH:	The systems are different and do not communicate with each other
USER:	That's not acceptable.

Various solutions are then discussed ranging from dual terminals, (one for each system), the elimination of one of the mail systems (if not the entire machine!) and, of course, programming the systems so that they exchange mail automatically.

1.1 THE USER'S VIEW OF ELECTRONIC MAIL EXCHANGE (cont'd)

The conversation takes various twists and turns, but the need to exchange messages persists. I believe the desire for simple, effective communication is strong enough to override more traditional data processing considerations such as costs, performance, etc. In the case of Hewlett Packard for example, the ability to effectively exchange messages, documents, etc. with IBM systems is very strategic. This capability or the lack of it could be the determining factor in new installations or in the continued success of HP co-existing in an IBM environment.

Having been assured that message exchange is feasible, the user has several basic assumptions about the proposed interface:

1. The procedure for sending mail to an "external" user will be simple, in fact, the same as for a local user. No special codes, menus or routing information will be required the system will handle it automatically.
2. The delivery of messages will be totally reliable. Once sent, the mail will go through.
3. Unauthorized access to notes will be prevented. This is a general assumption that encompasses all systems involved in the delivery process.
4. Delivery will take place with no noticeable delay.

The HP and VM support groups of the BCSC Advanced Systems Centre embarked on a joint project to exchange mail between HPDESK and PROFS, initially addressing item 1.

1.2 BRITISH COLUMBIA SYSTEMS CORPORATION

British Columbia Systems Corporation (BCSC) is a company wholly owned by the government of the Province of British Columbia, Canada. The company provides data processing services to all departments of the government, such as analysis of alternative systems, acquisition and installation of selected systems, technical support, research activities related to software and hardware products and enhancements, and of course, computing and network facilities.

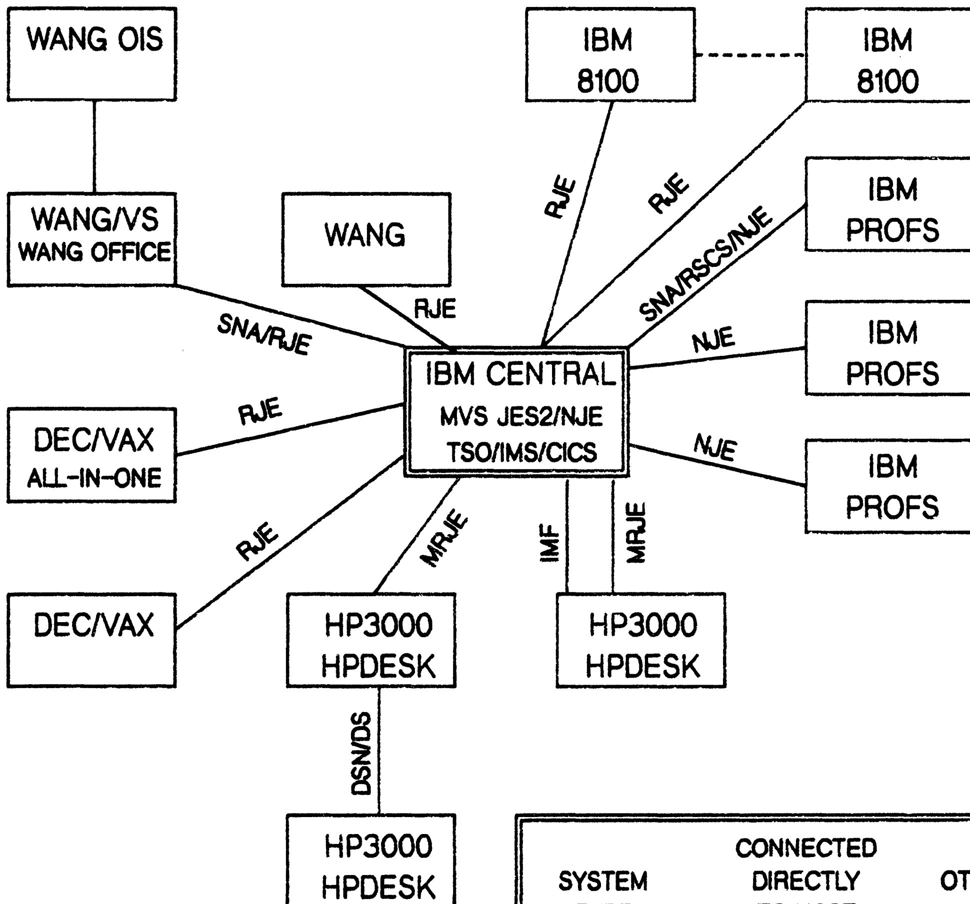
The main computing service is provided on a large IBM system running MVS, TSO and IMS. Network Services are provided to IBM users via a private province-wide SNA network.

Besides using the IBM host services, many departments have distributed their processing on HP3000, DEC/VAX, WANG VS & OIS, IBM 8100 and IBM 4300 systems. BCSC maintains technical support staff and an in-house "testbed" system representative of each of these architectures.

1.3 ELECTRONIC MAIL & BCSC

During the last two years, electronic mail was introduced within BCSC as a pilot project on an HP3000 system using HPDESK MANAGER (HPDESK). The pilot was successful. The concept of electronic mail has been widely accepted and now is in use in several government departments with the largest user community being PROFS[1], on three IBM 4300 systems running VM/CMS. The SNA network maintained by BCSC which connects over 2000 327x terminals to the central system, provides wide potential access to the PROFS systems.

BCSC MAIL EXCHANGE COMMUNITY



SYSTEM TYPE	CONNECTED DIRECTLY TO HOST	OTHER
WANG VS	4	
HP3000	3	1
DEC VAX	2	2
8100	10	
4300	4	

FIGURE 1

1.3 ELECTRONIC MAIL & BCSC (cont'd)

Most other distributed systems, including the four HP3000's are connected to the main IBM system via RJE links. The RJE protocol provides the transport vehicle for mail between the various systems. Although limited in function, RJE has the advantages of being in widespread use, reliable (proven software) and simple to use. Mail is currently exchanged between PROFS and HPDESK and there are plans to exchange mail with IBM 8100 systems, DEC ALL-IN-ONE, WANG OFFICE and HPDESK users not connected via DS links. (See Figure 1) BCSC also intends to implement an X.25 network which will facilitate development of data exchange between different systems without the use of an intermediate IBM system.

The next sections will describe our first pass at providing mail exchange between HPDESK and PROFS, in which we address not only the delivery of mail, but also point 1 above. Points 2, 3 and 4 (among others) are currently the subjects for discussion but have not been satisfactorily realized to date.

2.0 HPDESK TO PROFS MAIL DELIVERY (Figures 3 and 4)

2.1 SUMMARY

To get mail from HPDESK to PROFS:

- define PROFS users as usernames in a remote HPDESK mail node
- HPDESK delivers mail for these users to MPE files external to the mail database
- a user program reads these files and transforms the HPDESK names to PROFS userids using a lookup file
- the message text is reformatted with the new userids, enclosed in IBM JCL statements and transported to the central IBM system via MRJE
- the routing information in the job stream causes the data to be delivered to the VM/CMS system running PROFS
- another BCSC program running on the VM/CMS system places the mail into the receiver's PROFS in-basket

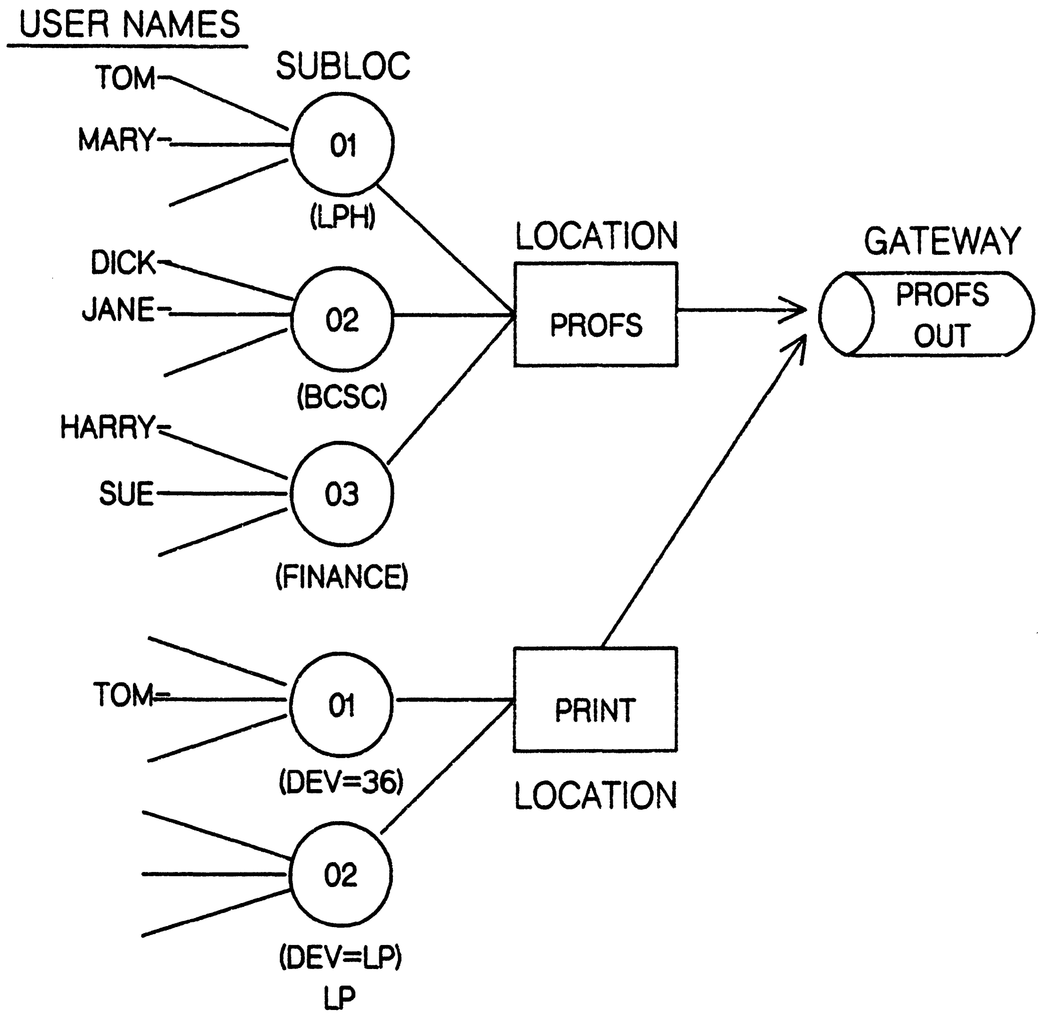
2.2 DEFINING REMOTE USERS AND LOCATIONS TO HPDESK

The detailed procedure for defining PROFS users to HPDESK is fully described in the HPDESK MANAGER Administration Guide. A summary of this procedure follows.

The HPDESK administrator uses the MAILCONFIG program and defines a remote node (e.g.) PROFS/01. In the Sub-location entry screen the remote designation must be "Y".

The routing of mail for each remote sub-location is specified via the NETWORK menu. First a Foreign Service Connection (FSC) "gateway" must be defined using the EFT/FSC selection.

This procedure will name a gateway and link it to an actual MPE filename into which HPDESK will deposit information about messages to be delivered to the remote sub-location. The mail administrator can also specify a gateway reference name which is appended to the sender's mail node address as part of the header information. The format of mail is selected; we use "ARPA format" (See Figure 2) for our interfaces in HPDESK II and "printable ASCII" in HPDESK I. (Other HPDESK II formats are described in the HPDESK Programming Reference manual.)



HPDESK REMOTE USERS

FIGURE 2

2.2 DEFINING REMOTE USERS AND LOCATIONS TO HPDESK (cont'd)

Finally, the delivery schedule is selected as for HP-to-HP remote locations. (Our experience indicates foreign mail is put in external files shortly after being "mailed", regardless of the schedule.)

The mail administrator must actually create the MPE gateway file named in the FSC/EFT screen as an 80 byte, fixed ASCII message file.

(eg) BUILD PROFS.IPC.HPOFFICE;REC=-80,16,F,ASCII;DISC=250,2,1;MSG

After defining the gateway, the administrator returns to the NETWORK menu and selects ROUTING. Remote sub-locations are connected to specific gateways in this screen.

Users can now be added to the remote sub-location. The mail administrator must also ensure that each PROFS username, HPDESK location, and corresponding PROFS userid and SNA node are entered into the lookup file. (See Appendix A).

2.3 HOW HPDESK SENDS MAIL TO REMOTE USERS

When mail is addressed to a user at a remote location, an MPE file is created by HPDESK. If ARPA format is selected for this remote location's delivery format, an ARPA header is prefixed to the mail message and it is written to the MPE file. A record is then written to the gateway file giving the mail filename, format type, destination sub-locations, etc.

Once all this is accomplished, the message is considered "transmitted" by HPDESK and it is now the sole responsibility of the external user-written interface to complete delivery of the mail.

2.4 THE HP TO PROFS DELIVERY PROGRAM (Figure 3)

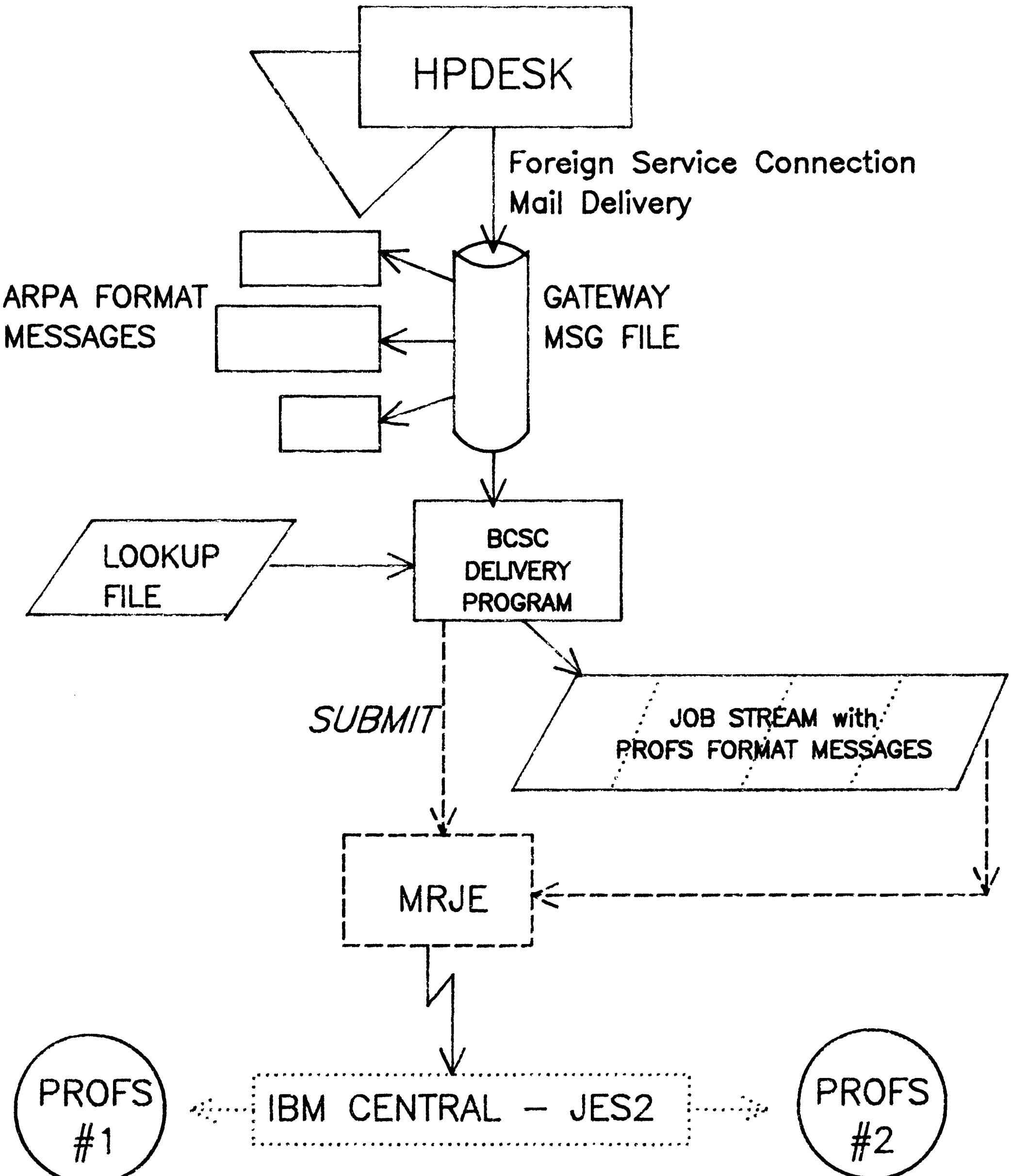
The delivery program reads the external mail files created by HPDESK, transforms the userids to PROFS equivalents, prepares an MRJE job stream and "submits" it.

The program initializes itself by checking the availability of the MRJE system and HPDESK. MRJE is used to transport mail; HPDESK is accessed by activating a process and is used to deliver error messages to the mail administrator.

The program accesses the lookup file and ensures that work files from previous runs do not exist. If work files exist, it is assumed that the program has previously aborted and the system should be checked for errors and undelivered messages.

The HPDESK FSC message file is periodically checked for records. If none are present, the program checks another message file where the system operator can place a shutdown command.

If a mail message record is found, the actual text file is accessed and the header is processed. The lookup file is accessed to determine the destination names and locations (ie) PROFS userids and SNA nodes. If no match is found the error is logged and a message is sent to the sender and an HPDESK administrator using the HPDESK process. (There is an administration utility program which allows re-queueing of a message once a correction has been made). The message is delivered to all addressees successfully located.



HPDESK to PROFS - HP VIEW

Figure 3

2.4 THE HP TO PROFS DELIVERY PROGRAM (Cont'd)

The sender's id is also referenced in the lookup file and replaced with the corresponding PROFS id. This allows the REPLY facility to be used by receivers on the foreign system.

IBM JCL statements are created for NJE delivery, a PROFS header is built (APPENDIX D), prefixed to the message text and written into a collection file. When all messages have been processed for all addressees, the collection file is "submitted" to MRJE as a job stream.

Once submitted there is no further checking of successful delivery. This poses a reliability problem which is discussed further in section 5.2.

2.5 HOW PROFS RECEIVES AND PROCESSES FOREIGN MAIL (Figure 4)

Since PROFS systems have no inherent export facilities, the procedures and programs required to deliver mail to and from the PROFS environment were developed by the BCSC technical support group. The PROFS "OPENMAIL" procedure has been modified to process mail delivered by NJE in the format in Appendix D.

Once submitted, the IBM host-based JES2 system delivers the job stream to the appropriate SNA NJE node, in our case an IBM 4300/VM system. The job stream is attached to the virtual reader of the user addressed on the /*XEQ card as a "reader file". The PROFS "OPENMAIL" procedure inputs these reader files and transforms them into "NOTES" in the user's IN-BASKET.

All information required by PROFS can fit onto a "card", since all mail userids are the same as the VM/CMS logon userids and are limited to eight characters in length. This single header format will be limiting to other mail systems and is discussed in section 5.

For example, there is no priority or privacy levels available to PROFS users. Acknowledgement of delivery of mail within PROFS is available, but not to foreign systems.

Due to the nature of the PROFS system wherein all users have their own "virtual machine" (computer system). Users do not share a common database and mail distribution lists are not useful in the header of the delivered messages. Therefore, a single message with two addressees on the PROFS system is sent twice by the HP system, one copy to each addressed user.

If volumes warranted, this inefficiency could be handled by creating a mail service on the VM/CMS system to receive mail for all users and perform the distribution function.

3.0 PROFS TO HPDESK MAIL DELIVERY (Figures 5 and 6)

3.1 SUMMARY

To get mail from PROFS to HPDESK:

- define HPDESK users in a remote format in the PROFS nickname file
- mail sent to these users is intercepted by a BCSC program (exec), the "POSTMAN"
- the RSCS routing information is modified and a header placed on the message
- the JES2 system delivers the mail as print output to the HP3000 via MRJE

PROFS TO HPDESK - PROFS VIEW

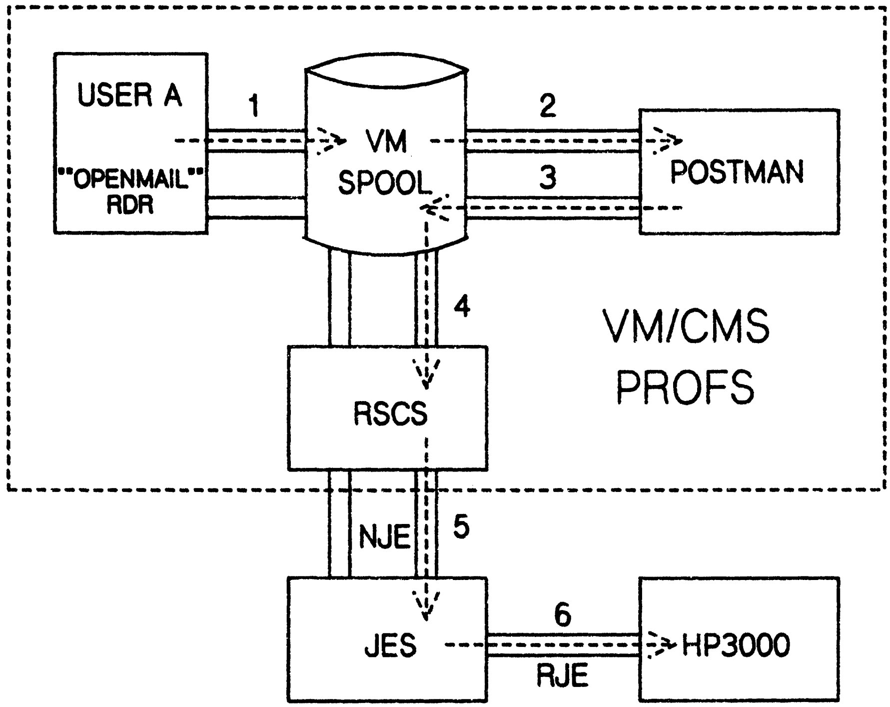


FIGURE 5

3.0 PROFS TO HPDESK MAIL DELIVERY (cont'd)

3.1 SUMMARY (cont'd)

- a BCSC program on the HP3000 reformats the mail received into ARPA format
- an HPDESK utility program is invoked which places the mail into the mail database

OR

- the mail is placed directly into the mail database by a created HPDESK process.

3.2 PROFS MAIL DELIVERY TO HPDESK (Figure 5)

PROFS delivers mail to all users by means of the VM spooling system. This mechanism is used for general data transfer between virtual machines (users) since the spoolfiles provide a shared resource. PROFS does not utilize a common database for notes, logs (filing cabinets), or calendar data. An extension of the spooling system is the remote Spooling Communication System (RSCS), which allows VM systems to exchange data, in our case, through a central system with JES2. This system is used to exchange mail between the three PROFS systems currently in the network.

BCSC has modified the standard system by implementing an executive procedure (exec) called "POSTMAN" which intercepts outgoing mail. If the destination is known to be another PROFS system, the message is passed on to RSCS as usual. If the destination is determined to be a foreign system, such as an HP3000, the message is modified with the addition of a header and the delivery instruction tag for RSCS changed so that the message is passed to the HP3000 as (unsolicited) output from the IBM central system. The POSTMAN is also used to deliver mail to spooled printers and TSO users on the central system.

3.3 RECEIVING MAIL VIA MRJE (Figure 6)

Foreign mail is transported to the HP3000 via the same MRJE link as is used for outgoing messages. Incoming mail initially arrives as output for IBM output class=T, special forms=MAIL. This output class is routed to an MPE message file designated in the MRJE configuration. (See Appendix B). The RJE configuration on the IBM system specifies no operator intervention is required for the special form. The format of mail received consists of a two card header followed by the message text. (Appendix D).

Waiting at the other end of this incoming mail pipe is another BCSC-written program which delivers the mail to HPDESK. The sender and receiver names are translated to HP usernames using the lookup file. The actual delivery mechanism depends on the version of HPDESK.

HPDESK TO PROFS - PROFS VIEW

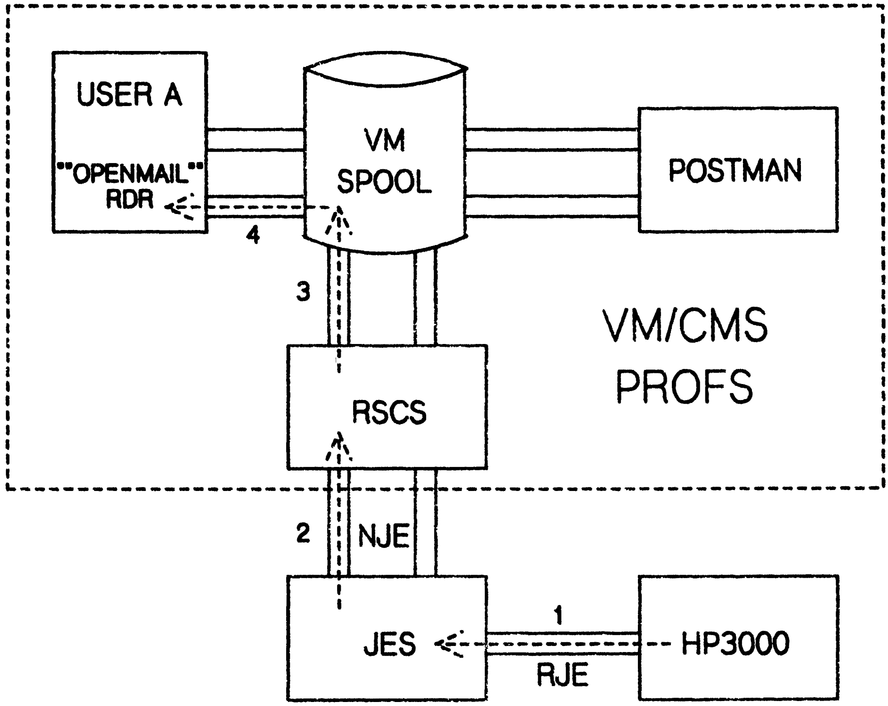


FIGURE 4

3.4 HPDESK I - PROCESSING PROFS MAIL

For HPDESK I, the program creates an HPDESK process, reassigning the standard input and output files (normally the terminal) to two message files. Using the input message file, the export program logs on as a special user, "PROFS TO HP", and delivers messages to the "real" users designated in the incoming message. The receiver can tell who actually sent the mail as the sender name and distribution list are included as part of the message text. This delivery method depends on knowing the exact sequence of the dialogue with HPDESK and could be prone to release changes.

Users receiving messages are unable to REPLY, since replies would merely go to "P.T. HP", the "sender". In order to assist users in avoiding this situation, a warning is appended to the message text.

3.5 HPDESK II/III - PROCESSING PROFS MAIL (Figure 6)

Subsequent releases of HPDESK will provide a clean method of foreign mail delivery. Essentially the user program follows the same procedure HPDESK uses to export mail.

The incoming mail transported by MRJE is scanned by searching for a PROFS mail header (see Appendix D). Each message found is converted to ARPA format (Appendix C). The ARPA header is created using information from the PROFS header and the lookup file. A separate MPE file is created for each message received and a record pointing to each file created is written to a message file using the same format as the "gateway" file described in section 2.3.

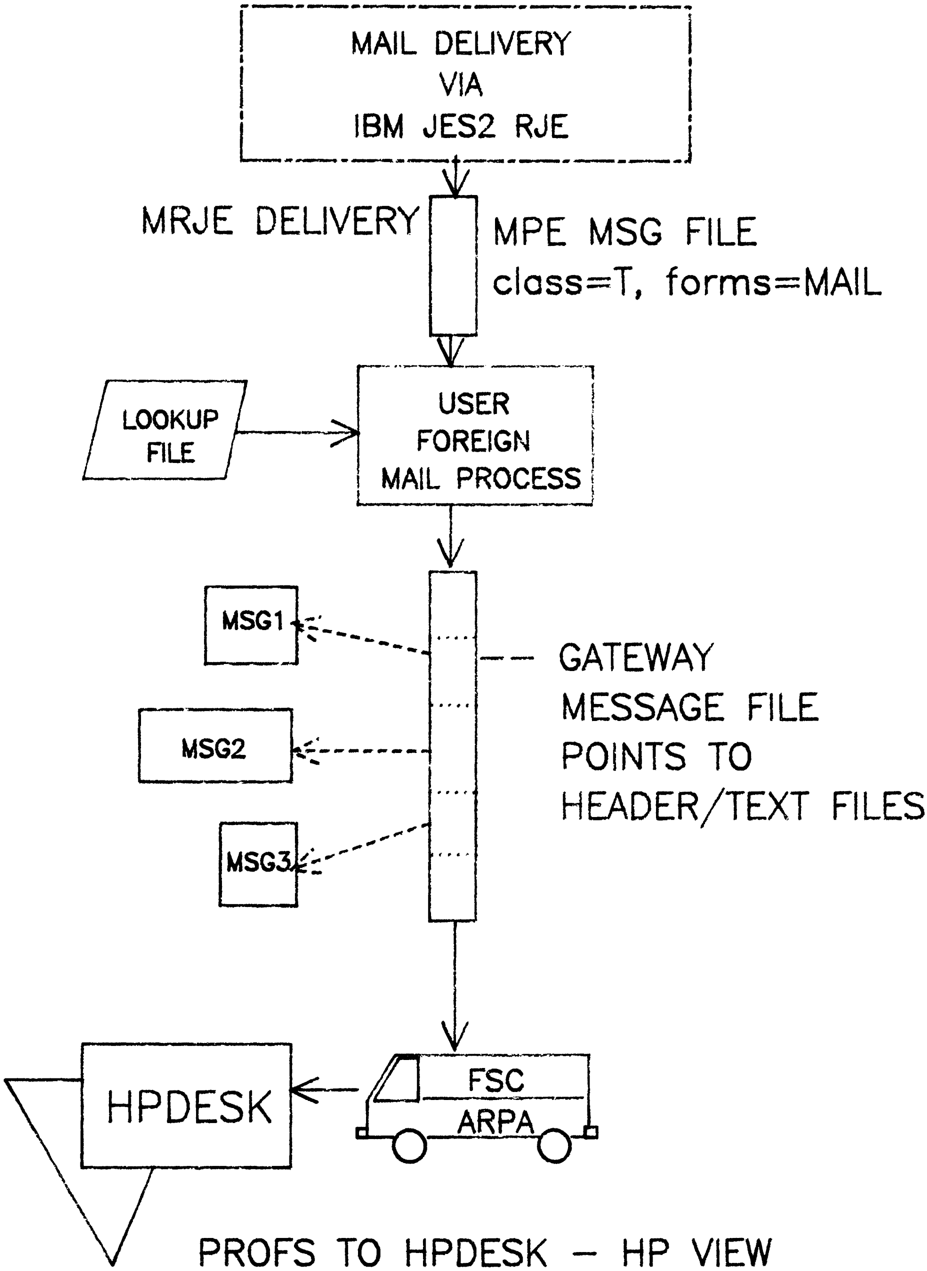
When all messages have been processed and are ready for delivery to HPDESK, a special Foreign Service Connection mail truck program (FSCARPA.HPMAIL.SYS - provided by HP) is activated. FSCARPA inserts mail from the files created above into the HPDESK database as if they had come from a normal HPDESK user (although there is an extraneous part to the message containing the text MESSAGE HEADER!).

If FSCARPA cannot deliver a message (e.g. unknown username) an error message is sent to the HPDESK "default user" specified by the mail administrator in the DATA Menu selection of the Network Menu.

4.0 MAIL EXCHANGE WITH OTHER SYSTEMS

4.1 THE VARIOUS SYSTEMS

The expansion of mail exchange to other types of systems is a requirement that increases the complexity of our environment substantially. The ability to transform different sets of username and destination formats as well as message formats is required. The running of separate jobs for each system type in the network would create unacceptable overheads so we are using process handling techniques for sharing code, files and general efficiency. The lookup file is also expanded in function to handle the different cases expected.



PROFS TO HPDESK - HP VIEW

Figure 6

4.0 MAIL EXCHANGE WITH OTHER SYSTEMS (cont'd)

4.1 THE VARIOUS SYSTEMS (cont'd)

The different systems we currently exchange or expect to exchange mail with are:

- SYSTEM PRINTERS
- HP3000 HPDESK
- IBM/TSO
- IBM 8100
- WANG OFFICE
- DEC DECMAIL (ALL-IN-ONE)

4.2 HPDESK - TO - PRINTERS

This is outgoing mail only. The lookup file designates the device on which the mail will be printed and information for a delivery header. This technique is not generally recommended for confidential material.

4.3 HPDESK - HPDESK

There are several HP3000 systems connected to the IBM central computer via MRJE that are not connected to each other via DSN/DS. There are no reasons why the same general strategy for exchange with PROFS will not work in this situation. In fact, there are advantages in that the ARPA header format can be maintained and almost used directly as produced by HPDESK.

4.4 HPDESK - TSO

This exchange only works from HP to TSO (to date). The JCL encompassing the HPDESK message invokes the TSO batch program and runs the Electronic Message Board Routing System program which allows TSO users to exchange short notes or files.

4.5 HPDESK - 8100

BCSC has developed an electronic mail system for IBM 8100 computers running the Distributed Processing Program Executive operating system. The exchange of mail is almost identical to PROFS except the mail is delivered to the 8100 as output from a file copy utility (e.g. IEBGENER) instead of an NJE job stream.

4.6 HPDESK - WANG OFFICE - DECMAIL

Most WANG and DEC systems are connected to the IBM central system by RJE links so the same techniques should be usable for transporting mail. At the time of writing our experience with the DEC ALL-IN-ONE and WANG OFFICE systems is limited and no interfaces have been designed. Initial discussions indicate that a more comprehensive exchange format than used between HPDESK and PROFS is desirable.

We also have projects scheduled to investigate direct data transfer using an X.25 network which could eliminate the need for an intermediate IBM system.

5.0 OTHER CONSIDERATIONS

5.1 ADMINISTRATION

There is a requirement for well-defined administrative procedures for handling directory maintenance and co-ordination between the various systems involved in mail exchange. This increases with the number of separate office systems participating in the mail exchange.

Each system will require a distribution list of the other system administrators. Whenever a new user is added to a system a decision is made as to which other systems must know about the new person. The procedures for each system would be different. For example, HPDESK administrators would be required to enter users in both the lookup file and the HPDESK database. The PROFS administrator need only add a user to the system nickname file.

When a user is added, the HPDESK administrator would send a message to foreign system administrators notifying them of the new name and location. They would be required to create a corresponding name in their system and notify the HPDESK "home" system of the foreign name given. The lookup file could then be updated with the corresponding foreign names. The lookup file might have the following information for HP user PAUL WILSON:

HP USERNAME	HP MAIL NODE	PROFS USERID	SNA NODE
PAUL WILSON	HP/01	PWILSON	BSCS02
		RPW	FIN01
		PAULW	LPH01

Careful co-ordination of name entry on each system is required to avoid loss of mail due to an incomplete route. Technical solutions involving acknowledgements and error handling are required to maintain system integrity but the administrative procedures for adding, deleting and changing usernames will play a large role in the success of the mail exchange network.

5.2 RELIABILITY

There is a requirement for system-to-system acknowledgement of the receipt of messages. A first step in this is for the HP system to record an audit entry into the lookup database. This will allow the retention of the external mail file until acknowledgement is received. The procedure will require programming on each system to generate these acknowledgements.

Without an acknowledgement technique there is always the potential that an MRJE failure, a problem with the central system or the interface to PROFS could cause delivery to fail. The protocol between systems should also handle negative acknowledgement messages to indicate error conditions such as "unknown user".

The ARPA format used by HPDESK allows for an incoming foreign message to give an acknowledgement to a message previously sent from HPDESK. This could be used as the basis for system-to-system acknowledgements.

5.3 SECURITY

Once HPDESK exports mail outside of the database, a certain amount of security is lost. This problem is compounded when mail is transported using MRJE.

The use of encryption/decryption would probably have to be implemented within HPDESK and PROFS themselves since user interaction on transmission and receipt would be required for encryption/decryption password input.

A second best alternative is to have the import/export systems at each system encrypt between each other.

The problem could be somewhat alleviated if the MRJE system was not used as the transport mechanism as discussed in the next section.

5.4 TRANSPORT MECHANISMS

A common facility to directly connect to other systems and exchange mail is preferable to using an IBM system as a store and forward message switch. The use of X.25 in a similar manner to HP's DSN/DS is certainly feasible with current DEC VAX/VMS and WANG VS systems.

With IBM systems, the use of IMF as a transport mechanism to IBM systems could be more flexible, more secure and more efficient, however, it would require development of either a VTAM, TSO or CICS application to receive and deliver mail. This system would have to use another transport (perhaps RJE/NJE again!) to exchange data with other connected systems.

A definite MRJE limitation is the transliteration process from EBCDIC to ASCII. In fact other ASCII-based systems may also incur an input translation from ASCII to EBCDIC. The use of special characters for delimiting headers text lines, etc. is currently avoided.

5.5 DOCUMENT TRANSFER

There is some requirement to exchange documents in "word processor" format, for example, conversion of HPWORD documents to DISPLAYWRITER, DCF etc. so that they can be printed, edited or stored in a DISOSS system. These aspects of inter-system office communications require both an accepted document format standard and a large commitment in resourcing for program development.

Another example would be for HP users to be able to receive copies of documents processed by either the IBM SCANMASTER or the WANG PIC systems.

5.6 FOREIGN MAIL FORMATS

Receiving foreign mail via an RJE file requires that there is some method of separating messages and determining from which system they originate, especially if more than one type of system is sending mail. The ARPA format is capable of handling all the situations we have encountered, however gaining consensus from other systems' support personnel to agree to implement a particular format takes some negotiation and time. Consequently, we currently translate a very simplistic (and limited) incoming format to ARPA format. (Appendix D).

An improvement would be a start record indicating the format following. This would at least allow one system with the development resources to interface to a variety of mail systems. Of course, this method would perpetuate different formats on all machines and increase the overall development load substantially - an old story.

Perhaps a new standard such as proposed in the CCITT Draft Recommendations on Message Handling would be more universally acceptable, and implemented by the various vendors!

THE LOOKUP FILE — APPENDIX A

This file is used to transform HPDESK usernames into userids on foreign systems and vice versa. It is also used to translate HPDESK remote location/sub-location designations to remote system node addresses.

The main use for this file is to make the addressing of messages transparent to the user. The objective is for users to only "know" their own systems, users on other systems will be addressed in the same manner as local users. Although HPDESK II gives the user the ability to specify information for the foreign system, we felt this was better handled by the system and the mail administrator for ease of use, reliability of delivery and change control.

The prototype file is a simple KSAM file that keeps HPDESK user names and locations and corresponding PROFS userids and SNA nodes. The same record format is used for both incoming and outgoing translation.

A new lookup file design is implemented as an Image database. This allows more flexible maintenance access and additional functionality for logging, conversion data, error recovery, multiple systems, statistics, variable length usernames, mail administrator names, etc.

The lookup file concept allows flexibility in overall system design. For example, only one gateway file is required for many remote destinations. Since the addressee's name and destination/sub-location can completely define the reformatting process required.

BCSC's system uses sub-locations to match each foreign destination and one gateway file. All foreign deliveries are currently made using ARPA format.

The practical considerations involved in this design was the reduction of the number of jobs required to handle several different foreign system types and the complexity of monitoring more than one gateway file.

The use of multiple sub-locations allows resolution of similar names, although it complicates the job of the mail administrator.

The following listing show a sample configuration and job stream to allow the link to PROFS to receive mail on MRJE printer number three. The host system must have printer three set up as having forms MAIL, train 0, and class T output. Mail is re-directed from the printer to the file PROFSIN.IPC.HPOFFICE where it it picked up by the link.

MRJECON

```
 1. HOST MACHINE=
 2. PSEUDO CONSOLE DEVICE=054
 3. PSEUDO LINE MONITOR DEVICE=052
 4. SIGNON CARD IMAGE=
    /*SIGNON                    REMOTEXXX
 5. HOST SYSTEM=JES2
 6. USER HOST COMMAND=$D
 o
 o
 o
19. MRJE INPUT DEVICE="MRDR1"
20. PRINT OUTPUT DEVICE="LP"
21. UNSOLICITED PRINT DISPOSITION, PRINTER 1="LP"
22. UNSOLICITED PRINT DISPOSITION, PRINTER 2="LP"
23. UNSOLICITED PRINT DISPOSITION, PRINTER 3=!PROFSIN
24. UNSOLICITED PRINT DISPOSITION, PRINTER 4="LP"
 o
 o
 o
35. UNSOLICITED PUNCH DISPOSITION, PUNCH 7="LP"
36. OUTPUT PROCESS PRIORITY=CS
37. NUMBER OF CHARACTERS TO TEST IN JOBNAME=8
 o
 o
42. MAX TIME JOB REMAINS IN JOBLOG AFTER TRANSMITTAL=
43. HOST BUFFER SIZE (BYTES)=400
```

MRJECON

```
!JOB MRJE, MANAGER.SYS
!COMMENT
!COMMENT          JOB STREAM FOR SAMPLE MRJE HOST
!COMMENT
!COMMENT          PRINTER 1 -- CLASS A,J.P
!COMMENT          PRINTER 2 -- CLASS S
!COMMENT   PRINTER 3 -- CLASS T --- PROFS INCOMING MAIL
!COMMENT   PRINTER 4 -- CLASS F
!COMMENT   PRINTER 5 -- CLASS O
!COMMENT   PRINTER 6 -- CLASS N
!COMMENT   PRINTER 7 -- CLASS E
!FILE PROFSIN=PROFSIN.IPC.HPOFFICE,OLD;LOCK;MSG;ACC=APPEND
!RUN MRJEMON;INFO=" "
!EOJ
```

HPDESK II will deposit foreign mail into MPE files in several formats selectable by the mail administrator. These formats are:

1. HPDESK Internal
2. ARPA
3. ARPA Reference
4. ARPA Compressed

More information about these formats can be found in the "Programmatic Access to HPDESKmanager Files" manual.

The ARPA format is outlined here to assist with the reading of this paper, but is not meant to be definitive.

An ARPA format file may contain several messages, all separated by two pairs of carriage return, line feed (CRLF) characters.

The header has a set of fixed field labels, date, name, location and reference name formats. The text portion is a copy of the usual HPDESK mail message including distribution list, subject, etc.

Each line in the header and text portion is delimited by a CRLF. Lines cross 80 byte record boundaries.

The ARPA header only contains addressees at the same location. The message text will contain the full distribution list. Another header field, "X-HPDESK-ACK:" is possible for the receiving system to send an acknowledgement.

```
FFD

Date             :          12 Dec 84 1613GMT  CRLF
Sender           :     "Dick Battles"@  DSD/01.GATE.REF CRLF
From             :                                      CRLF
To               :          "Will Hopkins"@  PROFS/01   CRLF
                 :          "Barry Zilkie"@  PROFS/01   CRLF
cc               :          "Chris Slade"@   PROFS/01   CRLF
BCC              :          "S Reeder"@  PROFS/01   CRLF
X-HPDESK-ID      :    5432112 12345 6789 4 "PROFS/01"   CRLF
X-HPDESK-PRIORITY:    2                                 CRLF
CRLF                                      :
               text
                o
                o
```

FFD - "formfeed" character - starts each message.

CRLF - carriage return, line feed characters
- separates each line in header and text segments
- two CRLFs separate header from text portion

JCL AND MESSAGE FORMAT (Prototype)

```
        //HPMAIL  JOB (12345, 6789,A,X,Y)...
        /*XEQ USERID.NODE                    /*XEQ
        PROFS-MAIL-HEADER-1
        PROFS-MAIL-HEADER-2
            - mail text-
            - 80 byte records-
            - 1 per text line-
            - 1st column blank
                 .
                 .
                 .
    /*
```

PROFS EXTERNAL MAIL HEADER FORMAT (Prototype)

```
CARD 1:  col 1......%%
         col 3......blank
         col 4......up to 8 character TO userid
         col 12.....blank
         col 13.....up to 8 character TO node id (JES NODE)
         col 21.....blank
         col 22.....up to 8 character FROM node id
         col 30.....blank
         col 31.....up to 8 character FROM node id (JES NODE)

CARD 2:  col  1.....hex 'FE'
         col  2.....MSG:FROM
         col 11.....blank
         col 12.....up to 8 character of FROM userid
         col 20.....--
         col 22.....up to 8 character of FROM node (JES ID)
         col 31.....TO:
         col 34.....blank
         col 35.....up to 8 character of BASE TO userid (note this
                    the TOserid to whom the mail was sent.....do not
                    use this field for determining the recipient
                    but use the TO userid in CARD 1.....if this mail
                    is being sent as a COPY (.CC) the actual userid
                    is in CARD 1 and this id is the single TO
                    userid.
         col 43.....--POSTMAN
         col 63.....DD/MM/YY
         col 71.....blank
         col 72.....HH:MM:SS
```

MNEMONICS REFERENCE

APPENDIX E

BCSC	British Columbia Systems Corporation
CICS	Customer Information Control (IBM)
DPPX	Distributed Processing Program Executive (IBM)
HP	Hewlett Packard
HPDESK	HPdeskmanager (HP)
IBM	International Business Machines
IMF	Interactive Mainframe Facility (HP)
IMS	Information Management System (IBM)
JCL	Job Control Language (IBM)
JES2	Job Entry System 2 (IBM)
MRJE	Multileaving Remote Job Entry (HP)
MVS	Multiple Vitrual Storage (IBM)
NJE	Network Job Entry (IBM)
PROFS	Professional Office System (IBM)
RSCS	Remote Spool Communication System (IBM)
RJE	Remote Job Entry (IBM)
SNA	Systems Network Architecture (IBM)
TSO	Timesharing Option (IBM)
VM/CMS	Virtual Machine/Conversational Monitor System (IBM)

BIOGRAPHY

Paul Wilson is a product specialist with Brisith Columbia Systems Corporation in Victoria, B.C. Canada with responsibility for technical support and product evaluation for HP3000 systems. Paul has worked with BCSC for six years with experience in technical support for Honeywell 6000, IBM 8100/DPPX and VSAPL systems. Prior to working with BCSC, Paul was a systems engineer with Honeywell Information Systems. He graduated with a BMath in 1970 and MMath in 1971 from the University of Waterloo, Waterloo, Ontario, Canada.

OP - OPERATIONS MANAGEMENT

SHADOW - PROVIDING USERS WITH 100% UPTIME

Ian Campbell

CAROLIAN SYSTEMS INC.

Toronto, Ontario, Canada

ABSTRACT

This paper discusses a new method, now available to the HP3000 community, of keeping data from Image data bases on two or more systems up to date. Included is a discussion of the system logic, user interface requirements and performance requirements of this new facility.

INTRODUCTION

In any HP3000 site there are numerous planned and unplanned occurences that impact the availability of application software to users. These can include planned periodic backups as well as hardware and software failures.

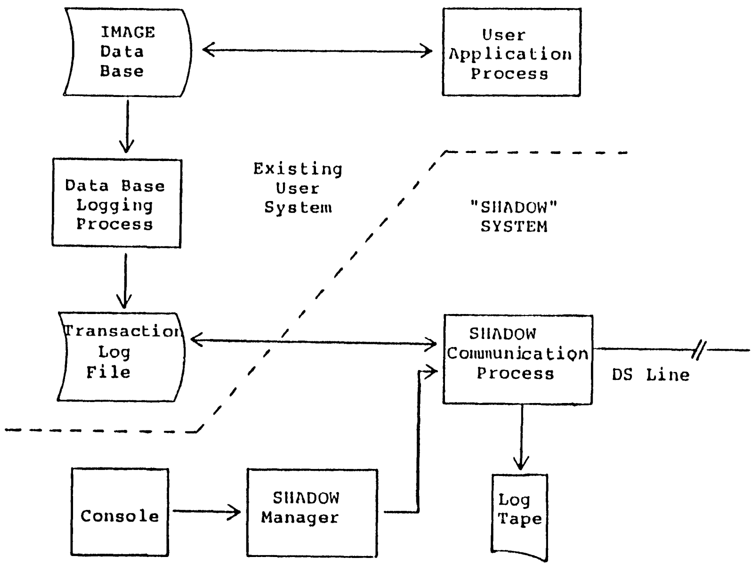
Unfortunately for those organizations which require constant availability of critical applications, <u>any</u> form of processing interruption can be considered as nothing less than disasterous.

A solution to the requirement for <u>total</u> user availability can be provided via a replication or "shadowing" of applications on a second HP3000. Once replicated on a second or on multiple machines, the applications can be kept current and available to any user in the event of a failure of the primary system or of the communications link between systems.

PRIMARY SYSTEM LOGIC

Since SHADOW software must reside on both the primary and the backup systems, the diagrams and functional flow for each system can best be illustrated separately. The following illustration is representative of the primary system containing the data base to be SHADOWed.

PRIMARY SYSTEM



The point at which the SHADOW system can interact with the user's data records is after they have been logged in a transaction log file. Once the data base transactions have been logged, the Communications process directs the flow of data from the transaction log file to the DS line. In addition to this responsibility, the Communications process must also be prepared to receive requests from a Control process which acts as the interface between the user and the SHADOW system. The Communications process must also ensure that transactions are logged to tape in the event that a prolonged line failure occurs.

The SHADOW system periodically examines the transaction log file for new transactions from the primary data base. These records are then assigned a sequence number and forwarded to the remote system. For reasons of line efficiency, the records from the log file assembled into larger transmission blocks.

During the process of forwarding transaction records to the backup system, the returning sequence numbers are examined to ensure that all records that were sent to the remote system have actually been received by it without any errors or omissions.

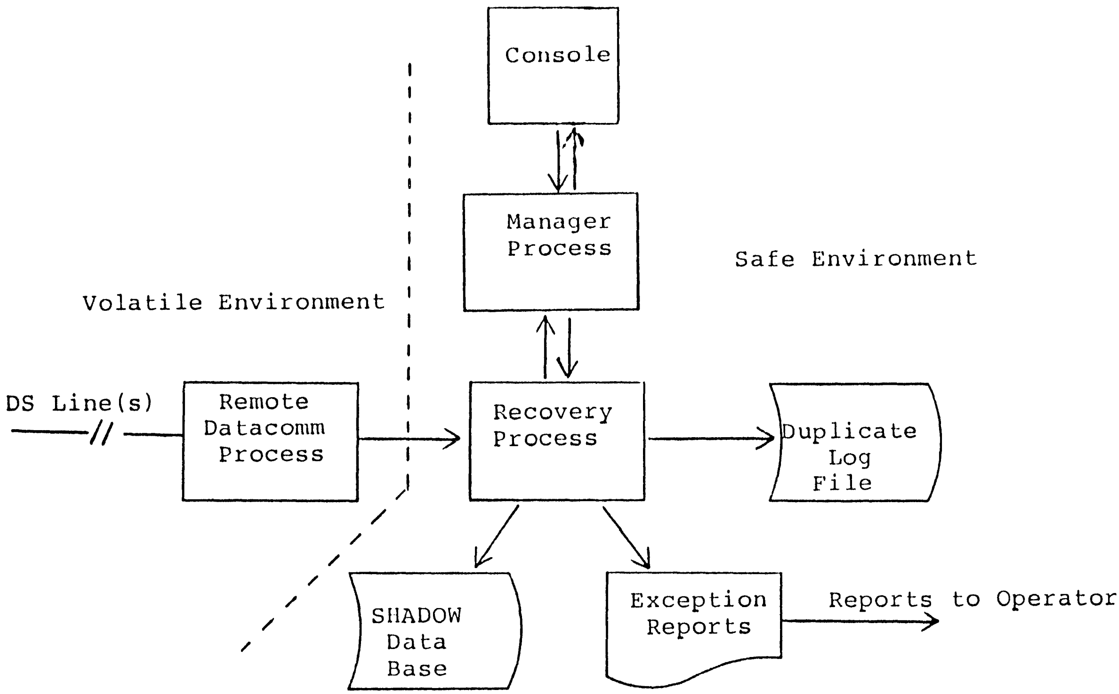
Although the diagram on the previous page makes reference to only one DS line, the SHADOW system can make use of more than one line concurrently. Traffic is spread evenly over the lines when more than one line is available. Should one of the lines become unavailable, the traffic is then spread over the remaining lines. If the only remaining line becomes unavailable, the SHADOW system gathers the records that have accumulated in the log file and forwards them to the remote system. In this instance, the backup tape is not needed as the natural flow of data causes the remote data base to catch up with the active system.

The purpose of the backup is to hold the data base transactions in the event that the main system suffers a fatal hardware failure while the communications facility was unavailable or the remote system was down. If this situation exists, the tape is required to bring the system up to date, as the disc log file may have been destroyed during the failure. If a tape is required for logging, a separate process is spawned to handle I/O to the tape. This prevents the Communications process from hanging on a console request for the tape.

REMOTE SYSTEM LOGIC

Once the transaction records have been assigned sequence numbers and assembled into transmission blocks, they are forwarded to the remote system where a datacomm process passes these blocks to an intermediate data capture process. When the data has been given to this process, the block sequence numbers are returned, thus ensuring that no blocks were missed during transmission. When the blocks have been handed to the intermediate process, they have been safely SHADOWed because the process runs as a separate JOB on the system. If a line failure occurs, the datacomm process and its environment are assumed to have terminated. However, the data records written to the intermediate process are still available to be used for Data Base updating. This logic is illustrated in the following diagram.

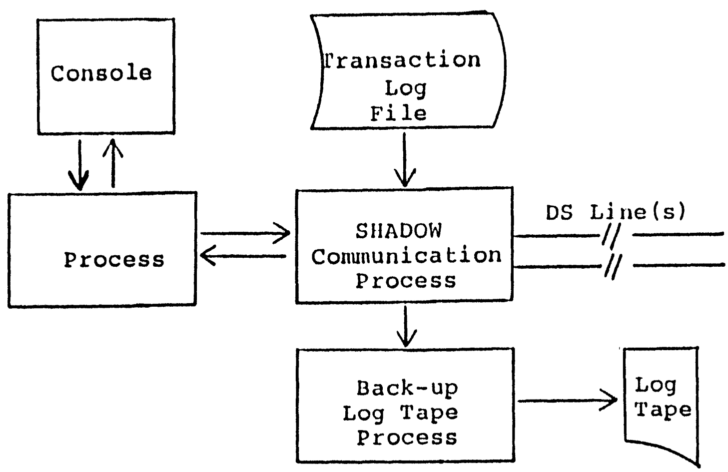
Remote System Logic



USER INTERFACE

Complete control of the replication process as well as switching, backup and status requests must be provided via an easy-to-use, command driven user interface. The SHADOW communication process is responsible for handling any request from the user interface or user Control process. All commands and requests are entered through the Control process and are sent to the communications module via a message file. This architecture permits the Communications process to run continuously without being impeded for any reason. This structure also allows the Communications process to be constantly copying data from the Image log file to the DS line, while also listening to the Control process for requests. The diagram below is a logical representation of the relationship and data flow surrounding the user interface and the Communications process.

Communications Process



To ensure the smoothest and easiest control of the SHADOW system, it is best to employ a totally user-configurable interface. The configuration file used by SHADOW is a standard MPE editor file containing the data base name, log file name and other pertinent information regarding the system. This design permits changes to be made to the configuration file quickly and easily. Changes to the configuration file are not allowed while it is being used for an active system.

This design also provides for multiple versions of the SHADOW system to be running concurrently with different applications, as well as with multiple backup systems. The design also permits commands to be passed to the system at run time. This feature assists in the implementation of user-defined commands which can be of even further assistance to the SHADOW user.

The SHADOW system may be operated from any terminal on the system provided the user has the MPE capability of System Supervisor. Access to the system is obtained by running the SHADOW Control program and entering the appropriate commands. For simplicity, eight main commands are used to supervise and control all of the system parameters and activities.

1. SYSTEM

This command is used to identify the specific data base system for which all subsequent commands will apply. The file name specified is the configuration file for the application being SHADOWed. It contains information such as which DS lines are to be used, how frequently to examine the log file, the transmission block size and other pertinent configuration data.

2. START / STOP / KILL

These commands are used to start and stop the SHADOW backup system. If the system was not previously specified with the SYSTEM cpmmand, it must be specified when starting and stopping. A STOP request will stop the SHADOW system in an orderly manner by not allowing new transactions to be initiated while waiting for all outstanding ones to complete. When this occurs, all the remote processes are shut down, the lines are closed and the main process terminates. A KILL request does an immediate shutdown of the main SHADOW system without regard to transaction status. Should this command be issued, it is recommended that a SWITCH command be issued on the remote system to initiate recovery.

3. USE / RELEASE

These commands are used to obtain additional DS lines for system use in the event that they were not specified in the configuration file used when starting. Any line already in use may be released from SHADOW service. This command will not release the DS line if it is the only one available. The USE command is also used when there has been a communications line outage. When the line fails and the cause has been determined, it may not be necessary to SWITCH to the backup system. Here, the USE command can be employed to inform SHADOW that the line is now available and can be used to continue backup activity. SHADOW will then catch up with all the records that were delayed pending line restoration.

4. STATUS

This command produces a display of all the pertinent information about the system specified in the SYSTEM or START commands. The response contains current information such as

a) the number of lines in use;

b) the log examination interval and current log position;

c) the number of outstanding entries to be sent as well as the number of entries successfully transferred;

d) error conditions that have occurred.

5. BACKUP / RESUME

These commands allow the system user to inform the SHADOW system that a backup is to be performed on the remote system. In such an event, no further transaction entries will be sent to the remote system (other than those which have been initiated). When all traffic has come to an orderly halt, a signal is sent to the remote system requesting a release of the data base. If there are any transactions in progress, a backup will not be permitted using this method. It is the resposibility of the system user to identify this situation and correct it. The RESUME command may then be used following a backup to resume the normal SHADOW updating of the data base. Unless a main system failure occurs, this will take place by the routine "catch up" ability of the system.

6. **SWITCH**

This command is used on the remote or SHADOW system to indicate a request to transfer all activity to the backup system and to initiate recovery procedures. A report of all complete transactions is provided to inform the operator of the current data base status. Once these reports have been generated, the SHADOW system has completed its task and will terminate. It is possible to restart the system in the opposite direction with the "backup system" now serving as the "primary" system.

7. **CONFIG**

This is a request to display the contents of the current configuration file.

PERFORMANCE REQUIREMENTS

Performance requirements are best discussed in terms of the availability and response time of the SHADOW system. One of the more obvious requirements is to minimize the time required to SWITCH over to the backup system in the event of a main system failure. Since one of the responsibilities of the SHADOW system is to be constantly aware of the last complete transaction for any data base user, the amount of time required for switching (for software) is in the order of seconds. The actual transfer time will depend on the methods used to switch the terminals to the backup system and how expediently the users are informed of the data base status, or their own transaction status.

The SHADOW system makes every attempt to utilize the facilities available to it in the most efficient manner possible, and, as a result, it does not cause performance degradation for other system users. Wherever possible, the system assembles data into large transmission blocks and does as few I/O operations and data movements as possible.

While the primary function of the SHADOW system is to maintain an up-to-date duplicate copy of an Image data base on a second computer system, there are a number of other beneficial capabilities which are part of the integral design of the system.

These capabilities include

1. the ability to use more than one DS line for redundancy and load balancing for performance reasons;

2. the verification of operator requests and input data. Items such as configuration file design errors or omissions are reported as such;

3. status of both systems is constantly available to the user for close monitoring of activity. Transaction status is available to ensure correct startup in the event of a switch over;

4. transmission integrity is maintained between both systems by the use of record numbering. This will ensure that no records are duplicated or omitted;

5. although this paper discusses the SHADOWing of one data base, the product may be used for applications where multiple Data Bases are accessed by any one transaction. The backup system used may also be located either locally or remotely. As well, the backup system need not be a SHADOW or a duplicate hardware configuration of the primary system;

6. user commands are kept to a minimum and most commands have only one parameter. This reduces the complexity of the operation and reduces the likelihood of user errors;

7. the remote data base can be made available for READ access while the primary data bases can be allocated for updating. This results in improved load balancing for the systems;

8. the remote data base can be used for backups, thus offering 24-hour uptime on primary systems.

AUTOMATED SCHEDULING OF COMPUTER OPERATIONS -- A CASE STUDY

Barrie W. Day
Northern Telecom
Santa Clara, California, USA

SUMMARY

In the past five years, sales of Northern Telecom's SL-1 Business Communications Systems division have tripled. The dramatically increased volume of our business coupled with bottlenecks in computer processing and hardware availability were a major challenge in the company's SL-1 Business Communications System Division at Santa Clara, California.

In August, 1983, we began to analyze and find ways to optimize our hardware performance with the help of Hewlett Packard (HP). From the beginning, HP's role was to provide advice and recommended solutions to a problem of throughput in the batch processing cycles.

Cooperation between Northern Telecom and HP, along with a concentrated effort brought success. The following description of our combined efforts includes explanations of our strategy, optimization tools, and implementation.

BACKGROUND: STRATEGIES TOOLS AND POST-IMPLEMENTATION REVIEW

The dramatically increased volume of our business coupled with bottlenecks in computer processing and hardware availability were a major challenge for our Business Communications Division of Northern Telecom, manufacturer of PBX's.

There was a critical need to equip the Computer Operations function with a sophisticated tool to improve management, control and performance.

In August, 1983, we began to study the entire scope of the problem and identify an approach that would lead to optimization of our hardware performance. We involved Hewlett-Packard from the beginning, with their role being to provide advice and recommend solutions to a problem with throughput in the batch processing cycle. We provided problem management, and monitoring so that HP representatives were aware of both our weekly progress and problems requiring their immediate and long-term assistance. Cooperation between Northern Telecom and HP, along with a concentrated mutual effort to produce results, brought success. The following description of our combined efforts includes explanations of our strategy, optimization tools, and post-implementation review.

In the past five years, sales through Northern Telecom's SL-1 Business Communications Systems Division had tripled. Data processing operations had progressed from a few batch applications running on the HP3000 Series III computer to several new, on-line applications with two HP Series 64 computers, one HP3000 Series III, two HP1000's and an IBM 4341 operating through a Northern Telecom office controller PBX. During this period, we had automated user areas, installed more sophisticated applications software, and purchased newer equipment.

However, in June 1983, our users insisted that the data center hardware be available for user-sessions from 6:00 a.m. to 12:00 a.m. each day. This substantially shortened the time we had to process batch production, from 14 hours to 6 hours per day. Also, the increase in sales volume and the addition of several new on-line applications resulted in a major increase in processing. We were rapidly losing our delicate balance of control.

We soon realized that management information systems would have to address a number of problems:

- The short time available to complete daily production,
- Maximization of computer equipment and peripherals use,
- Improving our ability to monitor and plan system use,
- Development of detailed and reliable operations statistics,
- Minimization of human error, duplicated effort, and job re-runs,
- Dealing with the number of interrelated applications that could be run only by operators familiar with their particular processing requirements,
- Justification of hardware expansion.

With these problems in mind we began the search for alternative solutions. The initial analysis revealed several areas of focus:

- Application systems architecture and design,
- The volume of transactions, and the timing of peaks,
- The market shift to a merchandise product requiring increased manufacturing flexibility, responsiveness, and consequently unplanned and unscheduled use of computer resources,
- Even with the implementation of several on-line systems, users were still requesting computer reports causing heavy processing requiring additional operations personnel to deal with the volume of paper,
- Business expansion had increased the window needed for session access which unfortunately reduced available time for batch processing,
- Additional hardware/Disc Cashing.

All of the above were analyzed in detail however with the frequent occurance of re-runs, high error rate and personnel turnover this led us to concentrate on the Operations Schedule initially. This was at the time, a tedious manual process that was also not well defined or documented. Yet, computer operations had continued to follow manual procedures to meet each day's processing requirements. Most managers of the computer operations department felt that our manual system was relatively efficient, although time consuming and hard to manage. Since most problems could be solved by adding personnel or working overtime, we hesitated to devote the time, effort, and expense necessary to improve efficiency. The lack of a clear-cut method of automating operations, the risk associated with change, and the inability to define a concrete payback had relegated data center automation to a low priority.

It was soon very clear that this element of the problem appeared to provide the biggest single impact on improving overall performance at the least cost.

At that time, a variety of Automated Scheduling software packages were available for IBM DOS and OS systems. There were also a number of small, HP3000 utility programs available in-house and through the contributed library. But none addressed enough functions to be of real use. In addition, we were reluctant to base the success of our data processing operation on software that had little or no support.

In 1983, an Automated Scheduling System was introduced at the HP3000 International Users Group Conference in Montreal. Although the product was new, its high payoff potential and apparent sophistication, as well as the complete and high-level support led us to evaluate its applicability on a trial basis along with other alternate software.

The task of developing the implementation plan and installing the software was undertaken by a team consisting of our operations manager, our systems manager, our lead operator, and a customer support specialist provided an initial "set-up" package consisting of a pre-site survey (which allowed them to configure the software) and an installation planning checklist (which allowed us to allocate and plan the necessary personnel and equipment resources for the project).

During training, a model data base representing an HP3000 installation provided a practical example of how the software should be used, and a number of utilities were provided to automate the "set-up" process at our site.

During actual installation, we collected and organized all of the information previously stored in everybody's heads. Our experience also stimumated rethinking our JCL structure. Later on, when we actually began using the product, we devoted ourselves to the additional effort needed to accelerate the conversion and to eliminate manual scheduling entirely.

The Automated Scheduling System, now in place, proved straightforward to understand and to implement. We have been rotating operators so that they are all comfortable, and reaction has been enthusiastic. The System is seen not as a threat, but as an advancement. The system doesn't eliminate operators; it makes their jobs much easier and improves morale. The table reveals a number of major time savings.

Previously, we were running the most difficult jobstreams sequentially; the System lets us run them concurrently. We have realized more than a 30 percent savings in our nightly batch processing time simply by increasing the number of jobs processed concurrently. With the job limit set to 10 on our Series 64, the jobs are launched automatically while managing all the job dependencies and resource sequencing. We also have achieved major reductions in processing by reorganizing the way we process our jobs and by smoothing out the volume of transactions.

The facts speak for themselves. Previously we were beginning the nights production at 4:00 or 5:00 p.m., and concluding at 8:00 or 9:00 a.m., a shift of 15 to 17 hours. We now process the same amount of data, with more complete reports and dramatically improved service, in 6 or 7 hours.

Cutting down the time required for nightly production has meant better service to users. In the past, we heard the same question again and again, "When can we use the system?" None of the users seemed to realize the time problem we faced. After the transition, users were astounded when we told them that they could have access from 6:00 a.m. to midnight. In addition, we consistently averaged 95 percent or better system uptime for our users.

Other indirect benefits derived include the reduced demand for staff involvement in routine tasks. Automated scheduling and job dispatching reduced the need for operator involvement in production setup and processing.

Before we implemented the system, our most time consuming jobs were repetitive, and things tended to happen the same way every night. Now, 70 to 80 percent of our production runs automatically. Decisions on unscheduled jobs can be planned ahead of time or at the front end of the schedule, not at 3 or 4:00 a.m. Now we know what happened the night before, and we go right to the core of any problems. This has improved system integrity, eliminated reruns, and improved our service to users.

Another key success factor has been the system's enforcement of data center standards and procedures. By instituting a complete system of procedures, we have almost eliminated the effects of absenteeism on our production schedule, and we have made the entire department more efficient. The Automated Scheduling System provided us with a daily plan showing the work flow through our data center highlighting instances of operator intervention.

The entire process allowed us to regain control of Computer Operations and at the same time improve our overall service level to the user community.

TYPICAL AUTOMATED SCHEDULING
vs
TYPICAL MANUAL SYSTEM

ESTIMATED MAN-HOUR SAVINGS

ACTIVITY	CURRENT HRS/DAY	ANTICIPATED HRS WITH OCS (% Savings)	ESTIMATED SAVINGS/DAY
Production Scheduling	2.0	.25 (88%)	1.75
Coordinating User Requirements	2.0	.5 (75%)	1.5
JCL Preparation	2.0	0 (100%)	2.0
Nightly Operator Processing	10.0	7.7 (23%)	2.3
Job Status Tracking	2.0	0 (100%)	2.0
Forecasting and Managements Reports	.5	0 (100%)	.5
Report/Media Distribution	1.5	1.0 (33%)	.5
Tape Library Management	.5	.25 (50%)	.25
TOTAL DAILY HOURS			10.8
TOTAL ANNUAL HOURS (250 work days per year)			2700
TOTAL DOLLARS $15.00/hr including salary, overtime, benefits & overhead			$40,500

Biography

Mr. Day is Director of M.I.S. at the Business Communications Division of Northern Telecom in Santa Clara, California. In addition, he has held senior management positions with Northern Telecom in other parts of the U.S. and Canada. He is also on the board of several computer/software related companies.

Prior to this Mr. Day held various management positions in the U.S. and overseas in the computer industry and other high technology fields with several international electronics corporations including Honeywell, ITT and N.V. Philips.

Mr. Day is a member of the Operations Research Society, the Data Processing Management Association, Association of Systems Management, and the Institute of Computer Machinery.

Mr. Day has published several technical articles on Data Base design and has lectured at graduate institutions in both the U.S. and overseas.

He is a graduate of Imperial College, University of London with a BSEE and holds an MBA from the University of California.

Mr. Day speaks four languages and his interests include sailing, skiing and photography.

SCHEDULING FOR BETTER SYSTEM USAGE

Kim Harris

Handsworth Programming Services
60 Hazelwood Lane, London N13 5HB

Summary

The need for more control over batch work than is provided by MPE is recognised by many installation managers. The implementation of such control has developed over a period of four years in the development of our 'Scheduler' product. This paper will describe the meeting of the need for control over ad hoc environments and more traditional batch systems. The problems of having many users, dependancies between jobs and the prevention of conflict are discussed. At the same time, the maximisation of throughput and the equitable sharing of computer resources will be considered.

CONTENTS

1. Scope of the paper

2. The Aims Of The System Manager

2.1 The maximisation of resources
2.2 Production control

3. The HPS SCHEDULER

3.1 History of development
3.2 Current facilities

3.2.1 For maximisation of resources
3.2.2 For production control
3.2.3 Scheduler from the User's point of view

3.3 Facilities under development

1. Scope Of The Paper

This paper aims to perform three functions:

i) To raise some of the problems encountered by system managers of the HP3000.

ii) To discuss in general terms some ideas for the solution of the problems.

iii) To present the facilities offered by the HPS SCHEDULER software to assist in the solution of the above problems.

2. The Aims Of The System Manager

2.1 To maximise the efficient use of system resources

The system manager has a fundamental responsibility to the users and to the institution or company owning the system. To the users, he (1) must make a fast and reliable service available with a minimum of effort on their part. For the company, he must provide the user service at a minimum cost both in terms of computer systems and personnel. For example, the installation may need to run systems overnight without the cost of employing three shifts of operators.

These are typical requirements in order to fulfil this aim of maximum resource utilisation:

a) Background use of available processing time during high demand periods.

b) Fast turnaround of small, urgent, or otherwise high priority jobs.

c) The efficient use of computer terminals entailing the running of non-urgent and 'offshoot' processes in batch. This means that the terminal is not monopolised by a long process.

   It is, of course, possible to perform offshoot processes as multiple Son processes but the disadvantages of this are numerous. Firstly the Father process must continue running and so the terminal is still locked into this process until the Son is complete. Secondly this involves more complex processing, therefore more programming and more maintenance, therefore more cost and more to go wrong. Thirdly the system load becomes more difficult to measure and control.

d) Maximisation of the use of off-peak periods with the minimum of effort for the user. Large or non urgent work should be automatically scheduled for off-peak periods without the user having to think about it. Rewards in the form of 'cheaper' computing (either actual or notional allocation of cost can be used) can be given for running jobs overnight or at weekends.

   A resource accounting package such as our ACCOUNTS software can be used to automate this principle.

e) Operators should be warned of jobs which require special or manual intervention. These jobs, the obvious examples are those requiring tape mounts of private volumes, should not be started until all the resources, media, staff etc are available because otherwise they may hold up processing waiting for a operator reply and thereby leave the computer running idle. This can also help to prevent such eventualities as deadly embrace or the overwriting of files. There will be more about this idea in section 3 of the paper.

2.2 Production Control

In the course of his work, the System Manager also has to perform a Production Control function. His tasks here are likely to include the following:

a) Schedule planning for routine and repetitive running of jobs. It is desirable for the computer to hold schedules and initiate processes automatically.

b) The prevention of conflicts in jobs which may not need to be run in any particular order but must not run together or while some other system is running.

c) The setting up of dependancies within jobs. There may be sequences of jobs which should be run or not run subsequent to the success or failure of a given job or jobs.

d) The provision of information for operators about the processing of a job including special requirements or perhaps something which should be checked before running of the job.

3. The HPS SCHEDULER

3.1 History of development

The site at which Scheduler was developed started in 1978 with a HP3000 Series II,was later upgraded to a Series III, and now has a Series 68. After initial installation it was felt that MPE lacked some essential operational control facilities in four main areas:

i) Enforcing job CPU time limitations

ii) Scheduling of work in time

iii) Notification to the operator of tape requests prior to the starting of jobs

iv) The ability for users to cancel jobs prior to execution (at that time the JOBSECURITY facility did not exist)

Item (i) was felt to be of primary importance because there seemed to be no useful mechanism in MPE (there still isn't) which could be used to enforce certain job CPU resource limitations within specified time slots. At that time we only differentiated between 'Prime Shift' (0830 to 1730) and non-prime shift. Some control could be obtained by using variations of input priorities and the JOBFENCE command but this method is only as reliable as the people submitting the jobs and the installation at that time was catering for many diverse users with varying levels of knowledge all submitting jobs in an ad hoc manner.

Due to our moderately active tape environment, we found that we needed some advance tape notification system, for our operations department, of tape requirements prior to access. It was also realised that if this facility was coupled with a scheduling system we could then overcome the resource conflict that frequently occurred. In order to completely implement such a system, it was necessary to prohibit tape access from sessions.(2)

At that time, no attempt was made to examine the system to see what was running already and schedule work accordingly, the object was merely to prohibit the running of jobs before a certain time if they requested more than a certain number of seconds CPU time. The limit was 300 seconds and the system configured default time limit was set to 300.

Later, as the load on the system grew, it became desirable to measure the work load and then stream work accordingly. Also, we wished to move away from the arbitrary cutoff time and be able to grade work in the same way as is done on many mainframe systems. This was not possible until the release of the 'Bruno' version of MPE because it was not possible to divert the output from commands such as SHOWJOB and SHOWQ into a file for analysis. When this feature became available we were able to move forward without resorting to the use of Privileged Mode to examine the JMAT table. It has always been our policy, and continues to be, not to use PM.

Details of the current facilities in this respect are described below.

During this period other installations had expressed interest in our system and purchased it from us and we were receiving feedback from installations which had rather different requirements from our own.

Until this time, Scheduler had been used in a very unstructured environment with each user largely on his own. Our new users were often production installations where batch work was controlled by the operators and what they wanted was better production control facilities for setting up schedules of routine work and ensuring that certain jobs do not run with certain other jobs. We have been grateful for the continuing input from other sites and it is through them that the product, only relatively recently actively marketed, continues to grow. We are currently on Version 5 of the system and are preparing Version 6 and this paper will now describe the facilities currently in Scheduler and those with which we are enhancing it.

3.2 Current Facilities

3.2.1 Facilities for the maximisation of system resource

The first thing to describe is the way in which the table of parameters is set up by the system manager to control the way that Scheduler handles work submitted to it.

When the software is first installed, it is supplied with a default set of parameters as described in the table below:

```
  JOB SCHEDULER .Version 05.10 .(c)HPS 1984. FRI, DEC 14, 1984,  4:45 P
Command?lp

SCHEDULER (C) 1982 . RESOURCE SCHEDULING PARAMETERS

Session Cutoff Number:  32   MRJE OFF.   Default Time Limit:    60
Default Q: D
INPRI Time limits: 5-32767 6- 5000 7- 1000 8-  400 9-  200

Scheduling Parameters:    Number of periods:  4
             Start   End  £/CS  £/DS  £/ES  JF/CS  JF/DS  JF/ES
Current       800  1745    0     1     2      9      8      7
ON-PEAK:
Period  1       1   759    2     3     3      4      4      4
Period  2     800  1745    0     1     2      9      8      7
Period  3    1746  2200    1     2     2      8      7      7
Period  4    2201  2400    1     2     3      7      6      6
OFF-PEAK:
Period  1       1   759    1     2     2      4      4      4
Period  2     800  1800    1     1     2      7      4      4
Period  3    1801  2200    1     2     2      6      4      4
Period  4    2201  2400    1     2     3      4      4      4

Off-Peak Days Of The Week:
Sunday    Saturday

Off-Peak Dates:   Number:   4
31/12/83  25/12/83  26/12/83  01/01/84
Command?e
```

The table can then be modified to reflect the precise needs of the installation.

To explain some of the parameters in some detail:

Session cutoff number:	The number of jobs after which no jobs are to be streamed.
INPRI time limits:	The system categorises jobs automatically according to the CPU time requested and according to these limits.
ON-PEAK/OFF-PEAK:	Days defined as off-peak in the list of weekdays or dates in the parameters can be treated differently to days defined as on-peak. For example more and larger jobs may be allowed during off-peak days.
£/CS, £/DS, £/ES:	The number of concurrently running jobs in each of the queues.
JF/CS, JF/DS, JF/ES:	The jobfences for each of the queues. This allows you to run large background jobs in the ES queue while only permitting small high priority jobs in the DS or CS queues.

3.2.2 Facilities for production control

Diary facilities are sophisticated in that they allow the user or the person responsible for production control to set jobs to be run on a completely flexible repeat basis of hours, minutes, days, months or the last day or specified day of each month.

At the same time, it is possible to prevent a job from being streamed when specified jobs, or jobs belonging to a particular user or account are running. This is called the Interdependancy feature.

Reporting of schedules is easy and comprehensive with sorted and selective on-line or printer reports available both to the user and the production controller.

Pre-Run Setup Information is available to the system operator allowing him to ensure that resources and media are available to the job before it is started. This prevents the necessity for a job to wait for a tape or drive or some other non-sharable resource and avoids the danger of deadly embrace or jobs clogging the queues.

3.2.3 Scheduler from the User's point of view

The user need hardly be aware of the Scheduler's existence unless he wishes to make use of its interactive facilities. We recommend that the STREAM command is replaced with the supplied UDC and the user can 'stream' his jobs in the normal way. The difference being that he will be told what category his job has been assigned and the identity number which has been allocated to it. If he wishes to examine the queues he can do so. The job can be deleted from the queue at any time up to its actual streaming and the user can change the date and time of running.

Applications software can submit work directly into Scheduler via the supplied interface routines and so the user gets his work done while the system manager can still exercise some control over the number and size of jobs being submitted to the system without inconveniencing the user. Jobs can be pre-processed from templates with JOBLIB/3000 (3) and submitted direct to Scheduler.

3.3 Facilities under development

Currently (December 1984) we are working on a new version with extended facilities for the setting up of dependancy chains between jobs and groups of jobs. It will be possible to determine whether a preceding job has successfully completed before streaming another batch of work and also to stream a different set of jobs if the preceding job has failed. We feel that this will enable Scheduler to be an even greater aid to the automation of processing and production control on the HP3000.

* * * * *

Footnotes

(1) "He" is used in a non-sexist sense throughout this paper to avoid the tedious use of 'he/she', 'his/her' expressions. We, of course, realise that DP personnel may be of either sex.

(2) Quotation from "HP3000 Job Scheduler" by Clive Oldfield, published in the Journal of the HP General Systems Users Group, Fall 1979.

(3) JOBLIB/3000 is a product of Oy Porasto Ab, Helsinki, Finland.

Biography

Kim Harris
has been in data processing since 1976. After spending three years in operations at Imperial College of Science and Technology on CDC mainframs, he joined London Business School in 1978 as Operations Supervisor where he and Clive Oldfield developed 'Scheduler'. Kim is now a partner in HPS and an Associate with a London Firm of Chartered Surveyors where he is the manager of computer systems and software development.

SPECIFIC SERVICES OF THE COMPUTER UITWIJK CENTRUM, IN PARTICULAR ON CONTINGENCY SERVICE LEVELS

J. Kooy
Computer Uitwijk Centrum BV, PO Box 2228,
8203 AE LELYSTAD the Netherlands

Summary:

The number of companies where operations depend to an increasing extent on computer facilities, is constantly growing. If, through a disaster of some kind, these facilities are no longer available, the companies operations are more seriously threatened then ever before. Often, the effects of (consequential) loss will not be felt in one isolated sector of the company, but rather in interrelated implications.
American insurance industry figures show that within half a day of a disaster a business operating efficiency has dropped to 96%. Within 11 days it is down to 9%. CUC provides a hot restart centre, dedicated to back-up, as well as empty shell and mobile computer rooms with and without computers.
CUC was established in 1981.
Shareholders: KLM (Royal Dutch Airlines) 32,5%, NMB (fourth Dutch bank) 32,5%, SG2 (Société Générale) 19% and FGH (mortgage bank) 16%.
CUC is a dedicated back-up centre.

1. Points of departure

Over the past five years millions of dollars were lost in Europe as a result of fire, explosions, floods, water damage, terrorism and vandalism damaging the computers of many companies. Did you ever think about the risk situation in your company ?

The number of companies where operations depend to an increasing extent on their computer facilities, is constantly growing.
If, through a disaster of some kind, these facilities are no longer available, the companies operations are more seriously impeded than ever before.

The consequantial loss caused by such disasters may materialize as e.g.

- in trade: reduced sales, or loss of market shares;
- in manufacturing: loss of control over stocks and purchase situation; disturbance of the distribution or manufacturing process;
- in management: loss of control over financial situation, or lack of management information.

Often, the effects of such consequential loss will not be felt in one, isolated sector of the company, but rather as interrelated implications. Progress in time adds to the problem. Growing backlogs, and complaints do not exactly simplify controlling the new situation.

The effects outlined above may usually be expected to be greater, as the computer is more intricately involved in company operations, through the use of terminals, data communications etc.. Which is the situation in your company.

Nevertheless most of you did not prepare for the disaster situation. A Butler Cox & Partners report produced for Amdahl learns us that 60% of you do not have a contingency plan on paper. From the remaining 40% only 50% went for a warm or a hot restart centre in order to minimize the consequential losses.

CUC intends to reduce consequential losses to a minimum, by offering back-up facilities. The type and volume of facilities to be offered are discussed with the client, and are determined by the size of the applications, that are involved in the case of back-up, as well as by the maximum period of time within which back-up facilities must be operational, viz. the realisation time.

2. The Company CUC

In the second part of the seventies a group of the bigger Dutch companies discussed their vulnerability in case of a computer disaster.
They felt the need for a back-up centre. For economic and professional reasons it was decided to investigate the possibilities for funding a professional back-up centre in the Netherlands instead of building a second centre per company. After an indepth study, leading to a sound business plan, CUC was founded in 1981. CUC is a dedicated back-up centre.
CUC started with back-up for the bigger IBM mainframes. Very soon it appeared that also 43XX and System 38 users needed the CUC facilities.
In the beginning of 1984 CUC started a preliminary marketing study for the need of back-up for HP 3000 users. Based upon the results of this study CUC announced the HP 3000 back-up product in the middle of 1984.
This first announcement was based upon the experience in the HP 3000 market and upon the intensive and positive cooperation with HP.
At the end of 1984 CUC is providing back-up for IBM mainframes (43XX, 308X), S/38 and HP 3000. The users of Bull, Dec, Burroughs, Prime, Datapoint and Wang show a strong interest in the CUC back-up products.

CUC has a share capital of Dfl. 5.000.000,--, divided over KLM 32,5%, NMB 32,5%, SG2 19% and FGH 16%.
KLM stands for the Royal Dutch Airlines, who highly depend on their computer systems. NMB (Nederlandsche Middenstands Bank) is the fourth Dutch bank, who decided not to build a second centre like the other three big banks, but to rely on CUC. FGH (Friesch Groningsche Hypotheekbank) is a mortgage bank.
SG2 (Société Générale de Service et de Gestion) is the second soft-warehouse/service centre in Europe directly after IBM, employing a staff of 4100, spread over 27 countries in the world.
SG2 has since 1976 experiences with back-up centres in France.

CUC itself has a staff of 41 persons amongst whom experts on system software and Data Communications.

3. HP 3000 back-up

The HP 3000 computer plays a very important role in many companies and institutions.
In the past few years several attempts have been made to create back-up, e.g..

In 1981	a committee of Dutch users studied to possibilities to start up a back-up facility.
In 1983	2 HP specialists studied the subject and tried to find the proper solution.

The fact that nobody succeeded in the realisation of back-up facilities illustrates the necessity to exercise back-up as a profession.

Early 1984 2 employees of CUC started a preliminary study into the HP 3000 market in the Netherlands. CUC was facing the question to offer the 'mainframe' services to HP 3000 users, or to create a completely new and dedicated combination of services.
The results of this study were presented to HP and a number of users, and after some adjustments were made Back-up for HP 3000 users was introduced.

The first conclusion of the study was that CUC should have 2 possible solutions for back-up:

- The Mobile Computer System.
- A HP 3000 configuration in the CUC buildings.

For the time being CUC has combined both solutions; one configuration is installed in a transportable computer room.

During the research it appeared that even among HP 3000 users big differences exist, from very small to very large installations, from very critical to almost 'sleeping' applications.

It was therefore reasonable to leave the initial point of view to offer a standard configuration at one standard booking fee to each of the users.

Three service levels have been created depending on the tome which is available after a disaster to start up again. Some users do need a new system within 2 or 3 hours, some accept 3 to 5 days.

Service level A

The basic product is a reservation of the HP 3000 hardware and an inventory of D.C. facilities.
A company or institution which does nog consider the use of the computer system as being essential for the operations might choose for service level A.
All preparations for back-up are made after the disaster has hit, and it will take 3 to 5 days to check and test:
- MPE operation system.
- Configuration.
- Application software.
- Peripherals (e.g. printers, telex, etc.).
- Datacommunications hard- and software.
- Procedures.

It is for this reason that CUC does not promote servicelevel A, level B offers much more security.

Servicelevel B

In service level B much time is spent to create the back-up scenario. Every action during back-up should be according to the back-up scenario to avoid unpleasant surprises. The back-up scenario has been built up in the past few years and is based on many practical experiences. Each user who chooses servicelevel B receives a book in which every step preceeding back-up is put down and which offers the opportunity to add the actions which are needed within the own organisation.

After the back-up scenario has been put together the back-up facilities are subject to an extensive test.

A CUC client who has chosen for the mobile back-up computer system has to make some technical preparations on his own premises, e.d. power supply and data communications facilities.

Servicelevel C

For those clients who have a real dynamic development in the electronic dataprocessing it is essential that the back-up facility and the back-up scenario remains up-to-date.

Servicelevel C offers the possibility to make an up-date of the back-up scenario every 6 months, and test the back-up operation at least once every year.

4. Data Communications solutions

CUC can offer adequate data communications solutions for the back-up situation. Adequate are those solutions which are technically and financially efficient. Starting point is the excisting situation within the clients organisation. The end-user situation should be kept unchanged. The back-up network should be realized without doubling the line costs and the risks should be spread as much as possible.
After consulting the Dutch PTT, IBM and the modem suppliers CUC drew-up the following solutions:

- back-up via the public switched X.25 network,
- back-up via dial-up lines,
- back-up via (simulated) leased lines,
- back-up with statistical multiplexers in order to reduce the needed line capacities,
- back-up via FEP-to-FEP communication,
- local network via remote clusters,
- local network via PARADYNE-PIX Boxes.

These solutions have been implemented in all possible combinations. CUC has proven to be able to cope with all possible networks, amongst which the seat-reservation system for the Royal Dutch Airlines.

Experiences have learnt that dc-back-up is less complicated than supposed at first glance. Furthermore that different situations ask for different solutions.

6. Procedures

Computer back-up does not stop at making available a computer centre with hardware, data communications facilities, or a mobile computer room.
Vitally important are proper preparations and tests, exactly defined in clear conventions and procedures, that are part of the back-up scenario.
The shorter the required realisation time for the back-up, the more essential the depth of the procedures. Only if adequate preparations have been made, and tests are carried out regularly, CUC will guarantee realisation times within which the back-up facility will be operational.
On the other hand it is also possible that precautions are more roughly defined as the demands in terms of realisation time are less rigid. If there are no or only very small demands and requirements with respect to realisation time, the precautions may even be (partly) omitted. In those cases a back-up operation starts off with an inventory and organisation activity, which gradually becomes an operational back-up. Thus, the back-up scenario is drafted. In such cases CUC does not guarantee any realisation time.

In the back-up scenario the following is laid down:

- the required computer configuration
  - CPU's with internal memory, channels & controllers
  - tapes, discs etc.
  - common I/O, card readers, printers etc.
- the required data communications facilities
  - the network switching concept
  - FEP entries
  - data communications links
  - modems, statistical multiplexers etc.
- the precautions, i.e. the availability and keeping up to date of
  - operating systems
  - DB and DC packages
  - program libraries
  - application of files
- the back-up procedures, describing - to client and CUC - who will be doing what and when, at the start-off, during and at the end of the back-up operation
- support by CUC
  - specialists, operators, and other staff
  - office facilities and other accommodation
- performance aspects
  - implementation schedule
  - back-up schedule

As far as utilisation of a mobile computer centre is concerned, an analogous scenario applies, which is addapted to the specific local situation at the client.

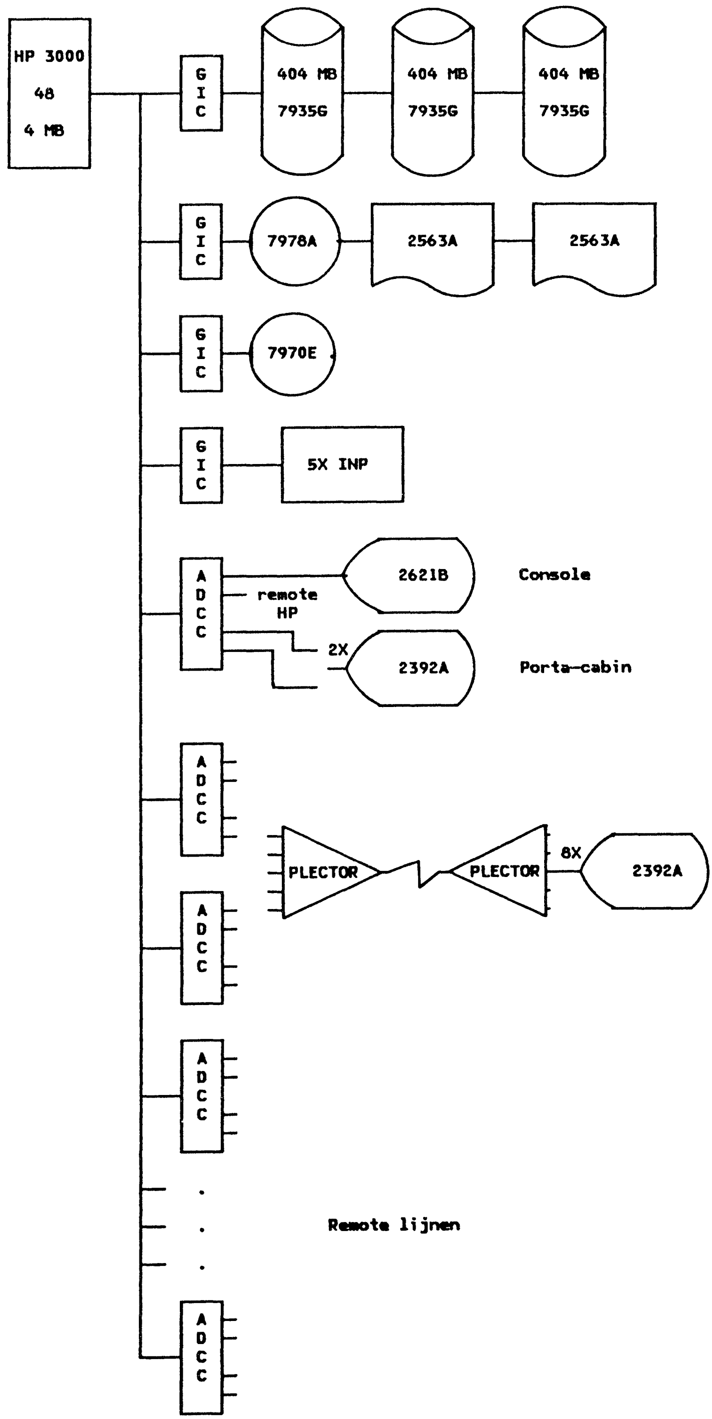
6. Production resources

CUC production resources include, in addition to an expert staff, a computer centre, its computer system configurations, the mobile computer centre, the empty shell, the portable computer rooms, facilities for data communications hardware, safes for keeping systems that are required with back-up, including files, manuals, and media als well as office room (4000 m^2), filled with terminals, data entry, telephones, copiers, telex, restaurant, etc.
Building and systems offer more than ample floor space and capacities. The computer room has a total floor surface of 1200 square meters, with possibilities for expansion.
Airconditioning and power supply systems have substantial reserve capacity. If required, CUC may convert to its private power supply system.
The building meets high safety demands, due to the site selected, and to its construction and fitting. Building and grounds are protected by means of a site monitoring system, with four zones.
A specific volume of square meters of mobile computer room is kept 'in stock' by CUC. The volume of stock is depending upon the number of clients wishing to use that facility (either as follow up to the CUC configuration or as stand alone). Quantities of spare parts/ installations having comperatively long delivery times, are also kept in stock, depending on the reserved number of square meters.

6.1 CUC standard configuration

See page 9.

6.1 CUC standard configuration



6.2 CUC standard data communications facilities

1 x 3705 FEP	200 lines	
2 x 3725 FEP		
3 x 1270 FEP Memorex		
1 x 3274 local controller	32 lines	
1 x 3274 remote controller	16 lines	
40 x 3278-2 VDU's		

Modems

V21	async.	0 - 300 bps
V22	async./sync.	1200 bps
V26 bis	2/4 wires	2400 bps
V27bis/ter	2/4 wires	4800 bps
V29	2/4 wires	9600 bps

14,4 Kbps
7,2 Kbps broadband
14,4 Kbps broadband

2 x Codex statistical multiplexer (6050)

V24 patch panel

Atlantic research datascope

line access units

6.3 Data communications for HP 3000

6 x 8 channel multiplexer M828-HP

4 x 4 channel multiplexer M824-HP

10 modems 9600 resp. 4800 Bps

7. Other services

CUC is a dedicated back-up centre. Back-up is the main product and will never be frustrated by production activities. CUC, however, can offer some back-up related products:

7.1 Risk analysis

CUC has its own computerized risk analysis system including consequential loss analysis. This is sold as a stand alone product irrispective the fact if a client signs a back-up contract or not.

7.2 Back-up studies

CUC is highly experienced in undertaking back-up studies. This is also done for companies into have their own back-up centre and for IBM and HP users.

7.3 Testing

Back-up clients can use the CUC equipment for testing at a higly reduced tariff. In case the equipment is needed for back-up reasons the testing will have do stop immediately.

7.4 System tuning, conversions, etc.

Often companies have difficulties in finding possibilities for system tuning, conversions, etc., without disturbing the normal production. CUC offers the possibilities to back-up clients to use their own centre as a clean smoothly running production centre and CUC as a testing centre.
When the equipment is needed for back-up, clients have to stop these activities.

7.5 Consultancy and courses

CUC employs a higly experienced staff of specialists. Clients of CUC can make use of the experience for consulting. Besides CUC is organizing courses and seminars.

7.6 Interim management

Often smaller computer users ask CUC for interim management in the EDP area. CUC can supply specialists during a limited period of time.

Biography

Johan Kooy
is commercial director of the Computer Uitwijk Centrum and involved in computer back-up from 1979 when the first studies for realisation of a back-up company in the Netherlands started.
Started 18 years ago within IBM and fullfilled several jobs in the information and organisation area.

IT IS TIME TO AUTOMATE YOUR DATA CENTER

By Betsy Leight
Vice President
Operations Control Systems
560 San Antonio Road
Palo Alto, California 94306
U.S.A.
Telephone: 415-493-4122

Summary:

Data processing professionals have automated and streamlined almost every department in their companies...except their own. It's time to increase productivity, lower costs and save time in the data center. It is time to automate the data center.

- -

As a data processing professional, you have managed to streamline the workload of almost every department in your company -- except your own. As users demand your department's time and resources, you must continue to produce timely and accurate results regardless of the obstacles. One solution is to apply ever increasing resources to the problem at ever increasing costs. But what if you could increase the level of service to your users with the resources you have? What if you could streamline your workload and maintain control by making your organization more efficient, cost effective, and responsive?

As we describe the automation of the data center, it's important to outline what duties are encountered in the day to day operation of the center. This merits attention as there are few people who are aware of and comprehend the total scope of a data center's numerous responsibilities.

A well-run data center will be responsible for all activities pertaining to the management, processing, and distribution of production data. These activities range from the distribution of daily reports to the receiving and handling of user calls reporting problems.

These activities may be broken down into several categories. They are as follows:

1. Daily Production Scheduling.
2. Report Distribution.
3. Tape Library Management.
4. Back-Up and Recovery.
5. Special User Requests.
6. Hardware Maintenance.
7. Disaster Recovery.
8. Restarting, Reloading and Rerunning of Aborted Jobs.
9. Asset Management.

Each of the above mentioned areas may now be broken down into a subset of activities that the data center is responsible for.

Daily Production Scheduling entails selecting and sequencing jobs for that night's production run, including tape and disc requirements as well as special forms.

Report Distribution is that age old problem of getting reports that seem to escape to Never Never Land as often as possible to the end user.

Tape Library Management is probably one of the most critical duties of the Data Center. However, it is most frequently ignored. We have all lost tapes or had them written over at one time or another.

Back Up and Recovery sounds simple, but try it sometime at 4:00 a.m. with an inexperienced operator and no decent tape library. In larger shops with 3 to 4 CPUs, a nightly back up can use 15 to 20 tapes.

Special User Requests are those phone calls usually at 4:55 p.m., requesting an emergency run that must be done by morning. This now has to be fit into the schedule (try this at month end).

Hardware Maintenance consists of scheduling P.M.'s, maintaining terminals, printers, plotters and an ever increasing number of P.C.'s. Just keeping track of all this hardware can be a full time job.

Disaster Recovery is a bedtime story used to frighten data processing managers when they are young. We spend much time discussing and, in most cases, little time implementing. This is due to the costs and lack of documentation required to move production to another site and continue without difficulty.

Restarting, Reloading, and Rerunning: the three R's. A very scary proposition. How many times have jobs been rerun and processed the data twice? How many 4:00 a.m. phone calls have been made because no documentation exists or what does exist can not be found?

Asset management means: what are my resources and how do I use them? Is my staff used at the optimum level? How much machine usage is there? Who is using what? How much do I charge and to whom?

All of these items represent challenges the Data Center must rise to meet. THE QUESTION IS HOW?

As can now be seen, the data centers' responsibilities are wide ranged and far reaching. Furthermore, the user communities' perception of the data processing department as a whole is based primarily on their view of the service level from the Data Center. In other words, when a user does not get a report that was expected, the whole department looks bad.

Now that we understand the problems, let's prioritize them and look at possible solutions.

The approaches to solving these problems range from ignoring them to automated systems with a hectic stop at manual systems.

The system used to automate the Data Center can range from stand alone, small systems managing each problem separately, to a total systems approach.

It has been my experience that the total systems solution gains the Data Center the maximum benefit in its day to day operations rather than a piecemeal solution.

The optimum system handles job scheduling automatically and is intelligent enough to support inter-job dependencies and job-stream non-concurrence. This system should encompass all aspects of the Data Center including Report Distribution, Job Launching, Automatic Tape Management, Resource Management, and Back up Recovery. In short, this system should provide the standards, controls and auditability necessary to support a successful Data Center.

Let's look at a company that chose to implement Data Center Automation Software. The first step was getting a commitment from management. Next, the system was installed and the problem areas approached systematically.

The most critical problem facing most data processing organizations today is not technical, but political in nature. This is how the user community views the service provided by the department. As stated earlier, most users contact data processing through the use of reports. This makes getting the users their reports of the utmost importance. An added benefit to getting users their reports is a drastic reduction in headache-generating phone calls.

To accomplish this, our shop automated their distribution list tying all reports to specific jobs.

Daily Production Scheduling would be next. If the schedule is wrong the right reports cannot possibly go out.

Automating the Production Schedule is complex but well worth the effort. The scheduler must be intelligent and be able to handle the complexities of inter-job dependencies, non-concurrence, job priorities, and run books. Run books are events that may require a response from the operator.

As the industry as a whole moves more and more toward distributed processing, placing control in the user hands, special user requests are becoming more numerous. These must be handled in order to maintain the image of a service-oriented organization.

These user requests are the most difficult to plan for as they are a constantly moving target. Not only must they be scheduled but any parameters required need to be collected at the time of the request. Automating this segment can lead to time savings of up to 4 hours a day.

Back Up and Recovery are key to the survival of the data processing department. Who wants to tell the user community that a day's worth of data has been lost and must be reentered? In many cases, this is not possible since source documents are not maintained after data entry.

Back Up and Recovery jobs that are scheduled automatically and coupled with automated tape management provide an extremely secure method of insuring data integrity.

Tape Library Management is crucial not only to maintain data integrity but also to maintain one's sanity. Have you ever felt that sick feeling in the pit of your stomach when you've sent for tapes stored off site and when they arrive, the one you're looking for is not there?

Tape Library Automation can range from entering tapes used into a database to a system that selects tapes automatically from the defined library. This takes the guess work out of library management. Our shop chose this latter method and found it highly successful.

The three R's: Restarting, Reloading, and Rerunning provide the operations staff with the incentive to become proficient at tap dancing, and search and rescue. The search for rerun documentation at 2:00 a.m. closely resembles the search for the proverbial needle in a haystack.

Disaster Recovery is like insurance. We plan for and in some cases pay for it and pray we never have to use it. The recovery of application systems is difficult because documentation is rarely updated as these systems evolve.

While there is no way to fully automate the three R's and Disaster Recovery, the fact that all jobs are documented in an automated system will greatly enhance the chances for successful recovery from aborted jobs or disasters.

Last, Asset Management is crucial for the planning of growth and maintenance of existing equipment.

Such a system should contain data pertaining to hardware inventory and resource utilization such as CPU and connect time.

In a shop that has any of the insurmountable opportunities stated, it behooves the management to investigate automated systems if not to gain some peace of mind, then to insure sanity. It is time to practice what we preach. We spend our lives automating the rest of the world and ignore our own needs.

- -

Biographical Information:

In 1976, Betsy Leight founded Operations Control Systems (OCS). Since the beginning, she has been responsible for the overall technical direction of the company including the management of both technical projects and personnel.

Mrs. Leight is considered an expert in the HP Community. She has lectured at the EDP Auditors Association and at numerous HP International and Regional User Groups throughout the United States and abroad.

Prior to founding OCS, Mrs. Leight spent four years as a systems analyst with Hewlett-Packard Company, where she was responsible for the installation and support of HP 3000 systems throughout the northwestern United States.

Mrs. Leight studied computer science at Boston University and Carnegie Institute of Technology. She has attended numerous technical and management training courses at IBM and Hewlett-Packard companies as well as other independent training organizations.

WHO NEEDS AN OPERATOR ANYWAY ?

The Potential for an Automatic HP 3000

John Parkinson

1. Introduction

Not so long ago, all computers were great big boxes, that needed a lot of care and attention if they were expected to work. All sorts of specialised tasks had to be carried out to ensure that data went in, got processed and then came out again in a more or less correct fashion. Systems were locked away in specially conditioned and protected rooms, and mere mortals were seldom allowed anywhere near them. To tend to the computer's particular needs, a new breed of specialist was created, called the computer operator. (A number of other descriptions have also been developed, but we will not discuss them here.) Indeed a whole hierarchy of job titles and functions was created by personnel departments to justify the increasing numbers of people who swarmed over the computers, and invented reasons why they failed to work correctly so much of the time.

As time went on, computers became smaller, more resilient and less in need of continual attention (well at least some did). Many of the jobs traditionally carried out by operators were no longer necessary, but, like many well established species, operators refused to go away. Instead they either invented new activities that "needed" doing, or better still, a new job title, that gave the impression that they were doing something new!

These days, computers are getting very small indeed, and are appearing in all sorts of places, where they seem to work quite happily, without all the special provisions that used to be necessary. So, why do we still need operators, if indeed we do? There are some clear advantages to being able to do without them, so let's look at the sort of jobs they are supposed to do, and see if we really need a human being to be present. After all, having paid a lot of money for the computer, we ought to get as much work out if it as possible.

For many new computer users, the cost of providing for operational support staff may in fact be prohibitive. Small configurations, or network node sites may not be able to justify the cost of a full time operator, or, worse, the requirement for staffing outside of "normal" office hours. So there is an economic as well as an organisational incentive to do without traditional operator posts if at all possible.

2. Operational Activites

We can start by looking at the kind of jobs that operators were originally entrusted with, when computers were still large and locked away. Here is a list, compiled from a number of job descriptions for various types of computer installation, during the period 1970 to 1980.

- Start and stop the system, both hardware units and operating system.

- Prepare and maintain card decks for SYSGEN and JCL (both production and test).

- Maintain the Tape Library, and retrieve and mount the appropriate tape, when needed by a job, or requested by a user.

- Maintain the exchangable disc library, and mount the appropriate disc (cartridge/pack) when needed by a job.

- Control printed output, ensuring that the correct stationary was loaded and aligned, and that printed output was distributed in a timely and responsible fashion.

- Schedule work on the system, according to the production schedule, and ensure that all work was completed on time.

- Maintain records of work done.

- Manage the re-running of any jobs that failed after finding and remedying the cause of failure.

- Control access to the system, so as to maintain the required level of security.

- Manage the stock of system consumables, to ensure sufficient on hand for production requirements.

- Monitor and control communications facilities.

- Take and keep secure copies of the system files so that recovery was possible in the event of a system failure of any kind.

and so on.

In multiple system sites, things could be even more complicated. In fact this list is by no means exhaustive, and you can probably think of many more activities that have traditionally been the province of the Operation's Department. Given this huge range of work and responsibility, it is hard to see how the average operator had any time for (for instance), reading comics, tea breaks, making paper aeroplanes.........

Of course, with all these complex, interacting tasks, it was essential that the operators be provided with clear, unambiguous, easily understood documentation. In many installations, the operations manager would refuse to accept a system that was not documented to his satisfaction (although no evidence actually exists that operators read documentation, once it is supplied) thus making life even more difficult for hard pressed system designers, who are not usually the world's best or most enthusiastic authors.

Most of the tasks on this list were developed to support very different computers from the HP3000. We can, therefore, remove some of them from our "Operator Requirement" (OR), straight away. For one thing, it is a good idea to leave an HP3000 running all the time, and never switch the various units off. This actually increases the life of most components (provided that adequate environmental provisions have been made) and reduces the probability of hardware failures.

For another thing, card readers are a relatively rare sight on HP3000's these days, so all those dusty trays of 80 column cards can go too. (Actually, I had to convert a large number of old 80 col data cards onto a HP3000 a few months ago, and I had a lot of trouble finding a system with a card reader. In the end, HP discovered that they themselves still had one!) Also the SYSGEN procedure (i.e. COLD/COOL/WARMSTART, RELOAD /UPDATE) on an HP3000 is rather more straightforward than on some other systems, so this, previously esoteric and time consuming activity can also be dispenses with.

In fact, it seems that even the relatively simple startup procedures on existing HP3000's are too complex, since HP have introduced an even simpler procedure on the new series 37 (Mighty Mouse) machines. Now you can load a system at the turn of a key, and not have to worry about options, time/date input etc. It is even possible to set up a series of autostart files, that display a simple menu from which the user chooses an automatic startup option. More of this type of facility is promised for the future, perhaps even for existing systems.

So now let's look at the remaining activities, and see if our HP system could manage without them.

3. Access Control and Security

We will start with an important and often neglected area, ensuring that only those people allowed to use a system actually do so, and that they only do the things they are employed for. Any system will need some type of security, whether there are operators about or not, because the HP3000 is designed to make it easy to get computing out of the computer room, and into the normal working environment of users. MPE gives you lots of ways of implementing security features, but does require at least one person (with SM capability) to be responsible for overseeing the general security of the system. Do you need more than this? Provided that you take some reasonable precautions, the answer seems to be "NO". All user's logging onto or off of the system can be recorded, via the system logging facilities, and reported at intervals. In fact, system logging can automatically record most system events, and provide a comprehensive audit record of user activity. The only problems involved are the impact on response time, and the provision of enough disc space for log files. MPE even manages the creation of a new logfile, when the old file is full.

Of course, the only real answer to your security needs is dependant on the applications you run and their requirements for controlled access. In general, however, so long as you use sensible security features (eg. log the console session off when it is not needed) and restrict the use of SM and PM capability, then you should not need a physical operator presence.

One note in passing. Most sites put a password on MANAGER.SYS and some remember to password FIELD.SUPPORT and the CEDIAG account (if they are to be left on the system) as well. If you run HP's office products or communications software (or PM/MM or FAS or.....) there will also be accounts like HPPL85 etc. on your system. These are the support accounts for the various sets of HP applications software, and they are often created with SM capability. Why not check yours when you get home? I bet they don't have passwords.

If you want to go further than MPE allows with security features, you can buy MPEX or a similar product and add MENU processors to your logon UDC's to restrict access to the system. The possibilities (and the problems of forgotten passwords) are endless. UDC's can also be a source of problems, especially if they are defined with NOBREAK, and something goes wrong while they are in operation. It takes a lot of thought and effort to undo some of these problems, and it may be better to think up an alternative approach. In any case, UDC's use up a LOT of system resources that might be better spent elsewhere.

HP have also provided some nice extensions to the system security features for the new Series 37, including the ability to enable and automatically logon sessions under system control. In some cases this may help control access, although you are still left with the problem of how to control a user once they are actually logged on!

You may have noticed that operators do not have one of the world's best memories. So they often employ one of their other skills and write the system passwords down on odd pieces of paper. Which they then forget, and leave in the canteen, loo......

The computer room is also a great place to develop and practice the traditional computer "hacker" skills of security breaking, moonlighting, trying out new programs and so on. It is usually isolated, seldom visited and without many alternative distractions except the systems games account, which is usually the first target for bypassing security, and the reason for poor overall system performance!

4. Scheduling

Most production systems have some sort of time dependency associated with their workload. Jobs are not run at random and it is important to maintain correct sequencing and to execute periodic work at the correct time. Traditionally, this has been one of the most important operator functions, and at first sight, this is also true of the HP3000. MPE JCL is much simplified in comparison to larger systems control languages, but still provides powerful controls over work that is not executed interactively. Use of JCW's or intermediate status files (eg message files) allows a job stream to maintain sequence and, if necessary to synchronise with other job streams. All we need to do, therefore, is ensure that the work starts at the right time, and we can have the JCL and the applications take care of the rest for us.

There are at least two approaches to doing this. One is to set off all "batch" jobs from a session programatically. This is a powerful method, but depends on the session user remembering to issue the relevant commands, either directly or through an application function. This method is therefore OK for ad hoc jobs and for some routine work, but is not really suitable for everything, especially for long interval periodic jobs, or work that needs to be carried out at hours when session users are not around.

The second approach uses a "Sleeper" job that runs constantly on the system, and wakes up from time to time to check the system clock and an event file to see if work needs to be initiated. This is a much more generally useful method, and by careful design can be completely file driven to allow for virtual sequencing. The contributed library contains several such programs, which can be used as a basis for a real control system.

Once again, the 3000/37 recognises the usefulness of this feature, and provides a mechanism within MPE to schedule jobs sutomatically. HP are also talking of extending this feature to other systems soon.

In most sites, a combination of approaches will be needed, and since, inevitably, some jobs will be in both classes of control, an activity record will have to be kept to ensure that job execution is not duplicated or otherwise disrupted. All this takes forethought and effort to set up, but once done, requires little human intervention and hence is less error prone than operator initiated work, which is subject to problems of misinterpretation in documentation. (Which is, of course, the fault of the designer who wrote it, not the operator who read it!)

5. Printing

Despite the advent of electronic mail and office automation, most systems still produce printed output, often by the ton. This has to be printed on the correct size and type of paper, collated, burst, bound, distributed and so on. Generally these are labour intensive activities, that have not been extensively automated, although many of them have been mechanised to some extent. Because of the need to control stationary changes, manage storage and house the various mechanisms used, all these functions have been carried out by the operations department on a centralised basis. This has also simplified the control of sensitive printing (eg payslips, cheques, statements), although the operators, as a consequence, do get to see everything that is printed, and so become the best source of rumour and scandal in the company.

Much of this is no longer necessary. With the development of low cost laser printing and quiet terminal printers, documents can now be produced from plain paper, instead of preprinted. Single sheet printing reduces the need for decollation/bursting and the combination of printing and duplication facilities in the new generation of printers reduces the need of multipart forms. Support for large numbers of terminal printers means that printing can be distributed directly to the areas that will use it, rather than be produced centrally.

There may well be cost savings in equipment too. 10 terminal printers can print in aggregate at c. 900 1pm with the full character set, for a cost of under £20,000. This compares favourably with the cost of a 1000 1pm printer (£24000) and a hardware fault only reduces printing capacity by 10% not 100%!

Of course, this approach uses up more terminal ports, but if this is a problem, some printers can be slaved from terminals or microcomputer workstations, and others can run on DSN/MTS links via an INP. It may also use up more disc space and spooling overhead, but this is easily solved by more hardware, system tuning and/or applications design.

6. Backup and Dumping

Because all computer systems fail from time to time, live sites need to make provision for this by taking copies of their files periodically. IMAGE users are provided with a set of tools to make the process of recovery easier but these still depend on the availability of a "good" copy of the database as a starting point. As databases get larger, the problems of regular backup get greater and hence backups happen less often. Some sites now copy the full data base only weekly or less often, and take daily copies of the image log files to recover from. Even with high speed tape decks, recovery of very large databases can be a lengthy process, and care needs to be taken, as tapes can develop problems. As well as the databases, other transient and permanent files need to be stored in case they too have to be recovered.

Traditionally, all this backing up has been to magnetic tape, mostly because of cost and resilience. Many current tape decks have automated the loading process, but someone still has to mount the tape. Very few installations, as yet, have the fully automated tape systems occasionally seen on TV, that have robots to do this job, so if we want to do away with human intervention, we need to look at alternative back-up strategies, that do not use tapes.

Essentially we have only one option, and that is to carry a duplicate of all our dynamic data on additional discs. This is the TANDEM route to nonstop computing, but can also be approached on the HP3000, albeit at a considerable cost. Ideally we would need the following items to insure that our data and files remain intact.

- Uninterruptable power supply. Powerfail/Restart is a wonderful thing, but powering off discs during a physical write is not.

- A second CPU to manage the update of our duplicate discs. This is to reduce the load on the "real" system, and to insulate our backup from system failures on the first CPU.

- Interprocessor or intersystem links, to pass the data accross to the seond CPU, once the first has updated its files. DSN/DS using the INP is all we have at present, but GIC to GIC DS at 1Mbyte/sec should be with us soon.

This approach is in use at several HP3000 sites around the world, and works well enough. Because the second CPU only updates files, it can be set up to be the second system for several different first systems, if required. Of course, this will eventually overload it, and make the probability of a system failure unacceptably high, but that's economics for you.

We can, in fact go further with is backup concept and interface all the terminals on our system through an intelligent port contention switch. Some of these devices now include the capability to detect that a system has failed, and to automatically load in a backup switching map to reroute terminals to the backup processor. By using the remote console facility on the HP3000 and techniques similar to those used in such CSL routines as SLAMMER (which is designed to issue a remote emergency powerdown command) we can fully automate the switchover of both the terminal network and the applications environment.

Of course, the actual mechanism for the automatic second copy update can be complex, as can the automatic switch from backup mode to live mode if this is needed. All this can be handled, however, without too much esoteric code, provided that the application designs are thought out in advance, and some basic techiques of good HP3000 practice are adhered to.

And don't forget that, if the mode switch actually occurs, there will be NO second copy updated until the problem is resolved and the normal mode of working resumed.

7. Other Things

That still leaves us with a smaller list of activities that need operator action. However we can also dispose of most of these too. If we are not using magnetic tape for logging and backups, we don't really need a tape library. Intersystem transfers of files can be carried out over DSN links (using DS, RJE, MRJE, IML or X25 as appropriate) and MPE updates should be done by your SE anyway. So we can dispense with the tape handling and associated record keeping entirely. Also we will use only fixed discs, for reliability and low maintenance cost, so there will be no exchangeable disc library to manage.

We can also dispense with the need to manually record work done on the system. Several automatic operational accounting systems are available for the HP3000 and some include full operational management capabilities and reporting. Since we have no need for consumables in the computer room, the need to control stocks can also go.

Communications network control can be handled by an automatic network management system, that checks and reroutes arround problem circuits, monitors traffic and balances out the system loads, according to preset criteria. As communications devices become more intelligent in their own right, this will be even easier to do. Already we have modems that will detect a broken circuit, redial and re-establish the circuit within 30 seconds, all automatically, and will notify the network management system of the problem. Autodial under program control is also readily available, and virtually all modems now offer auto answer as standard. The newest datacommunications test sets can be set up to monitor lines until one of range of conditions is detected, store data for later investigation and send action messages to the network control program to initiate remedial action, all automatically.

We are thus left with one final activity group for our operators. What to do if something goes wrong. We will look at this in the next section.

8. Problems

Computer systems being what they are (i.e. complex systems designed and built by fallible humans, and used by equally fallible other humans) things will from time to time go wrong. It is therefore reasuring to know that there will be an operator around to sort things out and keep the work going. However, if we examine some practical experience with this scenario, what do we find?

> Most problems occur when there is no operator about, and hence go unnoticed until they are disasters (e.g. Monday morning).
>
> In c 99% of cases, remedial action involves phoning the analyst who supports the application that failed and/or the manager of the department for whom the application is run, and asking for help. There is always a good reason for doing this, usually because the recovery documentation - (a) lost; (b) out of date (c) non-existant; (d) incomprehensible; (e) currently being revised to take account of the program change that caused the failure.

Under these circumstances, we can also automate the operator function (i.e. making the phone call) using a console message monitor, programmed to recognise message text for various failures, and in response, to autodial a preset number and speak a message. The system can try several numbers in turn, until a connection is successful, and (in the US at least) can accept dial tone codes to initiate some remedial actions, although these are essentially limited to console input command strings.

This device has in fact been available for several years in the USA, and is invaluable for sites where there is only callout cover overnight or at weekends. Now that approved autodial equipment is available here too, it could be added to UK systems at relatively low cost.

9. Conclusions

So where does all this lead us? We have seen that virtually all the traditional operator jobs can be in some way either automated or eliminated, although some if the solutions could be prohibitively expensive. Why, therefore, should we still use operations staff, and if we do, what should be their role? Well for one thing, the system backup strategy described in section 6 works better in theory than it does in pratcie, and it would be a brave DP manager that depended on this procedure alone. (Ask Shell UK about backup without tapes, which is less extreme than the method proposed). So some tape backup activity is necessary.

However, on a properly balanced configuration, this should not be excessive, if a sensible strategy is used. Say 30 minutes/day to copy logfiles and transient data, plus 2 to 3 hours once a week to dump the full databases. With the latest improved store routine in MPE-IV (Q-delta 2m patched) and MPE-V/P, and with the 7978A tape deck, backup rates of c. 800 Mbytes per hour are possible. Even with the 7974 medium speed tape drive, rates of c.300 Mbytes/hour are possible.

We have also, inadvertantly, created a new class of activity for our operators. The proliferation of terminals, terminal printers, and other remote printers and workstations also leads to the proliferation of minor problems with them that users are reluctant to learn how to solve directly. Things like ribbon and stationary changes, paper jams, locked keyboards and missing power cords need specialised help to sort out, so we end up using the operational staff to do this. This can fill in a great deal of their time.

There also remains the fundamental problem that, as an increasingly large proportion of errors are handled by the system, those that are not become more and more serious, and therefore more difficult to recover from. Without human supervision to monitor the progress of complex job sequences, the probability that an error will be major, or will remain undetected long enough to become major, increases rapidly.

So we are faced with the need for at least some operational staffing. The next question is, how much? Here I will simply relate our experience with a site that insists on the absolute minimum of operational staffing, yet runs a system that supports 60 terminals and printers at four sites on a continuous availability basis. They have 1.2. Gbytes of disc and only 7970E tape drive for backup to tape, and run a mostly online database orientated environment, with daily and periodic reporting and control jobs, some of which are streamed and some initiated from the interactive applications.

The site has one designated operator, who works about half time on system related tasks, and half time as part of one of the departments served by the system. He is covered, for holidays and sickness by a rota of 3 further staff, who have also received the basic operating training needed, but who normally work full time for user departments. The operator works a shift from c. 11.30 am though to c. 7.30 pm with flexible hours, designed to reduce overtime. This approach works very well, within the constraints imposed by the system, which has no central printing facility.

So what conclusions can we draw? Certainly the HP3000 can be made relatively operator independent, and provided that there is some degree of sensible resource monitoring, will run very well with a highly automated job management system. Some human supervision is still desirable, however, and the problem of adequate backup remains. It seems therefore, that we will have to keep the operator around for a while longer, and perhaps change the password on the games account more often!

References

1. Green G. & Farber R.C. Introduction to Security, Security World Publishing Co. Los Angeles 1975.
2. Parkin A. Systems Management. Edward Arnold London 1980.
3. Chandor A. Choosing and Keeping Computer Staff. George Allen & Unwin, 1976.
4. Lucas H.C. Measuring Employee Reactions to Computer Operations. Sloan
5. Management Review, 1974
5. Parkin A. Computing and People. Edward Arnold 1977.
6. Frielink A.B. Economics of Informatics. North Holland 1975.
7. Saville P. & Simpson D. Selection and Testing for Computer Staff. Data Processing Sept/Oct. 1973.
8. Wilcox R.H. & Mann W.C. Redundancy Techniques for Computing Systems. Spartan Books, Washington 1962.
9. Bingham J.E. & Davies G.P.W. Planning for Data Communications. Macmillan 1977.
10. Clifton H.D. Business Data Systems. Prentice Hall 1978.
11. DSN/DS Reference Manual. Part No. 32190-90005.
12. HPMAIL Administration Manual. Part No. 36570-90004
13. MPE Commands Reference Manual. Part No. 30000-90009
14. MPE System Manager/System Supervisor Reference Manual Part No. 30000-90014.
15. Computer System Console Operator's Guide Part NO. 30002-90004.
16. Hill J. Comparative Merits of a Central Database and Replicated Databases for an Inventory Control System. Proceedings HP3000 IUG Anaheim 1984.
17. Heidner D. Disaster Planning and Recovery. Proceedings HP3000 IUG Anaheim 1984.
18. Volokh. E. Burn Before Reading - Hp3000 Security and Your. Proceedings HP3000 IUG Anaheim 1984.
19. Diehl R. Central Site Support. A Case Study. Proceedings HP3000 IUG Montreal 1983.
20. Sabean C. & Dowling J. User Friendly is Great! But What About Us Operators? Proceedings HP3000 IUG Montreal 1983.
21. Cuthbert N.H. Decentralising the DP Function. Proceedings HP3000 IUG Edinburgh 1983.
22. Hande T.K. Remote Spooled Printers in an Operatorless Environment. Proceedings HP3000 IUG Edinburgh 11983.
23. Mansfield M.P. Remote Lineprinting. Proceedings HP3000 IUG Edinburgh 1983.
24. Pechulis M. System Security in an Operatorless Environment. Proceedings HP3000 IUG Copenhagen 1982.

SD - SYSTEM DEVELOPMENT

FRIENDLINESS STANDARDS

Henk Alblas
Hewlett-Packard (Canada) Limited
Toronto, Canada

Summary:

Many car buyers select their new pride not based on fuel efficiency, but based on dashboard layout or the color of the vehicle. Similarly, full user acceptance of software systems highly depends on the user friendliness of the application. This paper describes standards for ease of use and user friendliness as it applies to the Hewlett-Packard 3000 computer series. These standards deal with issues such as consistent screen layouts, error messages, menu hierarchy, menu optimization, separation of on-line, background and batch functions and information contents of reports. The paper also discusses the use of HP software products in combination with user developed applications, maintaining friendliness.

Intended audience: Everyone involved with program, design development and implementation.

The terms "ease-of-use" and "user-friendly" can be found these days in virtually any document describing a software product. Does that really mean that these products are easy to use and friendly to the user? The answer is...it depends. Let's for the remainder of this discussion just talk about the "friendliness" of software product, because we can look at ease-of-user features as ways to make the product "friendly" to the user. This "friendliness" can be defined in many ways, one arbitrary definition follows here:

> The friendliness of application software depends on the ability of this software to provide the intended user population with the environment to perform a defined task in an efficient way, with facilities to assist in the case of errors or when questions arise, resulting in an end-user that feels comfortable and in control.

I will look at some parts of this definition in more detail shortly and I will try to translate this into concrete suggestions and ideas, but first let us wonder why friendly software is important. Some people will say that all the extras to provide this friendliness just use up CPU cycles and they do not get any real work done. Or they say that the important thing is to make the application work and make it work right. All these things are true, but "unfortunately" the applications will have to interface with human beings. That is why user friendliness of batch programs is not such a hot topic. The successful interaction with an end-user at a terminal device will depend highly on this end-user's acceptance of the application. What I am saying here is that even the most efficient and technically perfect system is going to fail when the users do not want to or know how to use it. Obviously, a system that technically is not sound will be a failure, no matter how great the user interface is. So we really have to think of the user interface as an integral part of our systems and an important link in the chain. Thus, a good system has a good user interface, a user-friendly interface. This brings us back to the definition of user-friendliness. I will focus on some of the keywords and phrases in this definition.

"Intended user population"

The user or users of an application can be anyone from an executive accessing his electronic mail system maybe twice a day to an order entry clerk keying in orders all day, from a programmer in the MIS department writing programs to shipping personnel requesting packing lists. This variety of users dictates the "friendliness standards" that have to be applied. This means that an interface in the eyes of one user can be "friendly", while another more experienced user may find the same setup tedious and "unfriendly". In the design phase, the user will have to be characterized and the appropriate "friendliness standards" will have to be defined. The appropriate standards are based on the frequency of use, the user training time acceptable, the requirements of the program and the homogeneity of the user group.

If a product is used only occasionally by a user, the important parameters are things like very clear prompts, help text and simple descriptive commands or keystroke sequences. In this case, the number of keystrokes, shortcuts to go from function-to-function and so on, are not or less important. For this type of user a simple reference card with information about the application, as well as logon sequence and passwords, can be an excellent tool. Reference cards and user documentation are definitely components that determine the friendliness of the software. In spite of the emergence of the concept of the "paperless office", there are at this time many things that can be depicted better in ways other than on a terminal screen, especially if this screen does not have a graphics mode. This would be an excellent place for a pitch on the wonders of HP laser printing and HP plotting, but that goes beyond the scope of this presentation.

For the user that will be working on his/her terminal for long periods of time each day, the criteria will be significantly different. Lengthy prompts and messages are a major obstacle here, speed and minimal number of key strokes are the keys in this case. Going from one functional area to another, if this is consistent with the operators normal work method, should be direct, (i.e. with one keystroke). Commands and selection input should be as short as possible even if this means that these are less descriptive because of this. For data fields and sequences in screens common defaults should be provided, again eliminating all unnecessary activity on the part of the operator.

Between the user that only uses the system occasionally and the all day user, a variety of different user types can exist. Again, it is important to define in the design phase what type of user we are dealing with. In the case of an user population for an application that has a lot of variety within the group, a design whereby the user can select the sophistication level him/herself is preferable.

"In the case of errors"

Systems should have all the checks and validation to ensure that all detectable operator errors are identified before they are permanently entered into the system. The normal method for this is the generation of an error message on the user's screen. Important criteria for the make up of the message are that it should be concise and clear and above all give the user enough information to correct the error. In the case that the end-user cannot correct the error directly, what to do, who to contact and what information has to be provided. For example, if the application detects that a file is full, the error message should reflect that situation, advise the user to contact system management and tell the user the name of the file in question so that this can be relayed to system management.

Error messages can be backed up by help facilities that explain to the user what input and keystrokes are expected.

In the case of an experienced operator, the error indication can be more subtle. A terminal beep and/or a blinking field enhancement could be enough for this user. It is highly recommended to provide a means in this case for the operator to request the full error message if the problem is not apparent.

The user of screen labeled softkeys, as found on all current HP terminals and many non-HP terminals that are HP 3000 compatible, provide a means of selecting an activity with a single keystroke. Although used most often in a full screen or block mode environment, these softkeys can also make character or line mode applications more powerful. The labeling of the softkeys should clearly identify the function of the key and use of the softkeys should be consistent. Of course, this standardization of softkeys can be site specific, but, if HP software packages or third party products are used, a standard consistent with these purchased products is a must to prevent confusion for users switching from one application to the other. HP products will typically use the following softkey assignment, softkey 1 "OTHER KEYS", softkeys 2 and 3 "NEXT ..." and "previous ...", softkey 7 for "HELP" and softkey 8 "EXIT" or return to previous selection level. You will not necessarily find all these keys on all screens, although softkeys 7 and 8 are to be found in most.

The layout of the screen in full screen or block mode will be of major importance to the "high volume" user especially. The friendliest layout requires the grouping of related data and a close match with the source document. These two parameters are sometimes contradicting, in which case a change to one or the other or both is needed. In the special case where the terminal operator gets the information in a real-time manner, e.g., taking in order information on the phone or in direct contact with the source, the logical grouping of the data is of extreme importance.

The number of fields per screen should be limited to approximately 50 or less. This limit can be higher, if the format is repeated. For example, a screen with 20 lines with the same format of 6 fields is quite acceptable even though the number of fields is over 100.

Screen enhancements can be used to allow the user to quickly recognize the state of a field. The VPLUS/3000 Forms Management System "defaults" to the following convention: unprotected fields, also called modifiable fields, display in half bright inverse video; protected fields without enhancement and fields found in error in full bright inverse video and blinking.

In block mode, a line at the top or at the bottom of the screen can be designated as the area for error messages.

In a menu-driven environment, the user will select the choice of the function to performed or lower level menu from a list displayed on the screen. This method allows the user to migrate through the system to the required function by hitting softkeys or keying in the identifier of the choice. This does not require the user to remember commands and parameters as used in a command-driven environment. Going through menus, especially when there are many functions that have to be accessed, can be more time consuming than selecting the function directly at the command level. As with some of the issues I talked about earlier, the friendliness standard that you have to apply here depend on the user's sophistication. Menu-oriented systems require a "friendliness" standard addressing the issue of menu hierarchy optimization. Menu hierarchy optimization is the process of establishing what functions are needed by the user and the usage frequency of each of these, leading to a menu hierarchy with an overall minimum number of keystrokes to access the main selections.

Up to the T-MIT release of MPE, a full menu-driven system is not easily implemented. In most cases the user would have to key in his or her HELLO command. A full menu-driven system will present a menu or security screen as the system is started. But with the new feature of programmic creation of sessions, the end-user can be presented with an initial menu screen tailored to his or her needs.

"Efficient way"

Efficiency in the context of "friendliness" has a lot to do with the availability of the workstation. Tying up the terminal while a lengthy function is performed is not often considered to be "friendly". Standards should be developed to define what functions must be performed on-line, which ones can be executed in background mode and which ones can be run in batch. Let us define these modes. On-line execution means that the session serving the terminal user is performing the work to complete the required function. In many cases this is required. As an example, we can use an airline reservation system. Here we need to execute on-line to be able to confirm a flight to the air traveler. Background mode is implemented by initiating a stream job programmatically from the interactive program. This results in a situation whereby the terminal is available as soon as the background job is initiated. To inform the terminal user of the completion of the background job, the MPE soft interrupt mechanism can be used. An example of this mode of operation is the HPTOOLSET product, which will use the background mode for the compilation of source programs. Lastly, batch mode is the execution of many transactions without interaction from an terminal user. This process is normally fed by transaction files accumulated in on-line mode. Batch program streams are normally run at predefined times.

In the determination of the correct balance between on-line, background and batch, two more parameters will have to be brought in. These are response time expectations and the availability of system resources. If response time is no consideration or if system resources are unlimited, then all functions could be run on-line. In the real world, we will have to live with limited CPU, I/O and memory resources and users that expect reasonable response times. This means that we will have to limit on-line mode to functions that need "immediate" execution, background to functions that have to be executed in a timely but not "immediate" fashion, think of small reports like packing lists and workorders. Batch mode is to be used for all functions that can wait till the batch processing cycle, most often these batch streams are running during evening and night.

While talking about batch mode, I want to jump back to the statement that "user friendliness of batch programs is not such a hot topic". For batch programs generating reports, their "friendliness" depends on the match between information needed and information required. A good report gives only the information needed, not less and also not more. Don't give a user a 600 page report if he is only going to look at the page summary lines. Just give him the summary information then.

"Comfortable and in control"

To feel comfortable as a terminal user, most of us have to know that the system will not confuse us, thus, the need for simple and clear prompts and error messages, that is forgiving, when errors are detected, correction is uncomplicated or the system allows us to back out the transaction in error easily. The system should also be helpful, through text on the screens and a help facility and it should keep us informed. What do I mean by "keeping us informed"? When a function is completed, the application should tell the user that the function was successful, rather than the operator finding that out through the fact that his keyboard is unlocked again. Also, when a function takes longer than a few seconds, a message informing the user of that fact is an important factor in making the user feel comfortable. If the time needed for such a function is predictable, make that part of the message. An example is "THE SELECTION OF ALL RECORDS IS IN PROGRESS, THIS WILL TAKE APPROX. 5 MIN.

Some of the HP software products provide excellent examples of user friendly application:

- HPDESKMANAGER provides electronic mail and desk functions such as calendaring and personal filing. This product uses character mode to implement a combined menu and command-driven system with an extensive help facility. To complement this product, HP provides an interactive training facility to introduce new user to the program. This is a contribution to the "friendliness" for new users.

- HPDRAW is a member of the graphics products family and as it can be used by a wide range of users, it will ask the user if BEGINNER or ADVANCED mode is preferred over the default of REGULAR. This allows the experienced operator to use the product in the fastest way in ADVANCED mode, while a first time user is provided with a maximum of help to make him successful in BEGINNER mode.

  HPDRAW is a block mode application, that makes extensive use of softkeys to aid the user in the creation of presentation graphics.

- HPMENU is a product that allows programs and function to be selected from a menu rather than entering commands in response to the familiar MPE colon prompt. HPMENU is can be customized on an individual user basis, thus, the end-user is presented with a list of choices unique to his or her usage of the computer system. This allows for menu hierarchy optimization. With the MPE T-MIT feature of programmatic creation of sessions and HPMENU, the end-users can be provided with a full menu-driven system on the HP 3000.

These products implement many facilities that increase the user friendliness significantly. If you are not familiar with these, a demonstration or a demo copy of this software could help you in developing ideas for your own "friendliness standards" and design approaches. HPMENU can make access to existing programs very convenient for the end-user and as such it could be a first step in improving the friendliness of your systems without a major programming effort.

Summarizing, the adoption of friendliness standards in the realm of the end-user interface with the system can make the acceptance by the user and thus the successful implementation of the software more likely.

Making "friendliness" issues an integral part of the system design is key and includes issues such as identification of the user characteristics, well thought out screen layouts and error messages, menu hierarchy optimization, balancing of on-line, background and batch mode functions and the information contents of screens and reports. Many areas that effect user friendliness have not been uncovered in this presentation, but I hope I have drawn attention to some of the issues involved and that this can serve as a starting point for the development of an elaborate set of standards.

Concluding, I think we will have to keep in mind that computer system and software are in fact a model of the real world, mostly modeling your companies workings and as the company changes your systems will change and with that, new users will be introduced to the applications and the need for friendly software increases.

MY EXPERIENCE INSTALLING APPLICATION SOFTWARE

Martha DiCecco
Comshare, Inc., Ann Arbor Michigan, USA

Summary

With the ever increasing complexity of software applications pervading the business environment, the importance of quality training and installation procedures has become critical. Unfortunately, many current training programs are hastily conceived, poorly developed or presented and, in general, not given the attention necessary to do justice to the product. Vendors are faced with the need to "get in and out" of a client site in a minimal amount of time. The ability to impart the necesary skills and knowledge to use the application within these time constraints has a direct impact on the marketability of the product and its acceptance.

Data Processing personnel are generally overworked and understaffed, so they desire an application which will require minimal technical supervision on their part, as well as being personnel and resource efficient. The end users also need an application that will require the least amount of intervention from their data processing staff.

The trainer must be cognizant of both groups' needs and abilities. Being able to recognize these basic requirements is not an easy task, but the success of the installation and training process is contingent upon fulfilling them.

This can be accomplished with quality software, sufficient supporting documentation directed at both data processing and end users, pre-defined installation procedures, and training that has been planned carefully and is well thought out. If it is to be effective, this training should not be viewed as an afterthought by the provider, but as an integral part of the software package.

The following is based upon my participation and involvement in the development and installation of a human resource application for the HP3000 and the training of its users.

INTRODUCTION

Importance of Good Installation Procedures

From the onset of computerized application software, a special relationship has developed between the vendor and client. This association has evolved from a world of manual paper shuffling, through the "horrors" of the batch environment, to a stage of total reliance upon an external group of elite technocrats. Although this technology is evolving towards its ultimate goal of being totally "user friendly", it has not succeeded on the combined levels of installation, maintenance and end use. We are currently undergoing a period of transition in which there is a great need for giving end users access to this new sophisticated technology and providing them with the ability to manipulate it through education.

The need to acquire this new technology has been forced by the obvious advantages (many of them dramatic) that these services have to offer in terms of economy, efficiency, and ultimately the profitability of any operation.

Unfortunately today, few if any mechanisms exist for the client to assume total responsibility without assistance. The development of the needed techniques to teach the users to take on this new role is a complicated problem, without precedent. It has therefore fallen upon us, as trainers, to provide them with this capability. We are situated in a precarious position, being caught in the middle of high technology and the uninitiated user.

Expectations - Data Processing

In most cases, the data processing departments I have encountered are lacking adequate resources and staffing to assume responsibility for new applications. They are, therefore, seeking a product that will in fact be a labor saving device and not prove to be something that is more trouble than it is worth. The widespread fear among data processing departments is that they may inherit an overbearing, demanding, resource gobbling, time consuming nightmare. (A ghoulish monstrosity, seemingly spawned from the darkest depths of a vendor's sadistic imagination.)

A critical aspect of a trainer's job is to put these fears to rest and prove them to be unfounded. This can be accomplished in part by providing software that is well defined, machine and people resource efficient, requiring minimal maintenance and is easy to upgrade. Regardless of whether the application is simple or complex, the importance of providing understandable, comprehensive documentation can not be over estimated. This is the link between the user and the proper operational control of the system.

Of greatest importance, is the technical training, which for a variety of reasons, is often neglected. Failure of management to fully appreciate its impact results in inconsistent and ineffective training techniques, coupled with inadequate trainer performance. This can defeat even the best software application.

Expectations - End Users

The acquisition of an application package is the result of predefined needs of the end user. Most end users work with their data processing staff to select the best possible application to fulfill their requirements. Although not saddled with many of the constraints within which the data processing personnel must work, the users do have specific concerns which must be addressed.

These concerns are primarily directed at use, control and the ability to manipulate their data with minimal "interference" or support from their data processing department. It must not only meet their currently defined needs but prove to be upwardly adaptable and flexible.

Training procedures for these relatively new and inexperienced users must, above all, instill confidence along with a working knowledge of the system. Unfortunately, the end users also suffer from the same inadequate training procedures as experienced by the data processing staff.

THE PRE-INSTALLATION PROCESS

Initial Contact

Once the sale has been made and the contracts signed, control is turned over to the vendor's support staff for the scheduling of the training and installation. During this pre-installation process, the following need to be addressed:

* Arrange for on site training, including a list of participants, material and equipment needed

* Define and establish individual responsibilities and roles

* Outline steps that need to be completed (if any) prior to the on site installation and training

* Provide material and support for this preparatory work

* Furnish an agenda so that all participants are aware of events, timetables and expectations

A communication channel should be opened establishing rapport between the client and trainer. To avoid confusion and miscommunication, one person should be selected as the liaison. This personal relationship continues throughout pre-installation, training and follow-up support and lends creditability to the product. Initial contact should also resolve any questions, issues or concerns, generating an atmosphere of confidence. Consequently, the pre-installation process should be treated with more than casual consideration.

Assignment of Tasks

To ensure that the training and installation process will proceed in a smooth and timely manner and still be effective, pre-installation tasks must be assigned and completed (as best possible) prior to the onset of training.

Necessity of Tasks

The definition and assignment of these pre-installation tasks lays the groundwork on which future performance is predicated. It initiates the interactive process by encouraging early participation. Specification of needs from which the application is to evolve allows concerns and problems to manifest themselves before training. Consequently, the installation and training process will proceed efficiently once the foundation has been established.

Definition of Needs

The user must precisely identify his current and projected needs with regards to the application, with the understanding that certain modifications may become necessary and alternatives proposed. A pre-installation workbook is helpful in establishing guidelines to assist new users in defining these needs. This workbook should provide step by step procedures, instructions, and examples to aid in categorizing and systematizing their internal structure. Working through this on paper facilitates its incorporation into the new system.

The following are three illustrations from a hypothetical pre-installation workbook. Note this relatively simple format that requests responses to specific questions. Once these worksheets have been completed, the user will find their needs have been, for the most part, completely defined.

PROFILES/3000
NEW ITEMS WORKSHEET

(A) IRN: ∅255 (B) ITEM-NAME: LAST-EMPLOYER (C) LENGTH: 3∅

(D) DATA-TYPE (CIRCLE ONE): X=ALPHA Z=NUMERIC G=GREGORIAN I=INVERTED

(E) SCALE FACTOR: ∅

(G) UPDATE (CHECK ONE):

X = NOT RESTRICTED
___ = RESTRICTED
___ = SECURED

(F) IF NOT A GLOBAL, LIST SETS TO CONTAIN ITEM:

APPLICANTS

(H) VALIDATION-FLAG (CHECK ONE):

___ G
___ I ___ = INVERTED DATE
X N ___ = NO VALIDATION
___ C ___ = CODES LIST
___ R ___ = REJECT
___ W ___ = WARNING

NOTE: EXCEPT FOR GREGORIAN, ADD: X FOR ALPHA OVERRIDE, Z FOR ZONED OVERRIDE

(I) LINE No.	(J) REF-TEXT
1	LAST-EMPLOYER is the name of
2	the company where the
3	applicant last worked.
4	
5	
6	
7	
8	

(K) REF-MASTER UPDATED: ________ DATE   REF-DESCRIPTION UPDATED: ________ DATE

Example 1 - New Items Worksheet

PROFILES/3000
CODES TABLE WORKSHEET

ITEM-NAME: APPLICANT-SOURCE TYPE: X LENGTH: 4

DESCRIPTION: APPLICANT-SOURCE is a code in the EMPLOYEE-DETAIL set which designates the hiring source of the applicant.

CODE	DESCRIPTION	REQ?
A M	Acquisition/Merger	-
~~B R~~	~~Business Referral (Customer)~~	-
C O	Civic Organization	-
C A	Classified Advertising	-
C R	College Recruiting	-
C C	Community College Recruiting	-
E R	Employee Referral	-
E A	Employment Agency	-
H S	High School Recruiting	-
N A	Not Applicable	R
S E S	State Employment Service	-
T S	Technical School Recruiting	-
W I	WALK-IN	—

NEW CODES

UPDATE COMPLETED ON: ________ DATE

Example 2 - Codes Table Worksheet for code changes, additions and deletions.

PROFILES/3000
JOB-MASTER WORKSHEET

JOB-ID: M4∅∅

*FLSA: E
*EEO-CATEGORY: ∅1
*EEO-SUBCATEGORY: 99
*JOB-LEVEL: E
SALARY-LOW: 25696 . ∅∅
SALARY-HIGH: 38544 . ∅∅
SALARY-MIDPOINT: 32126 . ∅∅
*JOB-CLASS: AD
*WORKER-COMP-ID: N
*SALARY-GRADE: B13
*UNIONIZED: M
EFFECTIVE-DATE: 81121∅1 (YYMMDD)
1 { *SURVEY-ID: HY
SURVEY-CLASS: 99-99-3∅6
2 { *SURVEY-ID:
SURVEY-CLASS:
3 { *SURVEY-ID:
SURVEY-CLASS:

(* INDICATES CODED ITEMS)

JOB-DESCRIPTION

LINE 1: MANAGER of BASIC RESEARCH
LINE 2: oversees all laboratory func-
LINE 3: tions.
LINE 4:

Example 3 - Job Definition Worksheet (similar format can be used for organization, position tracking, etc.).

Provision of this type of workbook faciliates these types of preparatory tasks while keeping them within proportion.

Identification and Resolution of Problems and Concerns

As a trainer, I cannot overemphasize the necessity of adequate advance preparation by the client. The success of the installation process is heavily dependent upon the maximum on site utilization of the trainer's time, without being impeded by issues that could have been resolved prior to his/her arrival. Problems can be resolved most effectively with early recognition, preventing potentially costly errors and delays.

Success or failure of any training/installation process is contingent upon the client preparation and fulfillment of these pre-installation tasks.

THE INSTALLATION PROCESS

APPLICATION SOFTWARE:

Installation

Arrangements must be made to have access to the system, tape drive and any other peripherals necessary to load the software.

Customization

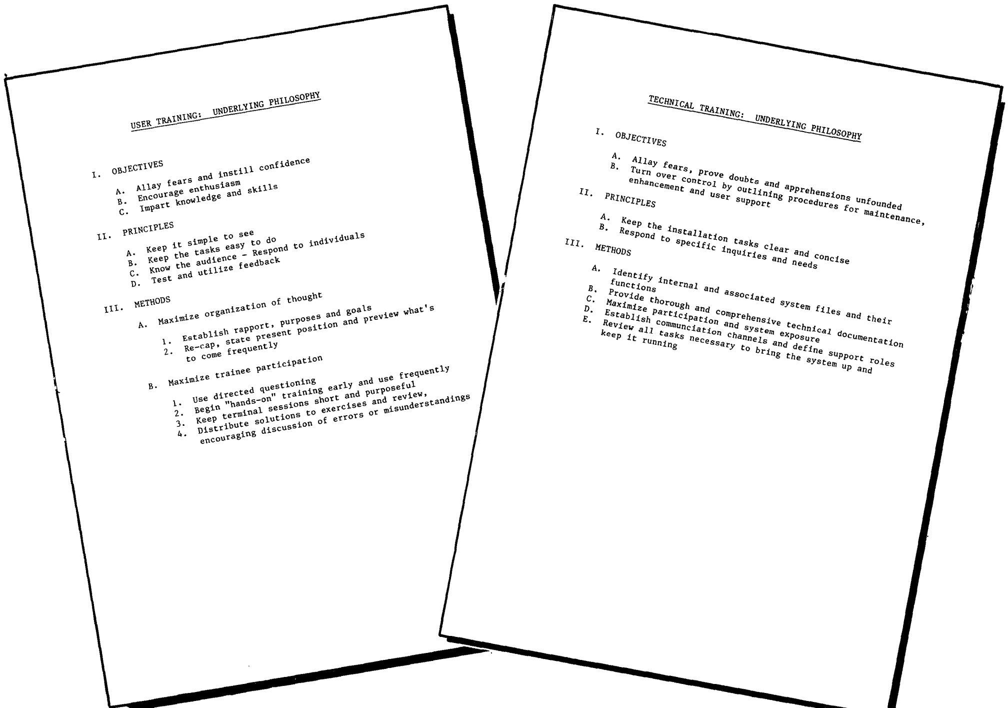
Having completed the pre-installation tasks, work can commence on the specifics that need to be incorporated immediately into the application. Although many requirements and modifications may have been dictated, the goal here is to get a basic, yet functional system up and running in a minimal amount of time. It may be determined after limited use of this operative system, that certain predefined requirements need not be incorporated after all. Sufficient direction must, however, be provided at this time to allow for any future modification of the total system.

Communication Channels

Communication channels and procedures must be identified and established. It is critical that the data processing department and the end users are able and willing to collaborate and communicate. Our role as trainer is not to act as mediators for possible political factions, but rather as mentors. It may become necessary to exercise authority in an attempt to promote cooperation. The lack of ability to compromise represents a gross misuse of time and resources.

Turning Control to Data Processing

The objective of all this is to be able to turn control of the application over to the client at the conclusion of the training period. The success of this can be measured in terms of the amount of follow-up vendor support versus the self-sufficiency and self-relience of the client to maintain and support their new system.


USER TRAINING: UNDERLYING PHILOSOPHY

I. OBJECTIVES

A. Allay fears and instill confidence
B. Encourage enthusiasm
C. Impart knowledge and skills

II. PRINCIPLES

A. Keep it simple to see
B. Keep the tasks easy to do
C. Know the audience - Respond to individuals
D. Test and utilize feedback

III. METHODS

A. Maximize organization of thought

1. Establish rapport, purposes and goals
2. Re-cap, state present position and preview what's to come frequently

B. Maximize trainee participation

1. Use directed questioning
2. Begin "hands-on" training early and use frequently
3. Keep terminal sessions short and purposeful
4. Distribute solutions to exercises and review, encouraging discussion of errors or misunderstandings

TECHNICAL TRAINING: UNDERLYING PHILOSOPHY

I. OBJECTIVES

A. Allay fears, prove doubts and apprehensions unfounded
B. Turn over control by outlining procedures for maintenance, enhancement and user support

II. PRINCIPLES

A. Keep the installation tasks clear and concise
B. Respond to specific inquiries and needs

III. METHODS

A. Identify internal and associated system files and their functions
B. Provide thorough and comprehensive technical documentation
C. Maximize participation and system exposure
D. Establish communciation channels and define support roles
E. Review all tasks necessary to bring the system up and keep it running

USER TRAINING:

Objectives

1) Allay fears and instill confidence by overcoming user apprehension and training phobias commonly resulting from lack of previous computer or technical exposure and/or experience.

2) Encourage enthusiasm by presenting the application as an exciting time saving mechanism, emphasizing the immediate availability of information.

3) Impart knowledge and skills so that users have at least a basic understanding and familiarity of their system. It is critical that they be capable of using the system prior to the trainer's departure, as once the users are on their own, they may lack the driving mechanisms to become self-sufficient.

Principles

1) Keep it simple and don't overwhelm the users by exposing them to unnecessary advance techniques. Concentrate on the simple day to day operations. However, the users must be left with the tools to obtain this advanced capability at a point in time when their comprehension of the system is greater.

2) Keep the tasks easy to do and proceed at a pace that is comfortable to the user (again working on the same premise of not overwhelming them). By pushing too much too fast, the developing client relationship is jeopardized, and user comprehension is placed at risk.

3) Know the audience and respond to individuals. Each training must, by definition be unique. It is therefore, imperative that we, as trainers, adapt on a client-by-client basis. (This ability to perceive and respond to individual needs is a measure of the quality and aptitude required to be a successful trainer.)

4) Test the trainees frequently to obtain the feedback necessary to determine their level of understanding. Reiterate any misunderstood and important points.

Methods

1) Maximize organization of thought:

   a) Begin by establishing rapport with the trainees and defining purposes, goals and outline of events;

   b) Re-cap, state present position and preview what is too come frequently.

   This allows proper pacing of the flow of material.

2) Maximize trainee participation:

   a) Use directed questioning to get individuals involved and to instill confidence;

   b) Begin "hands-on" training early and use frequently (via exercises);

   c) Keep terminal sessions short, purposeful and relevant. Although exercises and examples cannot realistically be individualized, attempts should be made to relate them to the client's organization and structure. This aids in providing for a better understanding of the material by associating it with a more familiar framework.

   d) Distribute solutions to the exercises. This provides the users with a hard copy that not only shows the "most straight forward" approach, but serves as a source for future reference. A copy of the material used during the trainer's presentation, should also be left. Review sessions upon completion and encourage discussion of any errors or misunderstandings.

TECHNICAL TRAINING:

Objectives

1) Allay any fears or doubts regarding acquisition of their new system. If data processing personnel do not possess sufficient understanding of the application, they may inadvertently pass along their frustrations to the end users, causing apprehension and uncertainty over future system use.

2) Prepare the technical personnel for taking over control and responsibility for the application. Outline procedures for maintenance, enhancement and ongoing user support. The better prepared they are prior to the trainer's departure, the more self-sufficient they will be.

Principles

1) Keep the installation tasks clear and concise. Do not confuse the issues. Care must be taken, however, to present sufficient detail to allow the technical users to optimize the use and understanding of the system and its potential capabilities.

2) Respond to the client's specific inquires and needs. Nothing can be assumed too trivial or unimportant.

Methods

1) Identify internal and associated system files and their functions. The technical staff must have an adequate working knowledge of system intrinsics if they are to assume control. It is their system and they must be provided with necessary details and instructions to allow them to understand and manipulate it.

2) Provide thorough, comprehensive and understandable technical documentation is a must. Quality documentation provides the key link between proper understanding and operation control of the system.

3) Maximize exposure to the system by working together and encouraging active participation. Experience has shown that this is still the most effective way of learning the complexities of a system. This exposure also serves as hands-on training for maintenance and future system modificiation.

4) Establish communication channels and define roles for technical support of system and end users. The client's technical staff is the primary source of information for post-training of end users. If this role and channel is not established, users are left isolated and without fundamental information. A communication channel and support role must also be defined for the vendor and client relationship.

5) Last, but not least, review all tasks necessary to bring up the system and keep it running.

CONCLUSION

Importance of Successful Installation

The task of the trainer is to introduce this high tech software to the the previously uninitiated, often uninformed public. Working within the constraints of limited time and preparation, the trainer must ensure that the user has attained the level of compentency essential to achieve maximum benefit from their system.

The clients must become more cognizant of their own needs and be aggressive in pursuing them. Interdepartmental communication and cooperation is also fundamental in attaining this maximum benefit.

It is imperative that the vendor re-evaluate the significance of the training program. The outmoded "sink or swim" philosophy is all too prevalent and totally inappropriate today. It is a contradiction that so many resources are invested in technical development, ignoring the importance of client training. Training is an integral part of most software packages, and realistically, any product is only as good as those who understand and are able to use it.

I strongly recommend that vendors invest in professional training programs to educate their trainers and provide them with the requisite skills. The ability to perceive and adapt to individual situations is characteristic of a better and more effective trainer.

Working with a quality product does much to ease the trainer's job. Also indispensable are pre-installation procedures, thorough, yet comprehensive supporting documentation, establishment of communication channels, purposeful exercises and well thought out user and technical training.

Common pitfalls which undermine even the best training programs are vague or confusing exercises and the overabundance of phrases such as: "don't worry about that now, we will discuss it later" or "yes, that is a tricky one".

The ability to communicate effectively and succeed in passing sufficient knowledge to the user for them to assume responsibility reflects the product's and vendor's image and may have an impact on the future of both.

Follow-up

Upon completion of on site training, it is important that the user is aware that a line of communication still exists. Checking in for periodic progress reports (possibly two or three times), demonstrates continued support and concern.

This further establishes confidence and trust in the product and vendor. One cannot say enough about the sense of security stemming from the knowledge that help is there when needed.

Personal Experiences

Following are exerpts from three actual installations.

The first example illustrates the outcome resulting from the failure to ensure that the prerequisite steps had been completed. Due to miscommunication, the client was unaware of the critical relationship that exists between the training and completion of the preassigned tasks. The consequences of having totally neglected this process lead to a less than "smooth" installation.

While user training proceeded unimpeded, none of the preparatory work had even been started. Many uncertainties existed pertaining to system definition, including interpretation of system parameters and decisions, as to the inclusion and utilization of specific system functions and capabilities.

No attempt was made to bring up an "inoperative" system at this time. Rather this extra time was incorporated into the technical training. The installation steps were reviewed in detail, although the supporting documentation also covered these procedures indepth.

I was able to leave, on time and with a clear conscience, despite initial misgivings. However, both data processing and end users admitted to having a comprehensive understanding of the remaining tasks and amount of time needed to be invested, before the system could become operational.

Regrettably, as what often happens when lengthly delays are encountered, the users lost their working knowledge of the application and the skills they had obtained. This coupled with complete data processing staff turnover, left the client facing an enormous burden.

Lack of adequate vendor communication prior to the training, combined with the incompletion of the client's preparatory work, follow-up support has been extensive and costly for both parties.

(Post-script: Today, despite the many problems and delays encountered, the client, partially resulting from the quality of post-installation support, is one of many satisfied and enthusiastic customers.)

In the second case, a communication lapse existed between the client's data processing staff and users. (This, unfortunately, is all to common a phenomenon.)

Program timetables and requirements were established, but upon arrival, it was discovered that no specific users had been scheduled for the training. The response to this apparent lack of coordination was a general user consensus that as a result of the thoroughness and comprehensiveness of the supporting documentation (coupled with system ease of use), further user training was deemed unnecessary. Although very complementary, the logic of this reasoning remained in question. The users forsook a vehicle for gaining preliminary exposure to their new system, including valuable suggestions, examples and techniques. It also left the data processing staff feeling frustrated (and somewhat embarrassed).

In view of the apprehension felt by both data processing and myself several attempts were made to involve the users. It was clear, however, by the end of the first day, that there was little to be gained by continuing with this phase of the training course. Time was instead spent trying to prepare the data processing personnel for the inevitable and enormous support burden that lay before them.

(Post-script: Upon resolution of their internal problems, this client has also proven to be a more than satisfied customer and a valuable reference.)

Finally let me present an example of everything proceeding according to schedule. There was strong initial communication and an astute adherence to pre-installation tasks and their completion, reflecting their sense of dedication and commitment to ensuring successful implementation.

The primary advantage encountered here was a user possessing a strong technical background, in addition to requisite Human Resource skills. This person was able to assume the lead role and responsibility of their system, overseeing all pre-installation and installation procedures as well as ongoing support, maintenance and enhancement. The benefit of having an individual within the Human Resource department with this technical expertise, allowed the group as a whole to excell in all aspects of preparation. Upon my arrival only a few minor issues needed to be addressed. By the end of the second day, the loading of data was well under way.

The point here is to show the major advantages of having users with some degree of technical proficiency on the departmental level. Being most familiar with their departments specific requirements and having the exposure to the techniques necessary to achieve them is a definite asset. Maintenance and future development are well within their capabilities.

Until such time as all systems are totally and functionally "user friendly", or all departments become independently technically sophisticated and self-sufficient, trainers will remain at the center of user and software interface and education.

Each installation, by nature, must be unique. The first two examples are illustrative of but two in a multitude of difficulties which may be experienced, inhibiting the best possible training and installation. The third illustrates the outcome when both client and trainer are committed to ensuring a successful installation process. Although infrequent, they do much to prove the worth of quality training programs.

The vendor and user are equally accountable in committing the necessary resources to facilitate the implementation of these procedures. Although the failure may rest with either the trainer or the client, it is the trainer's responsibility to rely upon his/her professional resources and expertise to ensure that a client always receives the best possible training and installation, regardless of the various impediments encountered.

Failure to recognize and acknowledge the importance of the trainers' education and the training program will inevitably lend itself to a gross misuse of time, money and resources.

Biography

Martha DiCecco
has been with Comshare Inc., for the last 7 years. She is Technical Product Manager, with responsibility for coordinating and developing timetables, prioritizing implementation requests, scheduling and performing production and maintenance updates and providing background support for 20 field offices and over 50 clients throughout the United States and Europe. She is also involved in keeping all aspects of documentation and training current and up-to-date.

She has gained her expertise by training and installing 15 clients in the course of the last two years. Currently most of her time is committed to ongoing application development, although she still looks forward to training opportunities, as time permits.

Prior to her work on the the HP3000, Martha developed applications on Comshare's timesharing systems. This custom work included cash flow analyses programs and reports, long-term financial budgeting and modeling for hospital cost centers and an application to determine and calculate variable retirement benefits.

Software Technology for the 80s (Man <--> Software <--> Computer)

Abstract:

As the interface of man to machines moves through an evolution, we are seeing software tools and applications being developed and used in new ways. Both programmers and users are being affected by these trends and new technologies. While we are seeing change accelerating, it is not change itself that causes concern, but how we deal with it. These changes are causing a new dimension to the term "Man-Machine" to develop. Since technological tools/products are developing faster than many organization's ability to incorporate them into their strategy (gain organizational acceptance), it is becoming increasingly important to be aware of the alternatives as soon as possible in order to plan for desirable change. It is important to understand software technology trends, which key trends that organizations need to track, new software technologies (programmer and application) and how to use them effectively, what can be realistically expected, where organizational resistance can be expected and how to overcome it, and how to plan for the future. These are some of the areas that will be covered in this paper. It is really software that is the key element in linking "Man to his "Machine" (providing the interface). As we will discover, successful software planning is one of our ultimate challenges.

Introduction:

In John Jewkes's book, "The Source of Invention" he noted that "Peering into the future is a popular and agreeable pastime which so long as it is not taken seriously, is comparatively innocuous". However, it is important that we understand what is taking place in the computer industry to appropriately plan for the future. In Arthur Anderson's book Trends in Information Technology: 1985 it was noted, "In the days of rapidly advancing technology no executive can afford to fall behind. You must know where technology is going, in particular, where data processing, office automation, and communications are headed." The industry is changing at an accelerating rate and success is dependent upon organizations understanding and accurately anticipating these changes in order to plan effectively.

There are a number of reasons that we must understand and plan for change:

a. Organizations need lead time to adjust to changes.
b. Organizations that implement effective strategies are gaining an important business / competitive advantage.
c. Hardware and especially software are becoming a larger portion of capital investments as well as ongoing operating expenses.
d. It can be difficult to recover from wrong decisions.

In looking at some of the general trends, it is becomes increasingly clear that the 1980's and beyond will bring a new environment / corporate culture to organizations.

a. A real need for change is exemplified by surveys indicating that in many organizations less than 1% of all management decisions are being made using on-line interactive systems. Successful progressive companies are aggressively increasing this percentage.
b. The time is right for change during the 1980s:
   (1) Organizations must address white-collar productivity. Over the past 20 years, automation has contributed to blue-collar productivity increases of around 80% while white-collar productivity has increased by as little as 4%. Productivity and

efficiency are becoming key to many organizations today and this means effective software.

(2) The Japanese are working on Fifth-Generation systems that will provide what a number of industry watchers believe are some exciting new technologies such as:

(a) Expert systems by 1990.

(b) Cordic processors (3 dimensional extensions of the 2-dimensional array processors recently popularized) by 1995.

(c) Simple Artificial Intelligence by 2010.

(d) High Intelligence Quotient Artificial Intelligence Machines by 2030.

New technologies will start being introduced during the 1980's. Successful organizations will need to plan for them. We are constantly finding that unless change is anticipated and planned for, many organizations will not be ready. Software decisions will definitely be affected by the success of these projects.

c. As a result of developing technologies the cost of hardware and software is changing:

(1) Hardware costs are decreasing approximately 25-30% per year and technology is approaching the power of what we have previously defined as a mainframe in a personal computer.

(2) Software costs are increasing at 10-15% per year.

d. From a financial commitment standpoint, user organizations as well as vendors are increasing their commitment to software:

(1) Ten years ago, fewer than one out of every 5 dollars users spent on data processing went to software. Today the split is about 50-50. According to Stephen McClellan, author of The Coming Computer Shakeout by 1990, the ratio will be 4 to 1 in favor of the programs.

(2) According to McClellan the market for software supplied by U.S. companies will grow to $50 billion by 1988 from about $8 billion in 1983. This is a dramatic increase.

There are some changes that are reshaping our traditional roles and understanding of information systems and EDP.

a. Information systems are spanning over multiple departments, operations, and divisions within a company.

b. The number of people effected by software decisions is increasing. According to Trends in Information Technology: 1985 the number of electronic keyboard devices (word processors, personal computers, and terminals) in use in U.S. businesses will increase from 7 to every white-collar worker in 1980, to 3 to 1 in 1985, to one to every white-collar worker in 1990. With keyboards software is very important.

c. Technologies are being developed that increase the number of employees in an organization that use software.

d. There are some definite decentralization efforts taking place in many organizations.

Software strategies are now affecting more people in organizations than ever before and the complexities are causing organization changes. Arthur Anderson & Co. has noted that chief executive officers and chief operating officers who want to remain competitive will have to incorporate a new title into their management system. In Trends in Information Technology: 1985, they note that the emergence of the **Chief Information Officer** at the top executive level is essential to survival.

These are just a few of the trends that are pushing us into a new age of information processing. The accumulative effect of these and other trends will have a dramatic effect on software decisions in the 1980s.

Does your organization's software strategy reflect a clear understanding of the future and workable plans to take advantage of projected technological innovations?

The right decisions can be very critical to organizations' success. Successful effective software strategies while difficult to formulate, can make the difference between organizational success and failure. Some decisions are so key that managers are truly "betting their company" on software decisions.

How effective is your software strategy?
What does it include?
Does it include all the elements necessary for success in the 1980's?
Are you knowledgeable about trends in software technologies?
Have you set your organization up for software success?

The objective of this paper is to help provide the decision makers with an understanding of alternatives, insight into some of key technologies, and a framework to develop an effective strategy. It is not to give everybody a crystal clear software plan. Software strategies must be developed with your organization in mind. We can all say that we want to be on the leading edge of technology, but this is not practical or realistic. With an understanding of trends, organizations can move into the 80s with knowledge and insight. Software strategies must be developed with an understanding of key areas.

Each of the following areas will be discussed:

I. Operating Systems
II. Office Automation
III. Programming / MIS Management
IV. Database
V. Data Communications
VI. Applications
VII. Framework for Structuring a Software Strategy

I. Operating Systems:

There has been a lot of concern about where operating systems are going in the future. It is really important to understand the trends .

There are some definite battles shaping up in this area. Many users would like to see a common operating system. However it is important to look at the motivations of various hardware vendors who also have a great impact on the operating system. The operating environment that has the greatest chance of becoming a standard is "UNIX". "UNIX" is a U.S. trademark of AT&T Bell Laboratories. From analyzing trends and data we can make educated guesses on what hardware vendors might consider doing with UNIX. The following are thoughts on what might possibly happen in this area (this does not reflect any official HP viewpoint - only insight gained from the author's research): IBM could make UNIX available as an operating system choice on their smaller systems. Long term it seems likely that they will continue to emphasize proprietary systems in UNIX application areas. UNIX could present a threat if it ever becomes a reason not to buy IBM equipment, so they will probably try to diffuse UNIX as an issue. DEC could try and use it as a way to leverage into the commercial market. AT&T will probably view it as a way to catch up on software development. The Japanese could use it as a way to increase their low-end systems share. Since HP believes in supporting standards, HP will probably continue to support the standard AT&T version of UNIX.

The following are reasons that customers prefer UNIX:

a. They do not want to be locked into buying a system from a particular vendor. They see UNIX as a way to get a commodity operating system.
b. Universities and research institutes have endorsed UNIX - 1/3 of the computer specialists in the U.S. have experience with UNIX according to the Gartner Group.
c. Those who want a common operating environment on heterogeneous machines - including different vendors.
d. They want to reduce development, support and training costs for special purpose systems.

Users who really want to have a common operating system need to track this trend and buy the standard AT&T UNIX.

There are also some perceived UNIX deficiencies that need to be addressed such as security, back-up, and standard file structures. Watching how these areas are addressed will be interesting.

It will be very important to watch the industry and see if standards for UNIX can be developed and accepted by the vendors. A group has been working for over 3 years now. Watching how IBM and AT&T work together will be key. UNIX sales are projected to increase at least through 1988 according to various market forecasts.

Another trend in operating systems is that they will become bundled in the vendors firmware. This makes it difficult to implement other vendors operating systems on other vendors hardware as well as provides the hardware vendor with increased efficiency. There are rumors from within the industry that some vendors plan to develop proprietary operating systems in firmware.

The dramatic technology advances being worked on by HP, IBM, DEC and approximately 10 startup companies to develop RISC machines will probably not have any dramatic impact on the operating system environments. It does appear that UNIX could be an alternative.

If we look at the Japanese experiment in 5th Generation systems, they plan to have parallel capabilities that will dictate new operating environments. This will be an interesting trend to track in terms of what we do with operating systems in the 1990s.

In the early 1980s users need to watch UNIX and determine if standards are developed. It is a time to continue to develop code that is not significantly dependent upon special features in the operating system.

In the later 1980s if standards are developed, UNIX implementations could be aggressively pursued with confidence. It will be important to track 5th generation system development and determine how they will migrate existing users into the new operating environment.

II. Office Automation:

There are a number of important strategic software decisions that are being made now that are very important to the office employee. What is really included in this category are those with less technical experience. Many of those affected by office automation are new software users. Needs in this area are becoming more visible. This can be exemplified by a Ruder Finn & Rotman Inc. study, "People in the Office", based on interviews with Fortune 1300 companies noting it is difficult for them to to keep up with the paperwork demands. Only 9% of those surveyed indicated that they turn out 80% or more of their office paperwork on word processors. The need to expand the use of productive tools is becoming evident.

As a result of these and other surveys of the current environment and needs, organizations are developing plans as noted below:

a. Nearly 60% of the Fortune 1,000 companies in the U.S. today have established policies and strategies that address the new office automation technology. Implementation are still in process for many.
b. By 1985, close to 85% of the Fortune 500 industrial and service companies in the U.S. will have working policies in place for evaluating, purchasing, and installing word processors, personal computers, electronic mail systems and other office automation equipment.
c. Within the next 2 years, half of all the small and medium sized businesses in the U.S. will have committed themselves to the development of practical office automation strategies.

Plans are also being funded. Organizations are committing funds to these efforts as indicated below:

a. A.D. Little research organization expects the office automation market to grow from approximately $11 billion today to around $36 billion by 1988.

b. Electronic Mail services will grow from $80 Million in 1983 to more than $2.1 Billion in 1988. There is an average rate of growth of 107% according to Link Resources Corp, a marketing research firm.
c. In the same "People in the Office" study referenced above it was estimated that by the year 2000, literally trillions of dollars will be spent on electronic equipment and personnel to run it.

As a result of these expenditures, the office worker / user is changing. Changes have already begun in many organizations and are continuing. The following are some of the trends that have taken place:

a. End users are becoming more educated and demanding more.
b. Within the past 3 years end-users have been given direct access to technology. We now have direct-users rather than end-users. More transactions are being collected / processed closer to the user.
c. During the 1960s (mainframe era), users were dissuaded from asking for applications that they might actually need. Systems analysts decided for them. End or Direct-users are now more in control.

There is momentum for office automation and plans are being put into action. However, the implementation of software is not always easy. There are areas of resistance that must be understood.

Office resistance can be exemplified by the results of an August 1982 Kelly Services survey. In the survey of 527 temporary office workers it was noted that 36% said they were worried about the "electronic invasion in the office". 1/3 of these were most afraid of being displaced by machines and 1/3 were concerned about where to learn new skills. This emphasizes the need for education. Many of the concerns can be alleviated through proper training (formal and self-paced). HP has a number of classes that can help in this area.

There is also concern about what vendors are calling "user friendly". In order for non technical employees to be able to use office software it must really be user friendly.

Some of the examples that HP has used in order to help satisfy this concern about user friendly is to quantify it. The goal for office products is to provide an environment where the user can be minimally functional competent within one-half hour of use. It can truly be done with a number of HP's products such as HPSLATE, and EZCHART. The integration of HPMENU provides an easy environment to change between various functions. These are a few examples of implementing the "ease of use" objective.

Increased "functionality" is also important in order to help alleviate concerns. With the ability to increase office productivity and perform functions that take time and have low rewards such as looking up words in a dictionary, software can help reduce some of the office worker concerns. The integration of a spelling dictionary into HP products that covers 74,000 words from the American Heritage Dictionary (99% of ones regular vocabulary) has a very positive effect.

Products engineered with ease of use and "functionality" in mind make automation easier by reducing resistance.

There are three technologies that I believe have the power to move office automation into a new age. They are (1) electronic mail, a technology that is having a dramatic effect today, (2) graphics, an existing technology that has a much greater potential as packages become refined and users understand the advantages and (3) expert systems that have the potential to change the way we get information.

Electronic Mail:

Electronic mail is changing the office environment and will continue during the rest of this decade for many corporations. With the implementation and use of this product, more employees are being introduced to office automation. They are becoming dependent on it for their communication. Along with mail implementations, other products are needed such as various word/text processors to create and edit messages. Office workers are creating and "mailing" words, text, and graphics. The ability to electronically send and receive messages has been a motivation to learn. With the introduction of personal computers and portable computers we are seeing an era where managers are composing letters/messages on air planes and going to a telephone booth and sending their message to one or many people. With new innovations in communication technology they will also be able to read/send mail while traveling in the air in the near future. At HP there are approximately 20,000 users of HPMAIL. This has had a very positive effect on communications at HP. This is really changing the way we do business.

Graphics:

Graphics is another important software technology. The mind thinks in graphics. Memory specialists convert words into pictures and can demonstrate amazing recall. In order to get information, we as human beings have 5 choices: touch, taste, smell, verbal, and visual. While the information transfer capacity touch, taste, and smell is obviously low, verbal communication is only between 150 - 300 words per minute. The channel with the greatest data capacity is visual. The eye has 150 Million light sensors, a million parallel fibers that link the optic nerve with the brain; and the ability to distinguish between 160 different hues. From this and other data we can determine that the visual channel is capable of processing information at 48 - 72 million words per minute. Humans read words at only 600 - 1200 words per minute. Graphics allow us to be able to take full advantage of this channel.

An interesting study was done at the Warton School of University of Pennsylvania showed how powerful graphics can be in getting information to people. The study had 2 speakers each giving a presentation 18 times, 1/2 the time they argued one side and the other time the other side, and graphics were used by both speakers for an equal number of presentations. When graphics were used, 2/3 of the audiences voted for that speaker. Graphics is a really powerful tool that gets results.

A survey conducted by Business and Professional Software in Cambridge, Mass. shows that less than 30% of all the visual material prepared for informal business meetings consisted of graphics of numerical data. Why?

Graphics as a concept must be understood and its value internalized. More people are becoming familiar with graphics as a result of the Personal Computer and its software. Lotus 1-2-3 and other packages have had a great impact on introducing people to graphics.

Graphics software is getting less expensive and easier to use. While in the 1970s it was possible to generate graphs, it was very difficult and expensive. Now graphics terminals are getting less expensive (HP2623A - $3,250) and personal computers are constantly being reduced. There is a trend in the market to eventually bundle graphics into the hardware offering due to the growing understanding and realization of its value.

How can organizations effectively plan for this technology?

The answer to this question will depend upon where each organization is in their development. Many organizations need to educate key managers on the value of graphics. Training is really key. Education on general graphics as well as details on how to design effective charts can be helpful. HP has a publication "Steps to Effective Business Graphics" that provides some key guidelines on how to design charts.

Expert Systems:

Expert systems are gaining in importance and becoming a reality. An expert system is a software program that contains general, experiential, and intuitive knowledge of an expert in some application domain. It uses this knowledge to infer new information and solve problems normally associated with human intelligence and not amenable to traditional, algorithmic computer. Expert systems contain a knowledge base, a reasoning mechanism, and a control mechanism. Unlike databases they arrive at a solution.

What expert systems attempt to do is to make decisions. In a experiment documented by Edward Fiegenbaum, co-author of the <u>Fifth Generation - Artificial Intelligence and Japan's</u>

Computer Challenge to the World, a case in a business class took MBA students 26 hours to find the answer while CEO's took 20 minutes. The difference was that MBAs had to think the problem through, while CEOs provided their best guesses. Expert systems won't work unless experts describe how they make decisions. While understanding / reasoning is difficult to quantify, information scientists are working at these tasks.

Psychologists over the last 50 years have shown that the number of pieces of data the conscious mind can most comfortably handle at any given moment is only about 4. Therefore with computers, this can greatly be improved and will help increase the demand for expert systems and artificial intelligence.

While expert systems are in their early stages of development, it is an interesting trend to track.

A prediction about the effects of expert systems was made by Fiegenbaum noting that number crunching will look very small by the end of the decade and that programmers will be displaced by computer-based intelligent software development workstations. He believes that programmers will need to be retrained to "knowledge engineers" codifying the knowledge gained from the experts.

Gene Amdahl in referring to expert systems noted that "it can be done with 4th generation equipment, although it might be cheaper to do on 5th". There are some "Expert systems" that run on the HP3000s today. However, there is still a lot of development that must take place to move them from their youth into maturity.

Chess machines can now perform at championship levels in playing chess better than 99% of the population. Medical systems have been developed and are being used that allow the diagnosis of diseases. We need to understand that this technology will be a reality in the future and include it in those key areas that are important to your organization. It is important that preliminary work be done on how decisions are made dealing with your organization's concerns.

We need to realize that access to information will improve. It is important to gather information and establish the databases that can be used in future decisions.

These trends are just a few that will dramatically change the office environment. A real software challenge is to integrate office software with current systems as well as develop a software support plan.

It is important to understand current options, technology that will becoming available, where office software is going, and how to integrate office automation into the organization (both organizationally and from a people standpoint). Training needs to be a key factor in the strategy.

III. Programmer / MIS Management:

Where is this role evolving?
How should staffing be developed for this function?
What type of expertise is needed?
What software planning should be done?

There are some definite trends that are taking place that focus on the need for a rethinking of how we view this area.

a. James Martin noted that by 1990 computers will increase in number by 25%, be 10 times as powerful, require 93.1 times as much code to operate resulting in a need for 27.9 million programmers in 1990 (up from approximately 300,000 in 1980). This explosion assumes no increase in productivity which is not realistic. He noted that COBOL and PL/1 will not be the language of the future.
b. EDP Managers loose or quit their jobs on the average of about every 2 1/2 years. The current pressures of trying to satisfy a demanding user community is a definite contributing factor. Inadequate software development tools and techniques contribute.
c. EDP backlog is still growing. Daniel McCracken noted that 2 to 3 year backlogs are becoming the norm and some organizations have 5 year backlogs. This is a definite problem and a trend that can not be allowed to continue.

d. As a result of the backlog, many programs are being written without concern to efficiency, coding techniques, or flexibility. It is normally said that it takes approximately twice as long to write a flexible application as it does to write an inflexible one that only addresses the actual needs of the user.

As a result of technology, the role of the programmer and MIS manager is changing. The following technologies are key:

a. 4th generation programming languages are gaining acceptance. Languages such as HP's TRANSACT allow 1 line of code to take the place of 30 - 50 lines of COBOL code.
b. Tools such as 4th generation programming languages are making the concept of "Prototyping" a reality and a much more feasible techniques to system / program development.
c. Application generators are being developed that can generate a program from having the user answer questions. HP's RAPID products as well as third party tools can contribute to this area.
d. Systems for "System Development" are also being developed. AT&T coined the term "Workbench" that provide aids in system development. HP's Toolset helps with program development.
e. Dictionaries are reducing the amount of code and data structures that need to be maintained.

These trends from both a technological and user-demand view point, will cause a change in this area during the 80s in many organizations.

In order to respond to these trends it appears that this function will need to make some dramatic changes such as the following:

a. Programmers will need to learn new 4th generation languages. While there is the tendency for many programmers to resist learning new languages that are not standard throughout the industry, they will need to learn them or find fewer job opportunities.
b. Prototyping as a concept will need to be learned and internalized into certain business situations. It has great value when users do not know exactly what they want and can not verbalize it or when the objective is to get them involved.
c. Programmers and EDP management must work at developing and buying tools for users to be able to do their own work. They need to provide the guidance in helping users become more productive on such tools as HP's INFORM.
d. Steering committees will lose their importance since end-users will be able to do more of their own work. Rather than have users provide guidance on how to allocate a limited resource, MIS will be helping to train users on how to address their own needs.
e. Organizations will need to look at developing guidelines on when to develop applications using new techniques and languages vs. the current standard used techniques.
f. Organizations need to determine how to implement end-user computing. It will be a real challenge to determine how to integrate it into organizations in an effective manner.

Personal computers are starting to be used in program development as well as being incorporated into software application strategies.

Languages are evolving from 1st Generation Machine Language to 2nd Generation Assembler to 3rd Generation COBOL, BASIC, FORTRAN, etc. to 4th Generation Languages today. While it is easy to say all applications should be developed using 4th Generation languages, it is more complex decision than this. There are some reasons that various languages should/could be used in particular organizations.

Language selection is key. There has been many languages that are being used for different reasons in organizations today. Some basic guidelines are:

a. COBOL is the most widely used business programming language today. Initial testing of COBOL-80 has shown productivity increases of 20% - 25% over COBOL-74 due to some new key verbs such as EVALUATE that reduces the number of nested "IFs". While it is not a standard yet this is a language that should be tracked.

b. ADA is being supported by the U.S. Defense department and some believe will be widely accepted in the future - especially by those companies that want to do business with the U.S. government. An ADA compiler for the HP Series 200 was introduced in 1984.

c. Pascal is being used in many schools. The College Entrance Examination Board chose Pascal as the programming language underlining its Advanced Placement Computer Science Examination in Computer Science.

d. BASIC is used in many secondary schools today and widely used on personal computers.

While this is only a few languages, there are over 100 languages in use today and more are being developed. The trend is toward higher languages.

The future will probably show COBOL maintaining a position in current applications but 4th generation languages gaining. The biggest problem is that it will take a long time for standard 4th generation languages to evolve. 4th Generation language users need to realize that they use more resources.

Organizations need to continue to track new tools in this area and evaluate how to use them. Training and adjustment to new environments will be key challenges.

IV. Database:

Database technology has been very important over the past few years. The basic trend in databases is that they are becoming easier to implement and more flexible.

There are basically 3 types of databases today:

a. Network - (ex. IMAGE)
b. Hierarchical - (ex. IDS)
c. Relational - (ex. INGRESS)

It is important to understand the definition of a database since this is becoming confusing due to the increase of "databases" on personal computers. A database can be defined as "being composed of more than one file and has the ability through commands to change multiple records with one command." This eliminates many so called databases (especially many of those on personal computers) from the formal definition.

There are many hybrids of these databases but new technology (hardware and software) will provide us with some new guidelines into how we use databases. We will be seeing data residing within a file structure and being able to be accessed in many different ways. The same data could be accessed for example using either relational or network database methods.

In selecting a database it is important to evaluate their ability to be modified to fit on other systems. The separation of functions as well as "scalability" will be important in the future. The separation of functions will allow the user interface functions such as SQL, the relational data base method that is becoming a "de facto" standard, to be easily incorporated into the language. If others become more functional and universally used, then they could become incorporated into the database due to modularity.

While relational databases are viewed as easy to use / modify, technology has not yet developed to provide fast effective access. Since there is a need for extensive memory for access, there is still the problem of providing for power failures. What happens to the data and pointers when memory loses power?

David Hsiao, chairman of the computer science department at the Naval Post Graduate School in Monterey and a noted database engine expert, noted that only 10% of the raw data relates to a users query and that only 10% of that relates specifically to the answer. In order to get the correct data significant I/O resources could be used. Database engines are a technology that can be implemented in one of 2 flavors (hardware or software). While a software implementation is possible, Hsiao believes that there should be only a temporary solution.

Britton-Lee shipped its first database engine in 1981 for the HP3000. They have increased the capabilities to increase support from 128 users on 24 computers and 4 disc drives to 4000 users on 64 computers and 16 disc drives. This database alternative has potential when you want to optimize the amount of data that you want to get in a timely manner.

It is important to evaluate database engines when access needs are significant and a separate processor is needed. While in the short term, database engines must be carefully evaluated to determine their viability in a strategy, long term they are expected to be a integral part of Japan's 5th Generation computer system. These systems are expected to have different computers performing symbolic manipulation (arrays), numeric computation, and database access.

Another trend in database technology that is important to be aware of is the move of having the same database run on a variety of systems from the large corporate system to the personal computer. While HP's IMAGE runs on HP1000, HP9000, and HP3000s, the current HP version does not run on the personal computer. Recently a 3rd party developed a package that looks like IMAGE and runs on the personal computer. This helps makes the entire product line compatible.

Today an effective strategy is to use network or hierarchical databases to extract that approximately 10% of the data needed to evaluate and then use relational database technology on personal computers or dedicated mini computers.

As hardware technological innovations are developed that increase processing speed, database engines and relational databases could be an effective solution in handling large data bases.

In the future "truly" relational database technology will be implemented that will be efficient and allow data sets for the same database to to reside on multiple systems. We will have truly distributed data processing.

Database technology is a very important area to track. It must be understood, planned for, and relevant data collected. There will probably be conversions between todays database technology and future database implementations by successful vendors.

V. Data Communications:

According to Stephen McClellan "As software gains ascendancy, a new word will arise to characterize the activities of the industry. The key words will no longer be customization, specialization, and market niche. The new word will be integration. All computer equipment will need to communicate."

Data communication speeds are increasing and having an effect on how we plan for software. Copper wire can be used at transmission speeds of 5000 characters per second, satellites at 100,000 characters per second, and fiber optics at 3,000,000 characters per second. This means that the entire Encyclopedia Britannia can be transmitted between 2 points in less than 2 minutes.

Global communications is expected to increase around 20% over the next decade. Along with this are some interesting developments such as a project among 17 different telecommunications agencies in the Pacific regarding the possibility of the first transpacific optical-fiber cable. The cable would go from California to Hawaii to Guam and then branch off to possibly Japan, the Philippines, or Taiwan. The cable would be capable of carrying approximately 37,500 simultaneous voice conversations.

While there are some very exciting developments taking place in this area, it would be impossible to cover all of them in this paper. Speed will not be as much of a problem in the future. There will also be more communications alternatives. Software must consider them. The following are some areas that are important to understand in planning software strategy:

Standards are being developed. As the ISO (International Standards Organization) OSI (Open Systems Interconnect) layered model open-system architecture becomes more formalized between all 7 layers from the physical to the application, communication will be possible between all hardware vendors. The standards committees have a long way to go but progress is being made. Another "de facto" industry standard is IBM's SNA. While there still appears to be some debate if this will be there long term strategy, it is today their recommended model. Recognizing that there are such "de facto" industry standards such as SNA, HP is also committed to maintain "de facto " industry standards. The reality is that there are some potential conflicts of interest in that it will be more difficult for vendors to protect their hardware base if all systems can be integrated. It is important for users to follow the developments in standards and encourage hardware vendors to follow the standards.

Security is becoming more and more of a concern with networks. There are adequate ways of planning for an effective security system on networks when it is understood and properly implemented. This can be done using a combination of hardware and software. Software techniques on the HP3000 can be effective when properly implemented.

January 23, 1984 issue of Business Week noted that 200 U.S. companies have experimented with some form of tele-commuting, and more than 30 of them have formal tele-commuting programs. Unions have expressed concern about employers taking advantage of their employee and expecting more hours. Business Week expects that tele-commuting will grow rapidly over the next decade, because employers are finding that it generates major productivity increases.

Interfacing between other vendors systems is becoming more of a concern and products are being developed to address this need. There are products that allow the IBM PC and the Wang PC to interface with the HP3000 by running a data communication program that provides HP2622 block mode terminal emulation. Also products are being developed to convert documents between HP WORD and Wang OIS WP, and an HPDESK to IBM PROFS text mailing capability. For communications with DEC products, HP currently supports the UNIX operating software on the HP9000 that provides terminal emulation and file transfer to DEC and other computers running UNIX operating systems. HP is working on a file transfer capability between HP and DEC systems through an IEEE 802.3 local-area network. These are only a few of the interfaces that are available now or will probably be available in the near future. This area will be solved by vendors or 3rd parties because there is a market for products like these.

The integration of personal computers into networks is becoming more of a challenge. Local area networks as well as software to integrate personal computers to central processors will increase in importance during the 80s. HP's AdvanceLink can bring the HP Personal computer user into the HP3000 and extract data and transmit it to the HP personal computer. HPMESSAGE enhances communication between the IBM PC and the HP personal computers. HP supports standards such as IBM's DIA/DCA and DISOSS which helps in multi-vendor communication.

Just like there has been a trend in organizations over the past few years to establish "Database Managers", a new trend is to establish "Network Managers". With the significant advances taking place in both communications hardware and software as well as the increased emphasis on distributed data processing, the need for a "Network Manager" is becoming a real necessity in successful organizations. This person needs to have both technical and managerial skills to appropriately address software needs that span complex organizational structures.

Advances in data communications are one of the most important trends to track. Successful organizations will definitely need to incorporate these advances into their organizational strategy.

VI. Applications:

Yasuo Ishii, of Fujitsu, Ltd. noted at the 1984 Telecommunications Conference in January that "Ninety percent of applications can be handled by 10% of the developed software; but 90% of the software must be developed for the remaining 10% of applications.

This sector of the software industry is becoming more important than ever before according to Stephen McClellan. He noted that 70% of the projected $50 billion sales in 1988 will be application packages (up from $8 billion in 1983 to $35 billion in 1988). Buying packaged software is a trend that has accelerated in the past few years.

It is becoming increasingly difficult to determine what applications to buy when there are over 100 being developed per month (this includes the Personal computer applications).

With the number of external databases being developed, companies need to start thinking about tying into other companies databases such as is being done with the airline reservation systems.

Some of the trends in this area that decisions makers must be aware of are:

a. Vendors are trying to protect their software. There is a trend to restrict source code.
b. Main stream solutions are being developed by computer manufactures that take advantage of all the systems' capabilities including micro-code. According to Stephen McClellan "IBM has been micro-coding or hardwiring more of its systems control program instructions into its mainframes, making it more difficult for third-party vendors." This could have an impact on the software business.
c. On-line applications are increasing. More data can now be kept on-line with the cost per megabyte on large disc/disks being decreased by around 20% per year. The cost per megabyte on HP7933H is around $65 / megabyte.
d. Batch systems are being converted to on-line.
e. Central applications are being distributed. Following the trend to move systems closer to the users.
f. The smaller the organization the more likely that they will buy a software package and the less likely that they will significantly modify it.

The flexibility of the organization's user community and the funds that are available to get an exact requirement match, will be key factors in the determination of how packages fit into an organization. Package decisions should be made in favor of those that contribute to protecting an organization's software investment.

VII. Framework for structuring a software strategy:

Basically we must ask ourselves what constitutes high quality software and then look to develop a future software strategy around these elements based on their importance to your particular organization.

Some of the key characteristics are:

a. reliability
b. flexibility
c. friendly
d. complete
e. supported
f. easy to maintain
g. easy to learn
h. integrated
g. good performance
h. hardware compatibility

Before we focus on detailed planning there are some general guidelines that could be worth considering. These could help provide insight into the objective establishment phase of the plan.

These general guidelines are:

a. In developing a software strategy it is important to keep in mind Pareto's "80/20 rule" as it applies to software. Werner Frank, a software consultant, applied the rule to software noted that

(1) "Eighty percent of the computer user's requirements associated with a given application are satisfied by 20% of that application's usual features and functions."

(2) "Eighty percent of the overall user population can satisfy 100% of their needs with 20% of the typical software application's capability".

(3) "...users employ 20% of a word processing product's capability during 80% of the software's execution time; during the remaining 20% of the time some of the less used, more obscure features are employed.

b. More's Law that "Information retrieval systems tend not to be used when it becomes more painful to get information than to live without it" is also interesting. It is important to make sure that software is designed / procured that will be used and solve needs - Not contribute to pain.

c. Software ergonomics focuses on designing and procuring solutions with the user in mind. Ergonomics comes from the Greek word "ergon" for "work" and "nomos" for "law". It is the science of adapting the work-place to the worker focusing on safety, comfort, and efficiency. It is important to apply this to software. Strategies need to have software ergonomics incorporated into them.

In developing a plan there appears to be some basic steps that can help contribute to success. By following the steps in the outline below and addressing the questions, a plan will evolve.

Software Planning Steps:

a. Requirements Definition:

The first step is to understand your current environment. Before any steps can be made to determine where you want to go, it is important to know where you are today. Some of the key areas that need to be addressed are: (1) What does my current environment look like? (2) What systems do we have in place? (3) How much support is the data processing department getting? (4) How committed is top management to automation? (5) To what degree do users want customized applications: (5a) How much are they willing to compromise on their exact desires to fit a package? (5b) How much are they willing to pay for an exact fit? (6) What languages are being used? (7) What is currently being supported? (8) Competitive pressures - to what degree are competitors automated? (9) How great is the demand for information? (10) How great is the current backlog? (2 years?. . .5 years?) (11) How technically sophisticated and interested are the users? (12) What software security is in place?

b. Objectives:

From an understanding of the current environment, realistic objectives for your organization can be established. Some of the key questions that can help determine the objectives for your software strategy are: (1) Budget potential? (is money a problem for justified applications?) (2) What is the expected pay-back period? (3) How urgent are the systems perceived to be? (4) Projected EDP support expenditures - What % of sales, etc.? (4a) Higher - more reliance on EDP and tools for programmer. (4b) Lower - more reliance on user and tools to support user. (5) Degree of state of the art desired? (6) Emphasize technology . . .make it a competitive advantage? (7) How fast does top management want to automate? (8) How many vendors are on your short list?

c. Plan:

As a result of answering the above questions a plan can be formulated. Some of the key elements of the plan that must be addressed: (1) Training required to obtain objectives addressing the needs of users, programmers, technical staff, MIS management, and others.; (2) The plan should be a "phased in" process with explicit list of milestones that will help in accomplishing the objective.; (3) Critical applications - what needs to be addressed and when.; (4) Languages wanting to phase into and when they should be used.; (5) Under what conditions should organizations use 4th generation programming languages vs. 3rd generation vs. buying packages?; (6) Criteria for evaluating software packages.

d. Implementation:

The following are some general guidelines that should be in an implementation plan: (1) Phased in software plan. (2) Start with critical applications. (3) Do not try to automate all areas overnight. (4) Follow a plan and remember the adage "When you fail to plan, you plan to fail".

e. Review:

Plans and implementations need monitoring. There needs to be a periodic review process incorporated into the strategy. The following are some key points to consider: (1) Need to be done at least once per year. (2) See where we have been and where we are going. (3) Is progress being made toward the objective?

Summary:

It is becoming evident that software is key and planning for it is important. There are a number of factors that must be considered. Developing an understanding of trends, technology, and terminology will help assure success. Software does link man to his machine but to make man effective, it must be planned for and addressed in a systemic fashion. While it is impossible to discuss all the factors and trends that organizations need to understand, hopefully some insight was gained from this paper that will help you design your software strategy / plan.

It is always helpful to have your plan reviewed by a number of other individuals. The key to developing a sound strategy that can be shared with others and discussed from a sound business standpoint is to formally document your software environment and organizational objectives. From this, consultants who understand software can help tailor a plan for your organization. Good luck.

Notes / Acknowledgments:

The data for this paper came from a number of sources including articles from Computerworld, Interact, the HP Chronicle, Software News, Business Week, unpublished studies, as well as 3 books that can be interesting and valuable to those interested in developing a strategy.

1. Trends in Information Technology: 1985 published by Arthur Anderson.
2. Fifth-Generation Computing by Edward A. Fiegenbaum and Pamela McCorduck.
3. The Coming Computer Shakeout by 1990 by Stephen McClellan

Biography:

Bill Franklin, BS, MBA, CDP is an Area Marketing Representative with Hewlett-Packard's Information Systems Marketing Center in Cupertino, California. Bill has been with HP for 7 years (15 years in the computer industry) and has worked in many functions both managerial and technical in MIS and marketing.

STANDARDS FOR ERGONOMIC INTERACTIVE PROGRAMS

Rudi Huysmans
SYDES NV, Mechelen, Belgium

Abstract :

This paper describes means and methods for the realisation of ergonomic interactive programs.

- To ease the way for the user, all screen views should have the same layout, all applications should be demandable in a similar way, all function keys should be at the same place all the time, etc.

- In this text we shall go deeper into some of the most important elements that contribute towards this standardisation.

In the first chapter, I am going to talk about the application supervisory system and the way to run the desired application.

The next chapter will be about screen design and related subjects such as the use of function keys, security, scrolling mechanism and so on.

In the third chapter, we shall see how to implement these standards in a quicker and more intelligent way.

1. The application supervisory system

1.1. Definition

The application supervisory system guides the user through his applications without requiring the user to have any knowledge of MPE. It is rather like a coat-stand where the user puts his applications on.

1.2. Concepts

From the point of view of the user, the application supervisory system offers a hierarchical path through the user's application using a set of menus and submenus. An experienced user can enter the taskname and avoid the need to use the tree of options. An inexperienced user, on the other hand, can walk through the menu tree and use a help facility to guide his choice.
The user can run programs, stream jobs, and execute MPE commands without directly using the MPE Command Interpreter. The system allows the management of the menus without the need to recompile the application programs.

1.3. Standard ways to select a screen.

There are different ways to choose a screen from an application.

- Walking through a set of menus to find the desired application.

- Giving the final option without having to go down the tree of options.
  Each form (screen) has a name and the form name can be used to give up a final selection.

- Invoking a form : The named form is called and displayed on the screen.
  After the form is finished, the current form is re-displayed and continued at the field following the INVOKE.

1.4. Features of a good and ergonomic application supervisory system.

The application supervisory system :

- is a utility to handle processes. If you want to add, delete or modify a selection for a given user, you do not have to recompile programs.

- secures user accesses. It identifies the user with his password and the user himself is responsable for handling his own password.

- guides the user's choice, prints or reprints a menu, immediately accepts the user's final option and has a help facility.

- gives the possibility to print different end menus depending upon the capability of the user after commencing with the same original menu file.

- allows parameters such as LIB, INFO, ENTRYPOINT, PARM, $STDIN,$STDLIST to be given at run time.

- tests if a selection may be used.

- increases performance because it can verify if a process is still alive and if it is, re-activate it (if foreseen in the user program code at program termination time). Then the process is still allocated, and the files are still open,... This is the best way to get a good response time.

- uses a logging system to ensure that a complete history is maintained of all special events.

- gives the opportunity to have all messages in the language of the user.

1.5. Example

HELLO - 3000 (1) is a good example of an ergonomic application supervisory system. It has the qualities I mentioned before and forms the base for standardized user's environment.

2. Standard for screen design

2.1. Designing a screen

The most common screens exists of a Keyfield with related attributive fields.
In this paper an elegant solution for working with such screens is discribed.

The actions that can be defined on these screens are :

- Add a new record. This implies that the key value does not yet exist.
- Read an existing record. This implies that that the key value exists.
- Update an existing record, implying that the record is already read.
- Delete an existing record. This also implies that the record has been read.

It is advisable to implement such screens as follows :

```
while
   prompt for the key value
   if the key value exists
      activate the delete key
      display the attribute fields
      while there are attribute fields
         prompt for a changed value
         next field
      endwhile
      if there are attribute fields changed
         update the record
      endif
      de-activate the delete key
   else
      while there are attribute fields
         prompt for a value
         next field
      endwhile
      add the record
   endif
endwhile
```

features
- this method is self explaining for the user
- the delete key is only activated when necessary

2.2. Implementing a security mechanism

The application supervisory system tests if the screen selection is permitted. It also passed a security value to the application as follows :

```
accesscode :  R = access allowed on read level
              U = access allowed on update level  (includes  R)
              W = access allowed on write level   (includes R,U)
              D = access allowed on delete level  (includes R,U,W)
```

The accesscode can easily be used in the previous algorithm as follows:

```
    while
       prompt for the key value
       if the key value exists
          if access code >= "D"
              activate the delete key
          endif
          display the attribute fields
          if access code>= "U"
              while there are attribute fields
                  prompt for a changed value
                  next field
               endwhile
               if there are attibute fields changed
                  update the record
               endif
               de-activate the delete key
           endif
       else
           if access code >= "W"
               while there are attribute fields
                  prompt for a value
                  next field
               endwhile
               add the record
           else
               error "key value does not exist"
           endif
       endif
```

2.3. The scrolling mechanism

2.3.1. Definition: scrolling window

A scrolling window (in short : scroll) is a data-window through which we can see or fill in a piece of a table at a time.
The data on a scroll-line can be of different types, but the lines itself are of the same kind. The user can run over the entire table (forward or backward), add lines, change and delete.
If the table is larger that 80 positions, the user must be able to run through the scroll horizontally. The user can decide by means of function keys, which scroll fields should appear on the screen.

2.3.2. Actions on a scroll

A lot of standard actions can be offered to the programmer by use of a scroll:

INSERT : inserts an element in the scrolling window, above the current line
DELETE : deletes the current scroll element from the scrolling window.
NEXT : advances the cursor to the next element
PREV : sets the cursor back to the previous element
NEXTP : advances the cursor to the next page
PREVP : moves the cursor to the previous page
BEGIN : puts the cursor on the first field of the first element
END : moves the cursor to the first field of the element with the largest number accessed during this run.
SHOW : defines the order in which the fields of the scrolling area will be arranged.
SORT : sorts the scroll area in ascending sequence
COMPUTE: performs the specified compute statement on all lines of the scroll area of the given page.
SEARCH : searches a field value in the given scrolling field

The best way to offer this possibilities to the user is by using function keys.

2.4. The use of function keys

2.4.1. Definition : function key

There are 2 different sorts of function keys : physical function keys and logical ones.

1. Physical function keys (FKEYS)

The physical function keys are the eight special, physically existing keys (F1, F2, ..) at the head of the keyboard.
To each key a label on the screen is connected.

On most HP screens this function keys are programmable. It is possible to attach a certain action to these function keys.

2. Pseudo keys

As the number of available function keys is always limited to eight, we can define the so called pseudo keys (PKEYS).
A pseudo key is activated through a 1 or 2 sequence defined by the programmer and acts like a function key.

2.4.2. General hints when using function keys

- use as much as possible the same function key for the same action. This is much simpler for the user and the programmer can reduce the same code program or make use of a macro-processor.

- use as much as possible the physical function keys and put in the label an ergonomic justified text.

- use as little as possible the pseudo function keys, unless the character string is an obvious abbreviation of the action, of if you use them only once.

- multiple sets of function keys can be performed by a single function key stroke.

- make sure that a function key is only active when the key can be used.

3. Standards and their implementation

All the examples are written in the SYDAID programming language (3).

3.1. Enabling and disabling a function key

The following commands are available

3.1.1. Enabling physical function keys

FKEY Keynumber, field, string, label

sets up the programmable function key with number keynumber. The key is loaded with string and label is displayed in the key label.
Key number can be a constant between 1 and 8 or a field that changes at run time to a value between 1 and 8.
'Field' indicates the field to be executed whenever the function key is hit.

E.g. FKEY 1,deleterout,"%","DELETE"

3.1.2. Enabling pseudo keys

PKEY Key number,routine,string

3.1.3. Disabling function keys

```
CLEAR  FKEY n
CLEAR  PKEY n
CLEAR  FKEYS
CLEAR  PKEYS
```

3.2. General function keys

Layout

```
 ___________________________________________________________________________
|   ACCOUNTING                  CUSTOMERS                        26/06/84  |
|                                                                          |
|              customer number  :  00023                                   |
|              name             :  SYDES                                   |
|              address          :  BRUSSELPOORTSTRAAT 8                    |
|              ZIP              :  2800                                    |
|              CITY             :  MECHELEN                                |
|                                                                          |
 ___________________________________________________________________________
| PRINT | HOME | HELP | EXIT |    |        |        |       |DELETE |
|__________________________________________________________________________|
```

physical function keys

The first 4 function keys are general function keys. The remain active during the whole process.

- 1 - PRINT : prints the current form on a file
- 2 - HOME : go to he first field
- 3 - HELP : invoke another form
- 4 - EXIT : the form is considered to be finished. The system prompts for a new form name.
- 8 - DELETE : to delete the record.

pseudo function keys

- "dd" : start debug facility
- "/R" : refreshes the screen

Implementation in SYDAID

```
subrouting pkeysys1
  debug

subroutine pkeysys2
  refresh

subroutine fkeysys1
  print now

subroutine fkeysys2
  goto customer number

subroutine fkeysys3
  if $invoke > 1  then message 1000
  invoke "HELP"

subroutine fkeysys4
```

```
  if $invoke > 1
  stop screen
else
  clear formid
  goto formid
endif

subroutine setsys
  fkey 1,fkeysys1,'!','PRINT'
  fkey 2,fkeysys2,'@','HOME'
  fkey 3,fkeysys3,'#','HELP'
  fkey 4,fkeysys4,'$','EXIT'
  clear fkey 5
  clear fkey 6
  clear fkey 7
  clear fkey 8
  pkey1,pkeysys1,'dd'
  pkey2,pkeysys2,'/R'
```

Enabling the function keys will be done by using the statement 'perform SETSYS'.

3.3. Binary questions

The possible answers on binary questions can be put in function keys as follows

```
 ________________________________________________________________________
| ACCOUNTING               CUSTOMERS                          26/06/84 |
|                                                                      |
|               customer number  :  00023                              |
|               name             :  SYDES                              |
|               address          :  BRUSSELPOORTSTRAAT 8               |
|               ZIP              :  2800                               |
|               CITY             :  MECHELEN                           |
|                                                                      |
| Modify the current data ?                                            |
 ________________________________________________________________________
| PRINT | HOME | HELP | EXIT |    |        |        |       |DELETE |
|______________________________________________________________________|
```

3.4. Scroll function keys

Layout

```
 ________________________________________________________________________
|  ACCOUNTING          CUSTOMER  OVERVIEW                     26/06/84 |
|                                                                      |
|     NO      NAME      ADDRESS             ZIP     CITY               |
|     ----    ----      -------------       ----    ----------------   |
|     0018    JAN       DORPSTRAAT 10       1000    BRUSSEL            |
|     0019    PIET      STATIONSTRAAT 14    2000    ANTWERPEN          |
|     0020    KAREL     NIEUWSTRAAT 1       9000    GENT               |
|       "       "          "                 "       "                 |
|                                                                      |
|                                                                      |
|______________________________________________________________________|
|NEXT |PREVIOUS| BEGIN|  END   |  ADD  | DELETE |  EXIT | STANDARD  |
|LINE |  LINE  |SCROLL| SCROLL | LINES |  LINES | SCROLL|   KEYS    |
|______________________________________________________________________|
|PRINT|  HOME  | HELP |  EXIT  |       |        |       |           |
|     |        |      |        |       |        |       |           |
|______________________________________________________________________|
```

Whenever the scroll is entered, following function keys are activated:

```
- 1 -   next line
- 2 -   previous line
- 3 -   begin scroll
- 4 -   end scroll
- 5 -   add lines
- 6 -   delete lines
- 7 -   exit scroll
- 8 -   set 1/set 2    selects masterset or scrollset
```

SYDAID code

```
subroutine fkeyscroll1
  scroll next

subroutine fkeyscroll2
  scroll prev

subroutine fkeyscroll3
  scroll top
  scroll begin

subroutine fkeyscroll4
  scroll bottom
  scroll end

subroutine fkeyscroll5
  scroll add

subroutine fkeyscroll6
  scroll delete

subroutine fkeyscroll7
  goto choice

subroutine fkeyscroll8
  perform setsys
  fkey 8,scrollstart,'*','SCROLL   KEYS'

subroutine setscroll
  fkey 1,fkeyscroll1,'!','  NEXT     LINE'
  fkey 2,fkeyscroll2,'@','PREVIOUS   LINE'
  fkey 3,fkeyscroll3,'#','  BEGIN   SCROLL'
  fkey 4,fkeyscroll4,'$','  END     SCROLL'
  fkey 5,fkeyscroll5,')','  ADD      LINE'
  fkey 6,fkeyscroll6,'^',' DELETE    LINE'
  fkey 7,fkeyscroll7,'&,'  EXIT     SCROLL'
  fkey 8,fkeyscroll8,'*',STANDARD    KEYS'
```

Enabling the function keys will be done by using the statement :
perform SETSCROLL

3.5. Scroll pseudo keys

Possible scroll pseudokeys are :

```
A  :  Add line at the end of the scroll
B  :  Begin scroll
D  :  Delete line
E  :  End scroll
F  :  Find a given value
I  :  Insert line
L  :  go to a given line number
N  :  go to the next page of the scroll
P  :  go to the previous page of the scroll
S  :  sort the scroll
^  :  to to the previous line
V  :  go to the next line
.  :  copy the       value of the previous line.
```

Implementation in SYDAID

```
A : scroll add
B : scroll begin
D : scroll delete
E : scroll end
F : accept findfield
    search findfield in scrollfield
I : scroll insert
L : accept  line number
    go to *[ line number ]
N : scroll nextp
P : scroll prevp
S : scroll sort
V : scroll next
^ : scroll prev
. : move *[ $occurs-1 ] to *
```

3.6. Implementation by use of a substitution machine

It is useful to compile a library made up from standard coding (macros) to implement function keys, screen layouts and handle database inconsistencies.

The final text of the program is obtained through expansion of all referenced macronames by means of a 'substitution machine'.

METRO/3000 (4,5,6) is a Sydes software product answerring to this demands.

4. Conclusion

The use of a good application supervisory system improves the ergonomic quality of standards for application software.

By using function keys and scrolling mechanism we increase the human/computer interface considerably.

This standardisation can be simplified by using a macro-processor :

5. References :

[1] : HELLO 3000 reference manual version A.04.00;
SYDES NV, Mechelen, Belgium; july 1984

[2] : Het gebruik van funktietoetsen in SYDAID;
R. Huysmans; SYDES Newsletter 1:2; sept. 1984

[3] : SYDAID reference manual version A.05.00;
SYDES NV, Mechelen, Belgium; july 1984

[4] : METRO 3000 reference manual version A.02.00
SYDES NV, Mechelen, Belgium; july 1984

[5] : An intelligent processor for 'text-creating' users,
H. Vanstappen; to be published in the proceedings of the
HP 3000 IUG Amsterdam Conference '85

[6] : Preprocessing, het middel voor standaardisatie en
productiviteitsverhoging;
R. Huysmans; SYDES Newsletter 1:3; nov. 1984

Biography

Rudi Huysmans was born in 1953 in Mortsel near Antwerp.
After High school he studied civil engeneer in the computer science at the Catholic University of Leuven. He joined SYDES in 1982 as commercial project leader. He is Vice Chairman of the Belgium National Users Group. Off business hours he works as teacher in the computer science.

APPLICATION PROTOTYPING - A METHODOLOGY FOR PERFECTING THE MAN-MACHINE INTERFACE

Orland Larson
Hewlett-Packard, Cupertino, USA

Summary

One of the most imaginative and successful techniques for clarifying user interfaces and generally improving the productivity and effectiveness of application development is a methodology called APPLICATION PROTOTYPING.

With waiting time for new applications running into several years and those applications failing to meet the users needs, managers as well as users have been searching for more efficient and effective approaches to systems development.

Prototyping, as an application system design and development methodology, has evolved into a real option for both the MIS professional and the user.

This paper reports on the growing body of knowledge about prototyping. It begins by reviewing the changing role of data processing, the challenges facing the MIS organization, and the traditional approach to application development. It then defines prototyping followed by the step-by-step prototype development process. The advantages and disadvantages, as well as the cost and efficiency of prototyping, will be discussed followed by the essential resources neccessary to effectively prototype applications. In conclusion, to illustrate the benefits of prototyping, the paper will present success stories of systems developed using the prototyping approach.

INTRODUCTION

The Changing Role of Data Processing

The data processing department has changed dramatically since the 1960s, when application development as well as production jobs were usually run in a batch environment with long turnaround times and out-of-date results.

The 1970s were a period of tremendous improvement for the data processing environment. One of the key developments of that period was the development and use of Data Base Management Systems (DBMS). This provided the basis for on-line, interactive applications. In addition, computers and operating systems provided programmers the capability of developing application programs on-line, while sitting at a terminal and interactively developing, compiling, and testing these applications. The end user was also provided with easy-to-use, on-line inquiry facilities to allow them to access and report on data residing in their data bases. This took some of the load off the programmers and allowed them to concentrate on more complex problems.

During the 1980s, the data base administrator and MIS manager will see increased importance and use of centralized data dictionaries or "centralized repositories of information about the corporate data resources." Simpler and more powerful report writers will be used by the end user and business professional. The programmer will see the trend towards the use of high-level, transaction processing languages, also known as fourth generation languages, to reduce the amount of code required to develop applications. Finally, the tools have been developed to effectively do application prototyping, which will provide benefits to the end user as well as the application programmer and analyst.

Throughout the 70s and 80s, information has become more accurate, reliable, and available, and the end user or business professional is becoming more actively involved in the application development process.

Challenges Facing MIS

One of the MIS manager's major problems is the shortage of EDP specialists. A recent Computerworld article predicted that by 1990 there will be 1/3 of a programmer available for each computer delivered in this country. Software costs are also increasing because people costs are going up and because of the shortage of skilled EDP specialists. The typical MIS manager is experiencing an average of two to five years of application backlog. This doesn't include the "invisible backlog," the needed applications which aren't even requested because of the current known backlog. In addition, another problem facing MIS management is the limited centralized control of information resources.

The programmer/analyst is frustrated by the changeability of users' application requirements (typically, the only thing constant in a user environment is change). A significant amount of programmers' time is spent changing and maintaining users' applications (as much as 60 to 80 percent of their time). Much of the code the programmer generates includes the same type of routines such as error checking, formatting reports, reading files, checking error conditions, data validation, etc. This can become very monotonous or counterproductive for the programmer.

The end user or business professional is frustrated by the limited access to information needed to effectively do his/her day-to-day job. This is especially true for those users who know their company has spent a great deal of money on computer resources and haven't experienced the benefits. The users' business environment is changing dynamically and they feel MIS should keep up with these changes. MIS, on the other hand, is having a difficult time keeping up with these requests for application maintenance because of the backlog of applications and the shortage of EDP specialists. Once the user has "signed off" on an application, he is expected to live with it for a while. He is frustrated when he requests what he thinks is a "simple change" and MIS takes weeks or months to make that change.

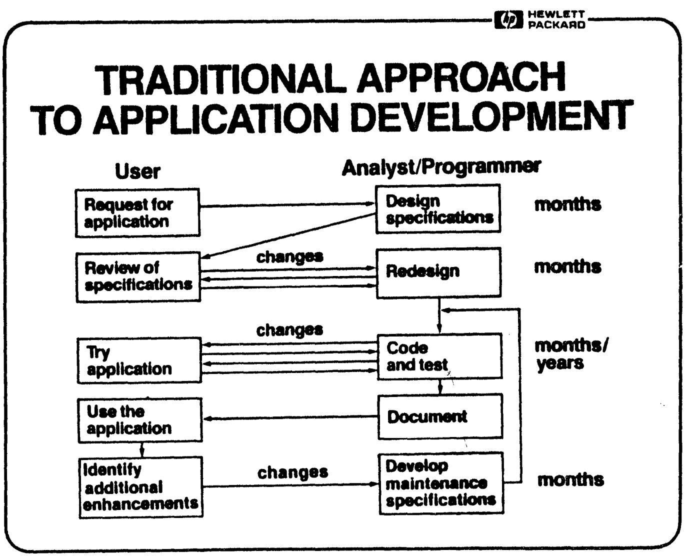
Traditional Approach to Application Development

There are some myths concerning traditional application development:

- Users know exactly what they want
- Users can effectively communicate their needs to MIS
- Users needs never change.

The traditional approach to application development has serious limitations when applied to on-line, interactive information systems that are in a state of constant change and growth. Communications among the user, analyst, programmer, and manager tend to be imprecise, a detailed analysis prolongs the process to the annoyance of the user, and specifications are either ambiguous or too voluminous to read. To compound this problem, the user is often requested to "freeze" his requirements, and subsequent attempts at change are resisted.

Let's review the traditional approach to application development.



- The user first requests an application and then an analyst or programmer is assigned to the application.

- The analyst or programmer takes the oftentimes sketchy user's specifications and designs more complete specifications.

- The user then reviews the analyst's interpretations of his specifications and probably makes additional changes.

- The analyst redesigns his specifications to adapt to these changes. (By this time, several days, weeks or months have gone by.)

- The user finally approves the specifications, and a team of analysts and programmers are assigned to develop, test and document the application.

- The user finally tries the application. Months or years may have gone by before the user gets his first look at the actual working application.

- The user, of course, will most likely want additional changes or enhancements made to the application. This is called adjusting the application to the "real world".

- Depending on the extent of these changes, additional maintenance specifications may have to be written and these program changes coded, tested and documented.

- The total application development process may take months or years, and the maintenance of these applications may go on forever.

In summary, the traditional approach to application development results in long development times, excessive time spent on maintenance, a multi-year backlog of applications, limited control and access to information, and applications that lack functionality and flexibility and are very difficult to change. The question is: "Can we afford to continue using this approach to application development?"

Prototype Defined

According to Webster's Dictionary, the term prototype has three possible meanings:

1) It is an original or model on which something is patterned: an archetype.
2) A thing that exhibits the essential features of a later type.
3) A standard or typical example.

J. David Naumann and A. Milton Jenkins in a paper on software prototyping (see reference 7) believe that all three descriptions apply to systems development. Systems are developed as patterns or archetypes and are modified or enhanced for later distribution to multiple users. "A thing that exhibits the essential features of a later type" is the most appropriate definition because such prototypes are a first attempt at a design which generally is then extended and enhanced.

Roles in the Prototyping Process

There are two roles to be filled in prototyping -- the user/designer and the systems/builder. These roles are very different from the traditional user and analyst/programmer roles under the traditional approach. The terms "user/designer" and "systems/builder" emphasize these differences and denote the functions of each participant under the prototyping methodology. Remember it is the user who is the designer of the application system and the systems professional who is the builder.

The user/designer initiates the process when he/she conceives of a problem or opportunity that may be solved or exploited by the use of an information system. The user/designer typically must be competent in his/her functional area (many times he/she is a manager) and usually has an overall perspective of the problem and can choose among alternative solutions. However, he/she requires assistance from the MIS organization.

The systems/builder is assigned by the MIS organization to work with the user/designer and is competent in the use of the available prototyping tools and knowledgeable about the organizations data resources.

Prototyping Process

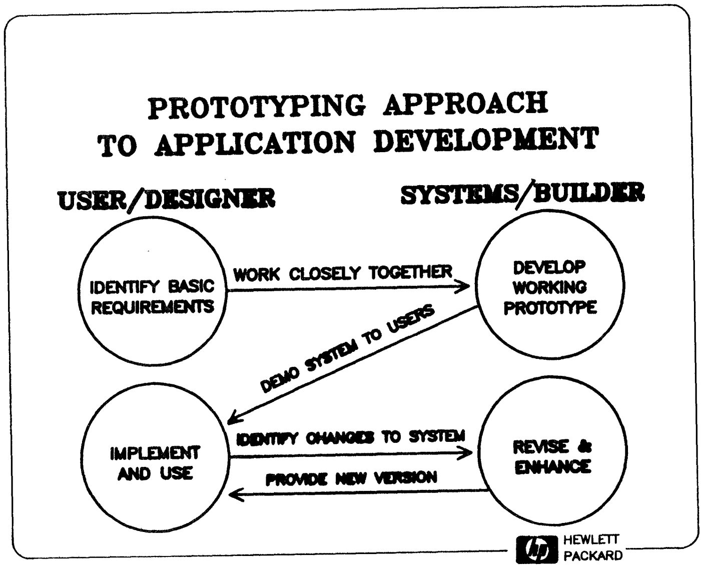
The process of application prototyping is a quick and relatively inexpensive process of developing and testing an application system. It involves the user/designer and the systems/builder working closely to develop the application. It is a live, working system; it is not just an idea on paper. It performs actual work; it does not just simulate that work. It can be used to test assumptions about users' requirements, system design, or perhaps even the logic of a program.

Prototyping is an iterative process. It begins with a simple prototype that performs only a few of the basic functions of a system. It is a trial and error process - build a version of the prototype, use it, evaluate it, then revise it or start over on a new version, and so on. Each version performs more of the desired functions and in an increasingly efficient manner. It may, in fact, become the actual production system. It is a technique that minimizes the dangers of a long formal analysis and increases the likelihood of a successful implementation.

Prototyping Methodology/Model

The prototyping methodology in general, is based on the following proposition: "People can tell you what they don't like about an existing application easier than they can tell you what they think they would like in a future application."

Prototyping an information system can be viewed as a four -step procedure.



Step 1. User/designer identifies the basic information requirements:

- Write a brief, skeleton-like statement that captures the essential features of the information requirements.
- User/designer and systems/builder work closely together.
- Concentrate on users' most basic and essential requirements.
- Define data requirements, report formats, screens, and menus.
- Need not involve lengthy written specifications.
- For larger systems, a design team may need to spend a few weeks preparing a first-effort requirements document.

Step 2. Systems/builder develops the initial prototype:

- Systems/builder takes the notes developed in the user discussions and quickly builds the menus and dialogs.
- A data dictionary would be useful at this time.
- Design and/or define data base and load subset of data.
- Make use of defaults and standard report formats.
- Write required application modules using a fourth generation language.
- Prototype performs only the most important, identified functions.

Step 3. Users implement and use the prototype to refine requirements:

- Systems/builder demonstrates prototype to small group of users.
- Users gain hands-on experience with application.
- Users are encouraged to make notes of changes they would like made.
- Users discuss and prioritize desired changes.

Step 4. Systems/builder revises and enhances the prototype:

- Systems/builder modifies the prototype to correct undesirable or missing features.
- May require modification or redesign of data base, changes to existing programs and/or additional program modules.
- Deliver back to users quickly.

NOTE: Steps 3 and 4 are repeated until the system achieves the requirements of this small group of users. Then either introduce it to a larger group of users for additional requirements or if enough users are satisfied, demo it to management to gain approval for the production system.

When to Use Prototyping

1. To clarify user requirements:

   - Written specs are often incomplete, confusing, and take a static view of requirements.
   - It is difficult for an end user to visualize the eventual system, or to describe his/her current requirements.
   - It is easier to evaluate a prototype than written specifications.
   - Prototyping allows, even encourages, users to change their minds.
   - It shortens the development cycle and eliminates most design errors.
   - It results in less enhancement maintenance and can be used to test the effects of future changes and enhancements.

2. To verify the feasibility of design:

   - The performance of the application can be determined more easily.
   - The prototype can be used to verify results of a production system.
   - The prototype can be created on a minicomputer and then that software prototype may become the specifications for that application which may be developed on a larger mainframe computer.

3. To create a final system:

   - Part (or all) of the final version of the prototype may become the production version.
   - It is easier to make enhancements, and some parts may be recoded in another language to improve efficiency or functionality.

When Not to Use Prototyping

1. When an application requires a standard solution that already exists and is available at a reasonable cost from a software supplier.

2. When you don't have a good understanding of the tools available to prototype.

3. When the organization's data and software resources are not well organized and managed.

4. When MIS management is unwilling to develop a staff of professional systems/builders.

5. When the user/designer is unwilling to invest his/her time in the development of the application system.

Potential Problems

One of the initial problems typically encountered is the acceptance of the prototyping methodology by the systems people. This is due to the fact that people naturally tend to resist change. It may also encourage the glossing over of the systems analysis portion of a project. It is not always clear how a large complex system can be divided and then integrated. Initially, it could be difficult to plan the resources required to prototype (people, hardware and software). It may be difficult to keep the systems staff and users abreast of each version of the system. Programmers may tend to become bored after the nth iteration of the prototype. Testing may not be as thorough as desired. It might be difficult to keep documentation on the application up to date because it is so easy to change.

Even with these concerns, prototyping provides a very productive working relationship for the users and the builders. So it behooves all data processing management to learn to use this powerful tool creatively and to manage it effectively.

THE ADVANTAGES OF PROTOTYPING GREATLY OUTWEIGH THE PROBLEMS!

Advantages of Prototyping

One of the main advantages of application prototyping is that this methodology provides a capability to quickly respond to a wide variety of user requests. It provides a live, functioning system for user experimentation and accommodates changes in a dynamic user environment. One interesting aspect of this approach is that users are allowed and even encouraged to change their minds about an application's interfaces and reports, which is a very rare occurrence during the traditional approach. Maintenance is viewed right from the beginning as a continuation of the design process. Finally, prototyping provides an effective use of scarce systems/builders. One or a limited number of systems/builders will be required for each prototyping project; and while users are testing one prototype, the systems/builder can be working on another.

Cost and Efficiency

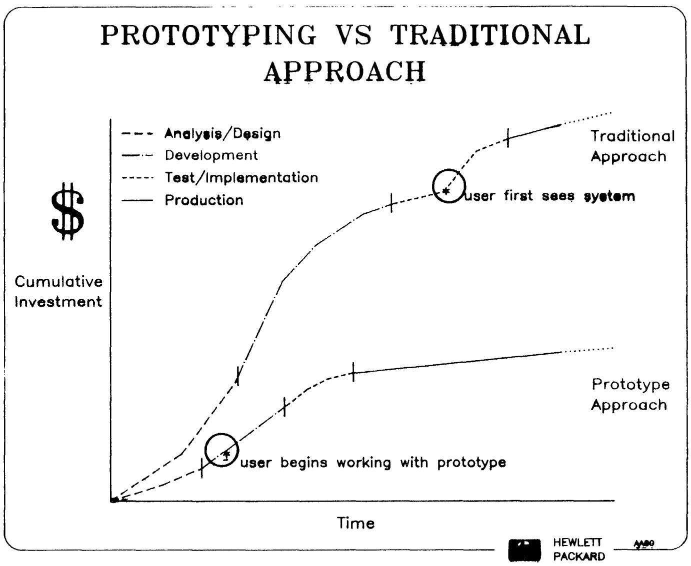
It has been found that there is an order of magnitude <u>decrease</u> in both development cost and time with the prototyping methodology.

It is often difficult to estimate the cost of prototyping an application system because the total costs of development, including maintenance, are usually lumped together. The cost of implementing the <u>initial</u> system is much lower than the traditional approach (typically less than 25%).

However, software prototyping could be expensive in the following ways:

- It requires the use of advanced hardware and software.
- It requires the time of high-level users and experienced systems staff.
- It requires training of the systems staff in the use of prototyping and the associated tools.
- Application run-time efficiency may be compromised.

The main thing to remember is that the main focus of prototyping is not so much efficiency but effectiveness.



Essential Resources

The following are the essential resources to effectively do application prototyping:

1. Interactive Systems

   - Hardware and Operating System - When doing application prototyping, both the builder and the system must respond rapidly to the user's needs. Batch systems do not permit interaction and revision at a human pace. Hardware and associated operating systems tailored to on-line interactive development are ideal for software prototyping.

2. Data Management Systems

   - A Data Base Management System provides the tools for defining, creating, retrieving, manipulating, and controlling the information resources. Prototyping without a DBMS is inconceivable!

- A Data Dictionary provides standardization of data and file locations and definitions, a cross reference of application programs, and a built-in documentation capability. These are essential to managing the corporate resources and extremely useful when prototyping.

3. Generalized Input and Output Software

- Easy to use data entry, data editing, and screen formatting software are extremely helpful in the application prototyping process to allow the programmer to sit down at a terminal with a user and interactively create the user's screens or menus.

- Powerful, easy-to-use report writer and query languages provide a quick and effective way of retrieving and reporting on data in the system. A report writer that uses default formats from very brief specifications is most useful in the initial prototype.

- A powerful graphics capability can be extremely useful for the display of data in a more meaningful graphical format.

4. Very High Level (Fourth Generation) Languages

- Traditional application development languages such as COBOL may not be well suited for software prototyping because of the amount of code that has to be written before the user sees any results.

- Very powerful fourth generation languages that interface directly to a data dictionary for their data definitions are ideal. One statement in this high level language could realistically replace 20-50 COBOL statements. This reduces the amount of code a programmer has to write and maintain and speeds up the development process.

5. Documentation Aids

- Tools to aid in the maintenance of programs written in a 4GL.

- Tools to aid in maintaining user documentation on-line.

6. Libraries of Reuseable Code

- A library of reusable code to reduce the amount of redundant code a programmer has to write is an important prototyping resource.

- This code could represent commonly used routines made available to programmers.

Hewlett-Packard's Tools for Prototyping

Hewlett-Packard is one of the few vendors that supplies the majority of the tools needed to effectively do software prototyping.

* Interactive Systems
  - HP 3000 (All Series)
  - MPE Operating System
* Data Management Systems
  - IMAGE/3000
  - KSAM/3000
  - MPE files
  - DICTIONARY/3000
* Generalized Input/Output Software
  - VPLUS/3000
  - QUERY/3000
  - REPORT/3000
  - INFORM/3000
  - HPEASYCHART
  - DSG/3000
* Very High Level Languages
  - TRANSACT/3000
* Documentation Aids
  - HPSLATE
  - HPWORD
  - TDP/3000

Note: There are several additional excellent prototyping tools available from HP third-party vendors which are too numerous to mention here. Please consult the Hewlett-Packard Business Systems Software Catalog (Part No. 3000-90251) for more information.

Summary

Prototyping is truly a "state-of-the-art" way of developing applications.

- Software prototyping promotes an interactive dialogue between the users and the programmer, which results in a system being developed more quickly, and results in an interactive development approach which is friendlier for the end user.

- The prototype provides a live working system for the users to experiment with instead of looking at lengthy specifications.

- The users are provided with an early visualization of the system which allows them to immediately use it.

- The users are allowed and even encouraged to change their minds about user interfaces and reports.

- Maintenance is viewed right from the beginning as a continuous process and because the prototype is usually written in a very high-level language, changes are faster to locate and easier to make.

- Software prototyping results in:

    * Users who are much more satisfied and involved in the development process.
    * Systems that meet the user's requirements and are much more effective and useful.
    * Improved productivity for all those involved in software prototyping: the user/designers and the systems/builders.

Biography

Orland Larson is currently Information Resource Management Specialist for Hewlett-Packard. As the data base and application development specialist for the Information Systems Marketing Center he develops and presents seminars worldwide on data base management, application prototyping and productivity tools for information resource management. He is a regular speaker at Hewlett-Packard's Productivity Shows and also participates in various National Data Base and 4th Generation Language Symposiums. His experience includes the development of a methodology for designing data bases and the application of software tools to measure data base performance. Previously he was the Product Manager for IMAGE/3000, Hewlett-Packard's award winning data base management system.

Before joining HP he worked as a Senior Analyst in the MIS Department of a large California-based insurance company and prior to that as a Programmer/Analyst for various software companies. Mr. Larson has been with Hewlett-Packard since 1972.

Bibliography

Boar, Bernard H., Application Prototyping: A Requirements Definition For The 80's, John Wiley & Sons, New York, New York, 1984.

Canning, Richard G., "Developing Systems By Prototyping," EDP Analyzer (19:9) Canning Publications, Inc., September 1981.

Jenkins, A. Milton, "Prototyping: A Methodology For The Design and Development of Application Systems," Division of Research, School of Business, Indiana University Discussion Paper #227, April 1983, (41 pages).

Jenkins, A. Milton and Lauer, W. Thomas, "An Annotated Bibliography on Prototyping," Division of Research, School of Business, Indiana University Discussion Paper #228, April 1983, (25 pages).

Larson, Orland J.,"Software Prototyping - Today's Approach to Application Systems Design and Development," Proceedings 1984 International Meeting HP 3000 IUG, Anaheim, California, February 26 - March 2.

Martin, James, Application Development Without Programmers, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.

Naumann, Justus D. and Jenkins, A. Milton, "Prototyping: The New Paradigm for Systems Development," MIS Quarterly, Vol. 6, No. 3, September 1982.

Naumann, Justus D., and Galletta, Dennis F., "Annotated Bibliography of Prototyping for Information Systems Development," Management Information Systems Research Center Working Paper (MISRC-WP-82-12), September 1982.

Note: The above working paper as well as the paper by Naumann and Jenkins entitled "Prototyping: The New Paradigm for Systems Development," MIS Research Center-Working Paper (MISRC-WP-82-03), October 1981, are available for $3.00 each from :

University of Minnesota Systems Research Center
School of Management
269 19th Avenue South
University of Minnesota
Minneapolis, Minnesota 55455

or by calling 612-373-7822.

Podolsky, Joseph L., "Horace Builds a Cycle," Datamation, November 1977, pp.162-186.

Wetherbe, James C., "Systems Development: Heuristic or Prototyping," Computerworld, Vol. 16, No. 7, April 26, 1982.

GUIDELINES IN DESIGNING USER-ORIENTED SOFTWARE

Bill Marsh
Del Monte Corp. Research Center
Walnut Creek, California, USA

SUMMARY

This paper will discuss how to create software with the user in mind. Most users are computer illiterate and require software systems that are simple to use, supply extensive error checking, and offer an on-line help facility.

The topics discussed in the paper are:
a)Project breakdown.- divide the system into workable sections.
b)Menu design.- design menus to optimize user interaction with software.
c)Computer languages.- recommend fourth generation languages (i.e. PASCAL, TRANSACT, etc) for ease of documentation and interpretation.
d)Error checking.- leave no room for user generated errors by using extensive error checking in VPLS and programs.
e)Help keys.- include help keys at every level of entry so user can query when in doubt.
f)Documentation.- provide user documentation replicating the software structure, use lay terminology, and provide the documentation with an emphasis on over-explanation.

INTRODUCTION

Computer end-users are people who know very little about computers and sometime are even hostile towards computers and computer application programs. To make life easier for these people, the computer industry must strive to make software packages easy to use and understand, but at the same time conserve the software's purpose. Designing <u>user-oriented</u> software will help convert computer hostiles and will make life easier for the computer professional who has to support these applications and these people.

An immediate question comes to mind. Why bother to design <u>user-oriented</u> software? The answers to this question are multiple and I will expound on the obvious answers only. In a pragmatic approach, end-users which are confused or intimidated by your software, simply will not purchase it. Or if you are designing in-house software, you will receive an inordiante amount of compliants and ultimately will have to rewrite the software. Another outstanding reason is competition. If your competitors design <u>user-oriented</u> software and whose customers are happy with this software and you don't design <u>user-oriented</u> software, then you will probably be without a customer base. And those customers you should have, will sooner or later learn about your competitors easy-to-use software and will likely switch to your competitor's systems.

Finally, we are left with the notion that we should design <u>user-oriented</u> software only. Alas, I do not preach an ultimate devotion to the user, but only where the need arises. We do not have a need to design <u>user-oriented</u> software for automated processes or self-contained processes, since these program do not interact with humans. Nevertheless, any application that involves human interaction, should be intelligently designed with the user in mind.

This paper will discuss how to create software with the user in mind. Software that the end-user can easily comprehend. Software that is intelligent enough to guide the end-user by the hand when confidence fades. The techniques discussed are very simple to use and paradoxically will accomplish a very complicated and thorough solution to creating user-oriented software.

The topics covered in this presentation include project breakdown, language selection, menu design, error checking, help keys and documentation.

PROJECT BREAKDOWN

This section applies more towards the system analysis part than the actual design. Nevertheless, I feel this section has enough bearing on the final product that I shall make a brief mention.

When you are designing the system flow, I recommend you break the process down into minimal modules. Modules small enough that can be easily programmed, yet complete enough to comprehend some logical function in the system. I will take an example from the last system I designed (all following examples will be from this same system).

I was confronted with the task to computerize a beverage formulation system. This system was extensive and would require large amounts of code. Hence, I divided the system into the different types of beverages to be formulated. By subdividing the system into these 6 modules, I now had an easier system to understand. But after further analysis, I divided the system into yet more modules, that is, each of the previous modules were reduced to add, modify and delete modules. Now I had the system down to minimal, yet logical modules.

The final design of the system was focused around this initial breakdown: six types of formulas and three types of functions within each formula type (add, modify, delete). The moral of this section is to make your design simple and modular from the beginning, as this definition will be a basis for your final design.

LANGUAGE SELECTION

In designing user-oriented software, a modern, state-of-the-art language is recommended. The newer languages (fourth and now fifth generation) are easier to use, document and maintain. In using one of these languages, not only will you design software for the user, but at the same time you will design programs for the programmers. Though the purpose of user-oriented software is for the users' ease, you can also alleviate a programmer's load by using a structured language.

I have used the TRANSACT language extensively. This language gave me the ability to program in a modular/structured fashion (the source code is nicely indented within each recursion and will be easy for another programmer to read), gave me the use of a data dictionary (variables are defined once in the dictionary and not repeatedly within each program), augmented my programming productivity five hold over Cobol (no need to make IMAGE or VPLUS intrinsic calls since efficient macros are used), and provided me with a system execution time comparable with Cobol. Thus, with this language I can create user-oriented software and at the same time create code that other programmers can easily maintain.

Other procedural/structured languages are Pascal, Cobol and SPL, all supported on the HP3000. Nevertheless, these languages do not offer the same high level commands, some cannot use the data dictionary, and none will augment your program productivity the same way as TRANSACT. Granted that a program written in SPL will execute significantly faster than the TRANSACT processor, but then, a TRANSACT program can be made to run as fast as an SPL program. Through third party software, you can translate your finalized TRANSACT source code, into SPL source. Then the SPL source can be compiled into an executable SPL program, which will run must faster than the TRANSACT processor, saving you the time and additional expense of writing the original program in SPL.

The example mentioned above, will serve well when you want a large system programmed in the least amount of time, but with the best techniques and discipline. But if your mission is to create a small system, then maybe the best decision if another language, though I recommend consistency in the languages you use. The moral of this section, is to study the languages available for your use on the HP3000 and make a decision as to which language will give you the best result.

MENU DESIGN

Menu design is the heart of creating user-oriented software.

The object of user-oriented software, is to make the software application easy for the end-user to understand and use. A technique to accomplish this easy interaction, is to use menus. Menus that provide an expanse detail of selection, but ask for a minimal amount of input from the user. Thus we optimize the user interaction with the software.

Menus will be designed to follow the project breakdown. If you have twelve modules defined, then you will have twelve menus, and so forth. The first menu will be a main menu that branches to other subsequent menus. Similarly, your first module (or main program) will branch (or call) subsequent modules. Thus you can envision a menu per module.

This methodology is straight forward and simple to follow. Though it may seem over-simplistic, the object is to make easy software for the user, and at the same time easy software for the programmer. Once you have your mind thinking in terms of modules and menus, then structuring your project into these modes becomes quit simple and eventually becomes second nature.

An example of this technique is the formulation system previously mentioned. Here I have already defined a given number of modules, that is, 6 formula types and 3 events per formula (add, modify, delete) for a total of 18 modules, plus one for the main program/menu. The main menu will contain the choice of either adding, modifying or deleting one of the six formula types (see figure 1). The only input requested of the user is the number corresponding to the choice of activity. From this point, the next menu will be the screen pertaining to the activity (see figure 2).

The second screen is actually the lowest level screen in this example. Thus this screen will handle the actual data entry portion of the system, where data for the creation of a formula is entered. Again, the screen is designed to optimize the users interaction with the computer, to follow some logical sequence of events. In the the second screen (see figure 2), the fields are sequenced in the same order to which the scientist is accustomed when creating rtd-base formulas. Hence, the scientist will see a familiar format when attempting data entry.

```
=========================================================================
                     FORMULA DEVELOPMENT SYSTEM
-------------------------------------------------------------------------
                          ****Main Menu***
-------------------------------------------------------------------------

         1.- Add Rtd-Base                       9.- Add Folded
         2.- Modify Rtd-Base                   10.- Modify Folded
         3.- Delete Rtd-Base                   11.- Delete Folded

         4.- Add Base                          12.- Add Ratio
         5.- Modify Base                       13.- Modify Ratio
         6.- Delete Base                       14.- Delete Ratio

         7.- Add Rtd                           15.- Add Specialty
         8.- Modify Rtd                        16.- Modify Specialty
         9.- Delete Rtd                        17.- Delete Specialty

-------------------------------------------------------------------------
                         [  ] Enter option
=========================================================================
```

Figure 1. Main Menu.

```
=========================================================================
                     FORMULA DEVELOPMENT SYSTEM
-------------------------------------------------------------------------
                        Add Rtd-Base Formula
-------------------------------------------------------------------------

Formula No [         ]      Instructions [                    ]
                                         [                    ]

Ingredient 1  [         ]               Base 1   [            ]
Ingredient 2  [         ]               Base 2   [            ]
Ingredient 3  [         ]               Base 3   [            ]
other         [         ]

-------------------------------------------------------------------------
Please enter new Rtd-Base ingredient data, then press ENTER
=========================================================================
```

Figure 2. Second Level Screen.

ERROR CHECKING

What lies behind the menu design, is what makes software user-oriented. The programs that drive these menus are extensive to the point of error checking for the most minute detail. The objective is to rule out all possible human error in the data entry. This is accomplished by performing entry level checks first through VPLUS (if used) then through the program.

Through VPLUS, you can check for numeric ranges, signs, string matches, minimum length entry, check digit calculations, and additionally can pad fields with blanks, zeros, upshift characters and so forth (consult VPLUS reference manual for more details). Typographical errors are easily caught with VPLUS and can be corrected by the program or redisplayed to the user for correction. Messages explaning the detected error can be used in VPLUS so the user can understand why the error occurred and how to correct the mistake. VPLUS should be used to a maximum for error checking so you won't have to write code for performing the same error checking (see figure 3).

```
=================================================================
FORMSPEC B.04.04 Field Menu                 FORM: CREATE_RTD_BASE
-----------------------------------------------------------------
                      Add RTD-BASE Formula
Formula No  [formula_no.]          Instructions:  [instruct_1..]
             ^
-----------------------------------------------------------------
Num[1] Len[10] Name[FORMULA_NO    ] Enh[IH] FType[R] DType[CHAR]
Initial Value [                                                 ]
-----------------------------------------------------------------
                  *** Processing Specifications **
UPSHIFT;
JUSTIFY RIGHT;
MATCH addaadddd "Formula must be letter-2numbers-2letters-"
                  "5numbers";
=================================================================
```

Figure 3. Menu Error Checking.

The next level of error checking is performed through the program. Error checking such as validating the existance of an item in a data base, performing some complex calculation that VPLUS cannot handle, or interaction with some other system for data validation. You must determine all possible combinations and permutations of errors the user can encounter and have some routine for checking the occurance of the error.

Error checking plays an important role in creating user-oriented software. Users will always make mistakes, regardless of the users' dexterity with computer systems. By performing extensive data error checking at the data entry level, you will avoid erroneous data permantly on file and thus avoid the cost of having to revamp your files in the future.

HELP KEYS

Every menu must have a help key. Users will not always carry the system documentation with them, nor will they always remember just exactly what sequence of buttons to push. Thus, help keys imbedded in menus can always take the user by the hand through the application.

Typically, I have designated funtion key 7 (f7) as the help help key in all my applications. This can vary to your preference. Even at the main menu, the help key is present.

The information displayed when the help key is pressed will, of course, depend on the complexity of the system and the determined screen. Nevertheless, the help explanation must be concise and easy to understand. Do not use computer jargon, but do use jargon pertaining to the application. Make sure you clearly delineate the choices available to the user, plus the formats of the data entry. Users will feel immediately comfortable with application software if they have help keys to get them out of tight spots.

The information displayed when a help key is depressed, can be stored in an MPE, KSAM or IMAGE file. No need to imbed the text within your program. The advantage of having the help text in files is for ease of maintenance. If your program changes for updates (which is always true), then modifying the help text is as simple as using a text editor. Some people may suggest using VPLUS screens for help text, but this becomes cumbersome and is slow on the data display. Switching between block mode and character mode in a program is a simple task, and the function keys can still be used (again, this is more easily accomplished with TRANSACT).

DOCUMENTATION

Finally, a system is not complete without its documentation. But most people don't realize the importance of the documentation, since the system cannot be adequately taught without documentation.

The documentation should follow the project breakdown modules, since this is the same format used in the program and menu design. The documentation will thus follow the same logical sequence of the application, and will make reference look ups more easier for the end user.

The language used in the application documentation must be concise and understandable. Don't use computer jargon (e.g. CPU, CRT, character, integers, etc.), but instead use everyday words (e.g. computer, screen, letters, numbers, etc.). The end user must feel comfortable with the documentation and never should the user feel intimidated with the documentation. If a user becomes aprehensive with the documentation, then most likely the user will not like the software and will do nothing but complain. Good documentation is easy to write and is even easier to maintain. Besides, what is in your documentation, will only reflect what is in your program.

What I have included in my documentation, namely for large systems is an index. The index is extremely beneficial for performing random searches of information. The way to set up the index is simply to read through the documentation and highlight the outstanding subjects, making annotations of the page. When you finish reading the documentation, you will have a substantial list for your index.

Finally, place the documentation in a nice binder, where optionally you can include a cover. I've taken advantage of the graphics application software and created impressive covers and title pages for the binders. The adage 'you can't judge a book by its cover' may be true, but 'first impressions are always the best.'

BIOGRAPHY

Bill Marsh
is the Systems Manager for the HP1000 and HP3000 computers at the Del Monte Corportion Research Center, located in Walnut Creek, California. With Del Monte for 3 years, Bill's experience has been with creating software systems for the Research Center's scientist and administrators. Developments include a beverage formulation, nutritional analysis, time reporting and other miscelleaneous systems. Previous to Del Monte, Bill has developed <u>user-oriented</u> software for ARCO, International Data Services and other foreign and domestic companies.

MANY MEN -- ONE MACHINE - THE DESIGN OF INTERNATIONAL APPLICATIONS PACKAGES BY ADAPTING THE USE OF THE MACHINE TO THE EUROPEAN MARKET

Julie T. Mowinski
Euroco Computer Services Ltd., UK.

Summary

The problems inherent in the fast growth and commercialisation of computer applications program design are becoming apparent to users throughout the european market place.

Tighter economic restraints mean lower budgets for internal computerized solutions to company administrative procedures. More and more easily recognised opportunities by commercially minded developers to write package solutions and sell in bulk mean low cost off-the-shelf solutions proliferate in the market place. A perfect fit? Then why has so much frustration, chaos and confusion arisen from this simple equation of demand and supply. Perhaps insufficient thought has been applied to real user requirements, ie. the real interface question. When is a package not a package? When it is no solution. It is not just a question of what do you want; but where, how, by whom, when, how many?

Remembering that we are living, doing business, in Europe where many different languages are spoken, where the total business environment has to transcend national borders, different laws, business practices, tax legislation, notation, governmental attitudes and policies - the company structure dictates that all these idiosyncratic features in various company locations be brought together to be planned, analysed and revised at one single level. In consequence, there is more than one single factor which it is vital to consider when designing an applications packages, irrespective of whether it is aimed at the international, european or national market.

A new concept in package design must be applied - not by shuffling current programs around to make them fit; but by using a fundamental structure at planning and design level which will enable the program to satisfy the requirements of users, europe-wide, which will make the package not only useable, user-friendly; but will also make it useful.

This paper will outline and discuss the strategy necessary in the design of a truly international and multinational applications package.

Introduction

The fundamental question which arises from the concept of trans border design of applications software is how to retain sufficient flexibility within the technical design, in order to enable a large number of differing criteria to be applied. Alternatives within these criteria are obviously going to vary from country to country and from user to user and therefore the mechanisms and procedures which the system provides for user definition of these alternatives must be sufficiently simple to allow them to be easily operated by the average user.

The structuring of applications software design must therefore take this environment into account. There becomes apparent an obvious split between two main areas of concern, namely that of definition of the system design itself. Yet these are not the sole areas of importance; there is another which has been given too little emphasis for too long - the user interface. The grey area which allows the user access to the processing power of the machines in order to satisfy his requirements. This interface determines whether his application specifications have been correctly interpreted and it also displays the product designer's ability or inability to successfully mask technical operations to reduce as closely to zero as possible, requirements for computer literacy on the part of the user. Colloquially, software packages which offer an easily understandable interface to the user are termed user-friendly. However, this is a much over-used term and also does not imply that the software is actually useful, merely easy-to-use, or even in some cases just good fun!

The general move towards the purchase of software packages, influenced by the cost of one-off in-house software development and by the wide availability of off-the-shelf solutions has meant that the driving force behind the design of applications software has moved across from being user requirement driven to being ostensibly programmer driven, where savings in time and cost are taken advantage of, where investment imposes limits on development resources, where project leaders are technical whizzkids wishing to impose thier latest developments on the products, or to mould the finished article around a particular technique.

This inherent lack of consideration of real user necessities which is apparent in so many applications products is born of the commercialisation of software production. The user interface is always the last part of any program design to be completed - user documentation is often produced even later and is rarely of a high standard. This commercialisation does not allow for application of scientific planning methods to the development of package programs. In fact planning a new applications package needs just as much thought and effort in the region of market research and product marketing as a new brand of chocolate biscuit or some new form of fish fingers. I guess taking a bite out of a 9mm tape may not be quite as tasty as some of the other alterntives; and may not provide a clear idea as to the preferences of the market place; but it is a fact that there are many marketing techniques which can and should be used when designing a new system. The general schema for doing this is as follows:

Identifying the product

To generate ideas for possible new products or services, areas of unfilled customer needs have to be identified ie. gaps in the market place. Another way to generate new products is to select promising areas for further development, where additional requirements have been identified and where the structure of the programs allows for the additional features to be added. However, it is important when doing this not to stretch the package too far beyond its original limits, unless a major structural re-write will be undertaken since this obviously leads to inefficiency in the execution of the program and provides no real additional benefit for the user. Also, if a product has been available for some time and has gained a certain reputation through wide usage or effective marketing, then it is questionable whether a major modification would profit from the full effect, from a market communications point of view. Previous knowledge of the product by consumers can lead to preconceived ideas about the product, however much it has actually changed.

When deciding upon a new area in which to develop applications packages, it is important to identify big, growing, profitable, not-too-competitive, and relatively easy to enter markets. Therefore information needs to be collected with respect to market size, structure, characteristics, trends, profitability, competitors and basic market facts.

Once new products have been generated, then it is appropriate to identify market segments since to concentrate on too wide a spectrum can lead to inefficient use of resources use of resources and wasting of investment. This identification can be effected by monitoring the reactions of different customer groups to product appeals and also by gauging consumer responses and characteristics.

Who is the product aimed at?

There are research surveys designed to discriminate between consumers and identify different market patterns, or segments. Segmentation criteria should predict or explain market differences and be exploitable in practice. The traditional segmentation criteria are geographic, demographic and product usage. In each case, collecting data using the criteria listed will increase understanding of relevant characteristics of buyers. It is very important at this stage to at least have some idea of the criteria which are to be used in the research. For instance, which geographic areas will be covered, are you going to emphasize research of relative purchasing power of companies, or are the questions going to be biased towards identifying a high volume sector of the marketplace etc.

A more complex, but very revealing type of segmentation analysis is termed "psychographics". This term distinguishes the approach from "demographics" which use general "people" characteristics. "Psychographics" are concerned with people's subjective attitudes to, and feeling about, specific products or services. It produces multi-dimensional groupings which are often highly explanatory of actual market behaviour.

To carry out this type of study, some background knowledge of the market is needed, including purchasing profiles and ways in which competitive products are used. From this a topic guide can be developed for use in qualitative research. In the case of international applications software this will probably take the form of a feature list.

This qualitative research then involves depth interviews which are used to develop ideas about different market groupings, indicating the language prospects use the market and products available in it, and suggesting attitude statements for developing measuring instruments, to be used in quantitative methods, thus defining the size of different segments.

Once the exact market segment has been defined, then the product must be designed, with direct input from prospects relating to product attributes, product quality, performance and comparisons.

Once the product has been completed then a test marketing plan needs to be put into operation, different marketing strategies tested and market sales predicted. In order to do this successfully, we need to measure customer reactions to elements of the marketing mix, ie. product, price, promotion, and ultimately monitor the rate of purchase of the product or penetration level, pattern of re-purchase, pattern of retail sales and consumer reactions to different marketing strategies.

The choice of producing an applications package which addresses the international market place becomes relevant when one is trying to identify the specific market segment that a new product will be aimed at and therefore any research which is undertaken after this decision has been taken must of needs involve aspects of the international market place which are relevant to the ultimate design of the product. These will include obtaining a statistically significant sample from all the countries within the target area, and formulating questions around topics of international relevance. This means that research will take place on a much wider basis than it would had it been restricted to a national market only.

Matching your product to your market

There are some typical marketing research techniques which can be applied to this particular situation.

The typical usage and attitude study

This is a quantitative survey which typically involves over one thousand respondants and would address such questions as current usage of similar products or alternatives to fulfil the application in question, attitudes to certain types of solution and perhaps to some competitive products within the group. To qualify market segments large-scale questionnaire surveys are necessary. The long list of attitude statements which we obtained from our qualitative research, ie. feature list, must be reduced to those which are most discriminatory. A quantitative survey with factor analysis of the results will reduce the specific attitude statements to more general factors relevant in the market place. Factor analysis is simply a technique which serves to combine questions, thereby creating new variables. It also analyses the interdependence between questions, variables or objects. Alternatively the list may be reduced through piloting and subjective analysis.

Identifying customer groupings

The developed questionnaire containing the final short list of relevant attitude statements is administered to a statistically significant representative sample. The statistical technique of cluster analysis is then applied to the results, and this will identify sub-groups of consumers with shared attitudes in the market place, for instance this might be done on the basis of the product benefits they seek. Psychographic analysis indicates target groups in the market, and may suggest new products which would appeal to them. Alternatively, it cam suggest which appeals would be most attractive to different groups in the market.

Test centres

One of the most useful ways of researching a specific applications market place is to establish test centres to include a representative sample of prospects (obviously in this case they would come from a number of different countries). These companies would then be exposed to the product, discuss its relevance to their individual requirments, undergo a typical training schedule etc. This has to be undertaken sufficiently early to allow input to be used in the released version of the product, and has specific relevance to user interface requirments.

Measure your resources

Although I do not intend to launch into detail regarding the organisational structures and technical facilities required in this exercise, since that is not the object of this paper, I feel it relevant to mention the fact that some thought must be given to this area and care must be taken to match the development and marketing program with relevant resources. Thus you should ask yourself questions like: do we have the resources to produce an international package, ie. do we have personnel who are native speakers of the target countries, do we have sufficient access to specialists in the proposed package's environment, ie. accountants for accounting packages, and who are well versed in international accounting methods if the package is to be marketed abroad and so on.

Do not underestimate the complexity of the requirements specification

The easiest applications software to design for the international market are general applications such as database management systems and word processing, where some adaptation is required for the international market place; but not nearly so much as is required for specific applications packages such as accounting packages. In designing an international package, the least complex marketing situation is where the product would be used instead of a local solution, since although foreign variations of procedures have to be taken into account, complex consolidation procedures may not be necessary. However, a more complex marketing situation may present itself. Let's look a little more closely at such a situation.

Sentimental Slush Co. Ltd. is a multi-national company, with Group Headquarters in Colorado Springs and European Headquarters in London. They have seven wholly-owned subsidiary companies in France, Germany, Italy, Spain, Belgium, Netherlands and Eire. Sentimental Slush Co. Ltd. manufacture Cabbage Patch dolls and have a requirement for a group accounting solution, including standardisation of reporting procedures to allow european consolidation to the UK and group consolidation to the USA, whilst at the same time adapting itself to the peculiarities of each national system. This is a typical, if relatively complex, example of the type of organisation which international software would be aimed at.

Therefore the development approach must take into account the following specific factors which directly relate to individual national requirements, and these can be grouped into two main areas of relevance. Firstly, there are factors which are demographic, or customary in various countries, such as language, value notation etc. These are relatively easy to identify and easy to implement. Secondly, there are legislative aspects and business practices of a certain environment - in this case accounting practices, standards within the various countries. Some standards have legislative backing in some countries and not in others etc. and therefore this area is rather complex, the more so since one company may span a number of different accounting practices as in the case of the Sentimental Slush Co. Ltd.

Broadly speaking, in order to satisfy the above requirement, we would have to consider some of the following elements.

i. Languages - whichever way one implements the use of different languages in the package, one must always consider the one with the longest words, such as German. The system must also provide for easy substitution of different languages.

ii. Date formats have to be flexible to cater for european and american formats, as well as catering for different date notation.

iii. Different value notation has to be catered for (comma, thousands separator, decimal points, number of decimal places etc).

iv. Different taxation rules must be included (eg. multiple VAT/TVA rates - how many?, sales tax versus VAT). Let us take a closer look at how corporate taxation differs from one Sentimental subsidiary to another.

TAXES ON CORPORATE INCOME

In BELGIUM, the basic rate of tax on corporate income is 45%. When the taxable income does not exceed the equivalent of £195 000 and no outside concern has a majority interest in it, then the rate is reduced according to a fixed scale. A surcharge is applicable at a rate of 30% on $\frac{1}{4}$ of income tax for the 1st quarter, 25% on $\frac{1}{4}$ of income tax for 2nd quarter, 20% on $\frac{1}{4}$ of income tax for 3rd quarter and 15% on $\frac{1}{4}$ of income tax for 4th quarter. This can be avoided by making advance quarterly payments.

In FRANCE, the basic rate of tax levied on net profit earned in continental France is 50%.

In WEST GERMANY, rates are imputed as follows: for a German company on profits distributed to stockholders - 36%, and for undistributed profits - 56%. For a branch of a foreign company - 50% (This excludes West Berlin).

In EIRE, 40% is levied on profits up to the equivalent of £21 000, 50% on profits over £30 000, and a sliding scale is operated between these two amounts.

In THE NETHERLANDS, the basic rate applied to the first £10.000 is 45%, for the next £2 500, 60% and for amounts in excess of this, the rate is 43%.

In SPAIN, the standard rate of corporate income tax is 35%, unless the company is assessed through its shareholders on a personal income tax basis.

In ITALY, the overall tax rate on corporate income (taking into account corporate income tax and local income tax) is 43.37%

In the UK, corporation tax is 52% generally, unless tax adjusted profits fall below a certain limit, when it is 38%. Also when a distribution is made, advance corporation tax has to be paid amounting to 3/7 of the distribution. This amount can be offset against the ultimate liability on income of the same accounting period; but for not more than 30% of taxable income. Any profits which accrue from the extraction of oil and gas from the UK and its waters are subject to 75% rate of tax.

(The above figures for the UK apply to period pre budget '84)

In the UNITED STATES, a sliding scale is operated, where profits up to £21 000 are liable to imposition of regular Federal income tax at a rate of 15%, profits up to £42 000 at 18%, up to £63 000 at 30%, up to £84 000 at 40% and over this figure at 46%. Generally a corporation is required to prepay 90% of its estimated tax liability for a tax year in instalments during that year. In addition corporations may be subjected to Federal minimum tax, accumulated earnings tax, personal holding company tax, state and municipal taxes etc.

This illustrates how rates of taxation not only vary from country to country; but that taxation criteria vary also, ie. they are not limited to the value of profits; but also in certain cases on whether the profits are distributed or not and also on whether the company is native or foreign, etc.

GROUP TAXATION

The qualifying criteria for group taxation vary from country to country and in some cases, eg. Belgium, it is not permitted at all.

SALES OR TURNOVER TAX

This also veries from country to country, mainly in terms of rate; but also in terms of exemptions, surcharges etc. For instance, Belgium has a standard rate of VAT of 19%, a reduced rate for basics of 6%, an increased rate for luxury items of 25%, as well as a special rate of 17% for buildings, utilities, restaurants etc., and an even higher rate of 33% for certain luxuries such as perfumes, watches etc.

France has a basic rate of 18.6% with some variations for different types of goods and services. Germany's basic rate is 13% and also 6.5% for certain items. Eire has rates ranging from 5% to 35%, the Netherlands 19% and 5% for prime necessities. Spain currently does not operate a system of value added tax, but a cascade tax. The UK has a basic rate of 15%, Italy a basic rate of 18% and other rates up to 38% for luxury goods.

The United States operate a different system again where most states and some cities impose sales tax.

v. Different countries have different ways of handling trade discounts.

vi. Depreciation rules may vary from country to country. For instance in France, Belgium, the Netherlands, United Kingdom and the United States, it is required or insisted upon that depreciation charges reflect the estimated salvage value at the end of estimated useful life; but throughout the rest of the Sentimental Slush Group it is not practised. In France, Germany, Italy, and Spain, depreciation methods and rates are predominantly governed by tax requirements, whereas throughout the rest of the group this is not the case.

vii. The system has to be adaptable to different (flexible) chart of accounts formats. For instance in France the chart of accounts format is defined by the Plan Comptable (legislation), and in the United Kingdom by the Companies Act, 1981.

The first mistake often made by the Corporate or European Headquarters is to assume that the standard system will address all the problems of the international subsidiaries and the evaluation of such systems is often quite advanced before the local problems become apparent and the individual subsidiaries start weighing up the advantages and disadvantages of the common group solution versus the local solution. When is a solution not a solution - when it is no solution at all.

It is not just a question of what do you want; but where, how, by whom, when, how many? We have seen a few of the differences which need to be accounted for when designing this kind of international solution. Now let's lookat two more aspects which are of direct relevance to the requirements of Sentimental Slush Co, Ltd. - consolidation and foreign currency transactions.

CONSOLIDATION

In many countries, the preparation of consolidated financial statements by parent companies is required or has come to be regarded as almost the invariable practice. In other countries, consolidations are virtually unknown; this may be because of legislative restrictions as to intercompany shareholdings or as to their preparation, because parent-subsidiary relationships rarely exist, or because their usefulness is not yet appreciated by the business community.

Thus, there are countries in which consolidations are seldom or never found and to some extent a geographic pattern can be identified. Consolidated financial statements are never or only rarely found in South and Central America, except in Mexico, Panama and Venezuela where they are either the required or predominant practice. Verious continental european countries do not follow the practice, although a few of the largest companies in these countries may produce some form of consolidated or combined accounts. In Belgium, for example, consolidation practices are still in the development stage but legislation has been enacted requiring consolidation by quoted holding companies for the first time in 1978. In Germany, it is the practice to consolidate only domestic subsidiaries.

Since all the Sentimental subsidiaries are wholly owned by the american parent company, and group structure indicates that consolidation is carried out on a country basis, through the UK and then to the States, they are not so concerned with these national idiosyncracies. However, an accounting package which truly addresses the international market has to be flexible enough to cater for all group accounting structures and local requirements.

FOREIGN CURRENCY TRANSLATION

Foreign currency translation is one of the more controversial subjects on the accounting scene today, evoking perhaps an undue amount of emotion yet seeming intractable and insoluble. In some of its aspects the problems are not acute and there is a consensus which amounts to a complete international harmonisation of views. In other aspects there is a severe dichotomy which stems from such fundamentally different underlying philosophies that international harmonisation seems unattainable, at least in the short term.

It is interesting to observe that back in 1975, little had been issued by way of accounting standards on the subject, other than a requirement in some countries to disclose the basis of translation. Since then, accounting standards have been set in a few countries prescribing a single method of translation. Some of these have been met with opposition, both from within the accounting profession and from the business community.

While some national accounting bodies were endeavouring to prescribe a single method of translation, the International Accounting Standards Committee proposed a standard which would allow prescribed alternatives. The controversy over this subject is such that this proposal not to prescribe a single translation method may yet offer a compromise solution to what many regard as an impasse.

The proposal recognised and dealt with the two principal aspects of foreign currency translation - accounting for transaction in a foreign currency and translating the financial statements of a foreign enterprise. It is the first of these principal aspects in which a greater degree of harmonisation is found in current practice.

In conclusion

These are then the two main areas of concern when looking to design international packages - demographic factors and local business practices/legislative requirements. To stress an earlier point, however, we must not be blinded by the complexity of the above requirements to the point of forgetting our underlying objective of satisfying user requirements so as to make the package useful. It is here, then, having looked at what we, the designer/developer of a package consider to be features necessary to the success of the package, that we must look again to the prospective user and find out, through the methods outlined above, in the section on qualitative research, how one can best present a system to the user and which are the factors which are going to influence his choice of system which we hadn't perviously considered.

It is only in this way that a package will be truly international and multinational in its appliation. Here we have given the example of an international accounting solution requirement. The procedures apply to all applications packages, whichever business transaction they are catering for,

We should not aim to please all the people all the time, only some of the people most of the time.

Biography

Julie Mowinski
is Marketing Executive at The London offices of Euroco Computer Services Ltd., with responsibility for planning and implementing product and promotional marketing of Euroco's financial, stock control and order processing software. Julie has 6 years commercial experience in Europe, of which the last 3 years have also been in the computing industry, specialising in data analysis consultancy, marketing of micro applications packages and marketing of applications packages for the HP3000 range.

Julie has a Bachelor of Science Honours Degree in European Studies and a postgraduate Diploma in Management Studies.

VPLUS: IMPROVING THE END USER INTERFACE

Nancy Ofslager
Hewlett-Packard, Cupertino, USA

Summary

Virtually every HP3000 customer utilizes VPLUS/3000, Hewlett-Packard's data entry and forms management system as the end user interface for data collection. The engineering and support teams for VPLUS recognize the immense software investments on your part and strive to continually support and improve the VPLUS subsystem.

This year, several major enhancements to VPLUS/3000 will be released which will make user friendly interfaces very easily attainable. These enhancements will include support of several special terminal features such as color and touch, as well as function key labeling for HP264X terminals, a cursor positioning intrinsic and the much awaited Native Language support. In addition to the enhancements which will aid in improving the user interface, many enhancements have been coded which will aid the interface designer in his job while using VPLUS.

This paper will describe the functionality of the new VPLUS enhancements and the manner in which they can be most efficiently utilized. HP feels that these enhancements to VPLUS/3000 will greatly aid in the continuing quest for user friendly, productive interfaces.

Native Language Support

The localization enhancements of VPLUS enable an application designer to provide interactive applications to end users which support both the user's native language and the local custom for numeric and date information.

VPLUS enables end user applications to process native language data for languages implemented via any of the HP-supported 8-bit extended character sets.

This means VPLUS/3000 will

* be sensitive to native decimal and thousands indicators
* use the correct native month names to reformat dates
* upshift alphabetics based upon native characters
* use native characters in single value comparisons and table checks
* use native characters in range checks

VPLUS will not support the interface design process in native languages. This means form names, field identifiers, and field tags will not support native characters. REFSPEC and REFORMAT will not be localized.

The language characteristic of an application is reflected in its associated forms file. It is defined in a field labeled "Form File Language" on the Terminal/Language Selection Menu in FORMSPEC. An application and its forms file may be unlocalized, language-dependent, or international.

The default language characteristic for a forms file is unlocalized. If no action is taken by the forms designer to assign a language-id in a forms file, it remains NATIVE-3000. All forms files created before VPLUS included Native Language Support are automatically unlocalized. An unlocalized forms file will have a zero in the Form File Language field on the Terminal/Language Selection Menu to represent NATIVE-3000.

If an application is to operate only in a single language, then the associated forms file would be defined to be language dependent. To define a language-dependent forms file, the form designer would enter a language-id into the Form File Language field on FORMSPEC's Terminal/Language Selection Menu. This assigned id code would be a non-negative integer indicating the desired language, as documented in the HP3000 Native Language Support Reference Manual and Conversion Guide (part number 32414-90001). If the language-id value indicates an acceptable language, which is not configured on the system, a warning message will be generated, but FORMSPEC will not reject the id. Language-dependent forms files would be used if the text on the forms must be in a native language. If this were the case, unique versions of a forms file would be created for each desired language.

In some cases, menus, commands, and titles are desired to be in the business language of a multinational corporation, yet the actual data captured or displayed is dependent on the language of the end user. A special value, -1, is selected in FORMSPEC, on the Terminal/Language Selection Menu for the forms file language-id. At runtime, the application would call the new VPLUS intrinsic, VSETLANG, to assign the desired language. VSETLANG will not affect unlocalized or language dependent forms files and will return an error code. If an application is run on an international forms file without calling VSETLANG, it will be executed with the default language, NATIVE-3000.

The components of a form which can be language dependent are the text, the initial values of fields, and some field edit rules. The language-id is used as the context for all data editing, conversion, and formatting when the form is used, at runtime.

Any edits for date and numeric fields must continue to be specified in NATIVE-3000 terms within FORMSPEC. This is indicated by the field labeled "FORMSPEC Language" on the Terminal/Language Selection Menu. FORMSPEC will initialize this field to 0 or NATIVE-3000. Currently, it can not be changed. VPLUS will convert any specified initial values to the corresponding native values when the form is executed. Single value comparisons (GE, LT, GE, LE, EQ, NE), table checks and range checks (IN, NIN) specified within FORMSPEC may contain any character in the 8-bit extended character set corresponding to the language-id.

VPLUS supports several date formats and three possible date orders: MDY, DMY, YMD. Any format is acceptable as input when the form is executed (providing the field is long enough to accommodate the format). The order is specified by the forms designer uniquely for each date type field. With Native Language Support, the alphabetic month names are edited and converted to numeric destinations using the months corresponding to the language of the forms file. The format and the date order are not related to the language of the forms file.

The date order for the date constants used in processing specifications can not be localized. It remains MDY. The date order for the system constant $TODAY will continue to be determined by the field type of the destination field.

The symbols indicating decimal position and marking of thousands are defined by the language. When entering data, moving data between fields, and defining numeric constants, the thousands and decimal symbols will be language-dependent. This applies to the NUM[n] and IMP[n] data types in FORMSPEC.

The FORMSPEC formatting statement UPSHIFT will use the Native Language UPSHIFT tables corresponding to the language of the forms file.

Range checks and simple value comparisons (LT, GT, EQ, LE, NE, GE) involve native collating sequences. When the form is accessed at runtime, the collation table corresponding to the language-id is used to check the field edits.

This release of Native Language Support in VPLUS does not include support for pattern matching with native characters. MATCH will continue to use ASCII rules.

The designer of the forms file controls the language option of an ENTRY user. In most cases, the language of the forms file determines the language for data editing and formatting. The ENTRY program uses the intrinsic VGETLANG to determine the language-id of the forms files which the user selected. Unlocalized forms files invoke the NATIVE-3000 environment. Language-dependent forms files invoke the particular language selected by the language-id. If the forms file being used is international, ENTRY must determine and set the language.

In order to determine the language of an international forms file, ENTRY will first call the Native Language Support intrinsic NLSGETLANG to determine if a language has been selected globally for the user. If the language cannot be determined using NLSGETLANG, the user is prompted for a language-id. Once a language has been determined, ENTRY calls the intrinsic VSETLANG to set the language to be used with an international forms file.

The batch file does not have a language indicator. Users with different languages may collect data in the same batch file if the associated forms file is international.

For more information regarding the Native Language Support, consult the HP3000 Native Language Support Reference Manual and Conversion Guide (part number 32414-90001).

Touch Support for the HP150

VPLUS will soon support a low overhead, easy to use touch feature. This feature is aimed to improve programmer productivity for the design of intuitive, friendly user interfaces using the touch feature of the HP150.

With this new enhancement enabled, VREADFIELDS returns a field number when a field on the screen is touched. This field number is the same as the field number which was assigned by FORMSPEC at form design time. In fact, all fields as currently defined on a form will return a field number when touched. No additional definition of touch fields is necessary. The field number returned is negative in order to distinguish from the positive function key numbers. Fields on the screen can be treated just like function keys. The application is free to interpret the field number any way it wishes; collect field numbers, branch to another procedure on a certain field number, ignore the field number, enhance the field, etc.

Thus, touch applications can be designed using basically the same VPLUS calls and the same forms file as you are currently using. The negative field number returned to the LASTKEY field in the COMAREA can then be interpreted for further processing.

Two important contributions can be realized with this new enhancement. First, the interface is more intuitive; selection is direct rather than association of function key on the keyboard to a label on the screen. Second, the interface will allow improved sensitivity; the application gets feedback on a field-by-field basis instead of the usual form sensitivity.

In the chosen implementation, an approach has been taken to minimize the extra escape sequences required for definition of touch fields. This approach provides the most powerful touch capability needed for forms design with VPLUS in a simple and efficient manner. The HP150 is simply put into column/row reporting mode by VSHOWFORM. VREADFIELDS interprets the coordinates and translates them into a field number. If an area of the screen which does not contain a field is touched, VREADFIELDS will return a field number of -999. For each field touched, the overhead involved is a 12-byte coordinate report transmitted by the HP150 and the translation of the coordinates to a field number.

To activate the touch feature, the application must set bit 0 of the SHOWCONTROL word in the VPLUS COMAREA to 1 before calling VSHOWFORM. This feature will remain activated until the SHOWCONTROL bit 0 is set to 0, or until VCLOSETERM is called.

In order to use the touch feature effectively, a method is needed to provide feedback to the end user when a field is touched. Changing the field enhancement is one of the best methods to indicate that a field has been touched. The enhancement of a field can be changed by using the existing intrinsics VSETERROR/VSHOWFORM, or family forms. To aid the application designer in taking advantage of the touch capability, an enhancement to VSETERROR has been made.

VSETERROR has been enhanced to toggle the error flag when called successively; i.e. if VSETERROR is called the second time for a particular field, the error flag will be cleared. Therefore, the error enhancement can be used with VSHOWFORM as feedback to the user to indicate that a field has been touched. This feature is only activated if SHOWCONTROL bit 0 is set to 1.

```
example:  IF COM-LASTKEY < 0 THEN
              CALL "VSHOWFORM" USING COMAREA
              CALL "VREADFIELDS" USING COMAREA
              IF COM-LASTKEY IS NOT EQUAL TO -999 THEN
                 CURRENT-FIELD = -(COM-LASTKEY)
                 CALL "VSETERROR" USING COMAREA, CURRENT-FIELD
                 CALL "VSETERROR" USING COMAREA, LAST-FIELD
                 LAST-FIELD = CURRENT-FIELD.
```

The first VSETERROR toggles the error flag of the new field touched, and the second VSETERROR toggles the error flag of the previous field touched. When VSHOWFORM is called, CURRENT-FIELD will be highlighted with the error enhancement, while LAST-FIELD will be reverted back to the normal enhancement. (The intrinsic VPLACECURSOR described later on will solve the problem of VSETERROR controlling the cursor.)

When designing touch applications, the designer should keep the idea of compatibility in mind. Applications should be able to run on other HP262X terminals, or on a touch terminal/workstation using keyboard input. For VPLUS/3000 applications, this objective can be achieved by designing one single form interface for all HP262X terminals.

To design a touch application with compatibility in mind, the interface should be built so that the user has the option of responding via the keyboard or the touch screen. This consideration is important in two respects. First, it provides an alternate path for the end user when the application is run on touch terminals. Second, the application can be run on non-touch terminals as well as touch terminals.

In the VPLUS user paradigm, two basic models can be postulated to illustrate how touch can be implemented to achieve easy user interfaces. In the first model, a form with many items is presented to the user for selection. The end user is to select multiple items from the screen. The second model offers the application end user many items and only one item is to be selected from the screen. Let's look at one of them in detail.

The first model illustrates how a form could be designed for the selection of one item out of many items displayed on a screen. A selection field is provided for users who would rather enter a selection number and press the ENTER key, or when the application is run on an HP262X terminal without the touch capability (see figure 1).

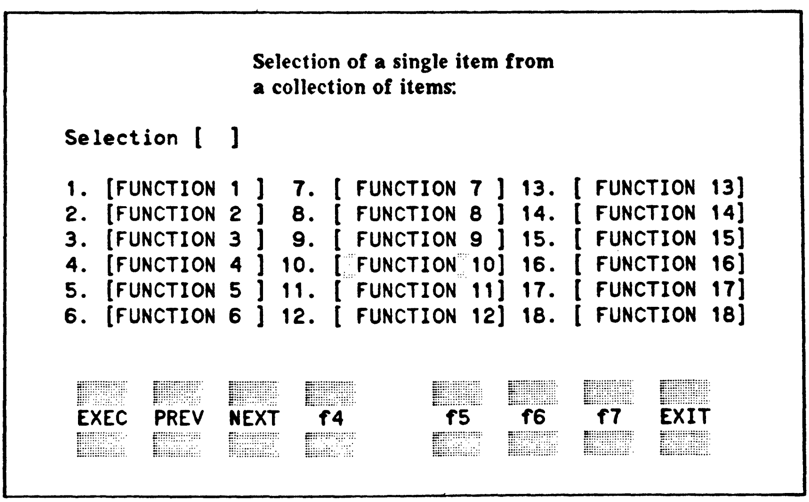


FIGURE 1.

Eighteen display only fields are created in this example. The functions can be selected by touching the corresponding field. When the field is touched, the enhancement will be changed to full bright inverse video. When touched again, the enhancement reverts to half bright inverse video.

To aid the selection of closely packed items on the screen, the application can be implemented so that two function keys can be assigned to move the selected item up and down to adjacent fields. For example, if item 10 has been selected via touch, the enhancement will be changed to full bright inverse video. If function key 2 or the touch sensitive label PREV is touched, item 10 will be reverted to half bright inverse video and item 9 will be enhanced to full bright inverse video. The same method can be applied to function key 3 except item 11 will be selected instead of item 9. Figure 2 demonstrates sample logic that could be implemented to achieve this model's functionality.

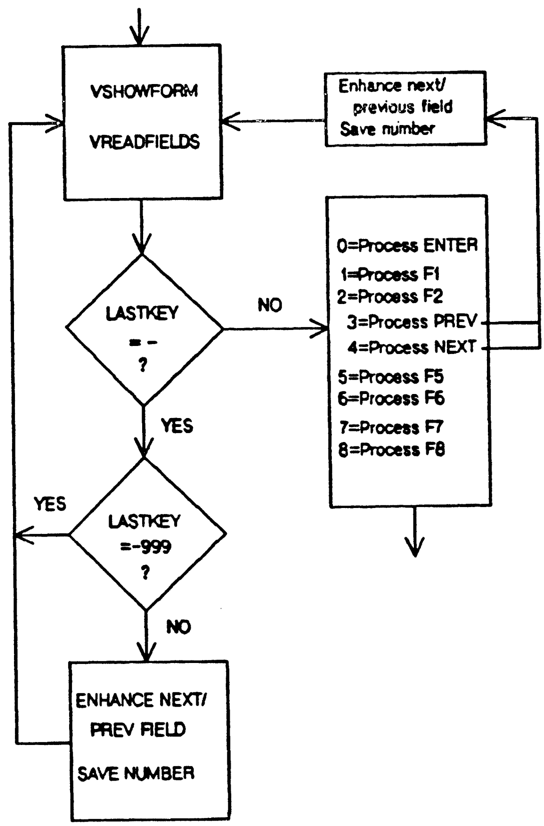


FIGURE 2

Cursor Positioning

One of the most frequently requested enhancements to the VPLUS product is the creation of a cursor positioning intrinsic. Shortly this desire will become a reality.

The new VPLUS intrinsic, VPLACECURSOR, will allow an application to position the cursor to any input field at runtime. Calling VPLACECURSOR with a positive field number indicates the field number; a negative field number indicates the screen order number. The former is much preferred. The reason being that if the fields in a form are rearranged, no modification to the application is necessary. This is consistent with the VPLUS philosophy of uncoupling the user interface from the application program.

This procedure places the cursor to the field specified by the second parameter of the intrinsic call. VSHOWFORM must be called before calling VPLACECURSOR. The procedure returns an error if the field number or the screen order number does not exist. An error will also be returned if the intrinsic is called to place the cursor to a display-only field.

example:

```
CALL "VSHOWFORM" USING COMAREA.
CALL "VPLACECURSOR" USING COMAREA, FIELD-NUM.
CALL "VREADFIELDS" USING COMAREA.
```

Function Key Labels for HP264X Terminals

With the advent of function keys, it has become apparent that these keys can provide a more intuitive means to interact with the computer. The HP262X terminals provide the additional support for function key labels on lines 25 and 26 of the terminal screen. The HP264X terminals do not support the equivalent HP262X function key labels. However, VPLUS will soon provide a means of displaying function key labels on HP264X terminals.

During screen design time using FORMSPEC, the forms designer will decide whether the forms file is to have function key label support by VPLUS on HP264X terminals. This function is selected if a "Y" instead of an "X" is entered in the HP264X field on the Terminal/Language Selection Menu. The selection "Y" signifies that the forms file is to be compiled for HP264X terminals and function key labeling of these terminals will be implemented. The labels must be entered on the Form Function Key Labels Menu or the Global Function Key Labels Menu in FORMSPEC, or set in the application using the VSETKEYLABEL or VSETKEYLABELS intrinsics.

At runtime, if the forms file was created as described above and the LABELOPTION word in the COMAREA is set to a 1 (prior to VOPENFORMF), the labels will be inserted on lines 23 and 24 of the HP264X terminals. If the window line is specified to be on either the 23rd or 24th line, it will automatically be moved to the 22nd line. If the window is specified to be at any other location, it will be displayed as specified. Screens with information on the twolines where the HP264X labels are to be inserted will appear with the function key labels still on lines 23 and 24, but with the two lines of information pushed down. The hidden information can be revealed using the roll up key. The HP264X function key labels will also be scrolled up at the same time.

For appending forms, the forms will continue to append with the function key labels remaining on lines 23 and 24. This is similar to how the window is handled by VPLUS.

Color Support

Like color televisions, the color terminal is undergoing a similar revolution in the 70's and 80's. Color terminals are becoming increasingly popular in the business environment. A well designed display screen employing color can improve the attractiveness and effectiveness of presentation. The added dimension may allow the expression of ideas not easily representable before or it may allow the expression of complex ideas in a simpler format using color.

VPLUS will soon provide a simple means of defining color field enhancements, error enhancements, function key labels, and window enhancements for use with the HP2627A terminal.

The HP2627A terminal has a set of 8 default color pairs as shown in Table 1. The user or application can alter these defaults by sending the appropriate escape sequences to the terminal. VPLUS will assume that the application has initialized the color pairs to the desired values if the default color pair values are not acceptable. VPLUS will not initialize the terminal's color pairs.

Eight color pairs:

color pair	foreground	background
0	white	black
1	red	black
2	green	black
3	yellow	black
4	blue	black
5	magenta	black
6	cyan	black
7	black	yellow

TABLE 1.

The Field Menu in VPLUS will provide the means for defining a color field enhancement to a field. The numeric digits 1 through 8 will be acceptable inputs into the enhancement field on the Field Menu at form design time in FORMSPEC. The fields are then enhanced at runtime corresponding to the color pairs as initialized programmatically or by the terminal's power on defaults.

Since the ENH field in the Field Menu is 4 characters long to allow for a combination of 4 different enhancements from an enhancement set of H,I,B,U,S or NONE, support for all possible combinations of color and display enhancements will not be possible. For example, a field defined to have a display enhancement combination of half bright, inverse video, underline, blinking (HIBU) will not be able to accommodate an extra numeric character for color definition.

VPLUS supports global definition of default field enhancements. A numeric digit (1-8) will also be allowed as input in the Globals Menu for the definition of a default color pair for all fields in the forms file.

VPLUS currently supports non-color error enhancements by allowing the forms designer to define the desired error enhancement in the Globals Menu in FORMSPEC at design time. The error enhancement applies to all fields in error at runtime. Like the field enhancement, the error enhancement can be defined using a combination of 4 characters containing the regular display enhancements and one color pair.

The window enhancement field in the Globals Menu in FORMSPEC also allows the definition of a 4 character combination to define the enhancement of the window. Like the field enhancements and error enhancement, the color pairs represented by numeric digits can also be entered to enhance the window line.

Color function key labels will also be supported. It has not yet been determined the method by which users will define color function key labels. However, at the time of the conference the method for displaying color function will have been determined, and will be revealed.

The last enhancement that was added with the color support was a "color lock". FORMSPEC allows color escape sequences to be added by the forms designer to enhance the text of a form to various colors when the form is being created. Unfortunately, color sequences will be stripped off the form if color forms are ENTER'ed in FORMSPEC on a terminal which does not support color. Thus, to avoid the accidental erasure of valuable work, this enhancement prevents screens from being changed if the terminal being used does not support the HP2627A color sequences.

A new field is provided in the Terminal/Language Selection Menu to specify that the forms file is to be maintained only with an HP2627A terminal. Selection of the field with an "X" indicates to FORMSPEC that the screens are not allowed to be changed if the terminal being used is not an HP2627A.

Dynamic Field Alterations

Currently, there is no way to change a field's data type, enhancement, or field type at runtime. The best way to simulate this desired alteration is to use family forms. The disadvantage to this approach is that all desired combinations of field/data/enhancement types must be decided before the application is run by creating forms with such specifications in FORMSPEC.

Soon, a new intrinsic called VCHANGEFIELD will allow field attributes, including enhancements, data type and field type, to be altered at runtime. One call to VCHANGEFIELD can be used to change the characteristics of more than one field at a time, or change more than one characteristic of a field at a time.

VCHANGEFIELD changes a field's characteristics at runtime. It is the responsibility of the programmer to ensure the new type is compatible if initial values or processing specifications are used.

Enhancements for the Interface Designer

Before I close, let me briefly mention several enhancements that will aid the users of FORMSPEC. These enhancements include: a batch mode FIELD command, a batch mode RENUMBER command, a percent full indicator on the Main Menu, a creation confirmation message on the Form File Menu, a new barcode format, a new intrinsic called VPRINTSCREEN and enhancements to the LIST commands.

Enhancements to Batch Mode FORMSPEC

Batch mode FORMSPEC was introduced in VPLUS version B.03.15. This feature allowed batch mode access to the forms file. The batch mode interface allowed users to add, copy, and delete forms, and compile and list forms files. Since the initial release of batch mode FORMSPEC, the RELATE command has been added both to the batch mode and interactive interface. The VPLUS lab has now added two new batch mode commands: FIELD and RENUMBER.

The FIELD command allows a batch job to update the field name, field enhancement, field type, data type and initial value. These updates correspond to the 5 input fields on the Field Menu in FORMSPEC. The syntax of this new command is shown on the next page.

```
FIELD formname {fieldtag}      [new fieldname],
               {old fieldname} [enhancement],
                               [field type],
                               [data type],
                               [initial value]
```

For example,

```
FIELD FORM1 FIELDA ,NONE,D,DIG
```

updates FIELDA on FORM1 to have the display enhancements of none, the field type of display only and the data type of DIGIT. Notice that the field name remained unchanged as did the initial value. The commas must be included to delimit the fields if field characteristics appearing later in the command are to be changed.

The RENUMBER command, also available from batch mode FORMSPEC, allows forms designers to reassign a form's field numbers to be equal to its screen order numbers. When forms are frequently modified, a form's field numbers can get very large and disorganized because field numbers are not reused. Thus a form may have fields with the screen order numbers of 1 through 8 (top left to bottom right), but with the field numbers of 27, 16, 3, 16, 64, 6, 44 and 29. This can become very confusing. The syntax of the RENUMBER command is simply RENUMBER formname. If the form named in the RENUMBER command is a parent form, the entire form family will be renumbered. If the form named in the RENUMBER command is a child form no renumbering will occur. This is because all members of a forms family must have identical field numbers.

A word of caution is necessary here. Before arbitrarily renumbering the fields in all of your forms, be sure to examine any associated applications for intrinsic calls which utilize field numbers. VGETFIELD, VPUTFIELD, VGETtype, VPUTtype, VGETFIELDINFO and VSETERROR all utilize the field number and may no longer function properly if the associated fields are RENUMBERed.

Percent Full Indicator

In a new VPLUS version a new message will appear when the Main Menu is initially displayed from the Forms File Menu or the Adjust Menu. This message will indicate forms file utilization, in a percentage. The percentage is determined by the formula:

file EOF / file limit

VPLUS has been enhanced to provide this message, so that users can be aware of the pace by which a forms file is filled. It is up to the user to realize when the forms file's EOF nears the limit and to expand the file as described in the VPLUS/3000 Manual, Chapter 3 (part number 32209-90001).

The new message appears only when the Main Menu is displayed from the Forms File Menu or Adjust Menu upon entry of FORMSPEC. It may be reviewed after modifying the forms file, by returning to the Main Menu and pressing the PREV function key (F5). This will return you to the Forms File Menu. Press ENTER. At this point you will be on the Main Menu, viewing the "percent full" message.

To get an idea of the utilized space in a forms file, consider the 19 form forms file used by FORMSPEC. If these forms were kept in a standard size forms file, the file would be about 10% full. If we were to add another copy of the Main Menu, the forms file size would increase to 11%. There are several things to note about this example. First of all, the FORMSPEC forms file contains no deleted forms. When a form is deleted from the forms file, it is not physically deleted. The space is "marked" for reuse, but is not actually released. Another point to note here is that the menus in the FORMSPEC forms file contain very few processing specifications. For these reasons, the FORMSPEC form forms file will probably contain less records than the "average" 19 form forms file.

A New Barcode Format

Soon, Interleave 2 out of 5 bar code will be supported. This bar code can be defined as the global bar code format on the 30XX Device Specifications Menu, by specifying ILV in the Global Barcode Format field. If you wish to use several types of bar code, including Interleave 2 out of 5, within one forms file, specify Interleave 2 out of 5 as the global bar code format, and the other barcode types in the Configuration Phase for the appropriate fields.

"This Forms File Does Not Exist, Press ENTER to Create File"

As many users of VPLUS know, it is very easy to enter a name in the Form File Menu in error. This is very frustrating as FORMSPEC actually creates a forms file if a file of that name does not already exist. In a new version of VPLUS, when the user enters the name of a forms file that does not already exist (either by mistake or intentionally), a message will appear on the screen stating that such a forms file does not exist. The user may at that point change the forms file name and press ENTER or simply press ENTER to confirm the desire to create a new forms file.

Printing of Composite Form Images

Currently, there is no simple way to print a complete copy of the terminal screen contents offline. The intrinsic VPRINTFORM will only print the screen design of the current form, and any initial data values. This can be a real problem for documenters of applications. In order to solve this problem, a new VPLUS intrinsic has been created titled VPRINTSCREEN. This new intrinsic will print all of the information currently on the terminal screen. This includes all forms (appended and otherwise), data, function key labels and the window contents. The information will be sent to a list file opened with the format and actual file designator SCRNLIST. SCRNLIST will be assigned to the device class LP. All escape sequences embedded in the form will be stripped.

VPRINTSCREEN will have 3 parameters: COMAREA, PRINTCNTL, and PAGECNTL. The PRINTCNTL, if set to 1, will cause the fields to be underlined in the listing. The PAGECNTL parameter will contain the carriage control code. These parameters are consistent with the parameters for the current intrinsic VPRINTFORM.

Enhancements to the LIST Command

Lastly, I would like to mention the enhancements to the LIST command in both batch mode and interactive FORMSPEC. The list command has been enhanced to print out all of the information contained in a forms file. This includes global and form level function key labels, the Globals Menu, the Terminal/ Language Selection Menu and the 30XX Device Specifications Menu. The batch mode FORMS command has also been enhanced to display the largest field number on a form, the number of bytes in the screen design, the number of child forms if the form is a parent form, and the parent name if the form is a child form. This additional information of the FORMSPEC listing should aid both in form design and documentation.

As you can see, the VPLUS lab engineers have been extremely busy in the past year. If there are other aspects of VPLUS which you would like to see changed, please let us know by the way of a Service Request. Service Requests are the way in which we determined that these were the enhancements which you wanted to see. We hope that these enhancements to VPLUS/3000 will help make friendly user interfaces easy to create and maintain.

Biography

Nancy Ofslager is the offline support engineer for the VPLUS product in the Computer Support Division in Cupertino California, with responsibilities for System Engineer training for VPLUS/3000.

A GUIDE TO SOFTWARE EVALUATION AND SELECTION

by: Roger J. Olsen
Productive Software Systems, Inc.
5617 Countryside Road
Edina, Minnesota 55436
United States of America

SUMMARY

There are many challenges facing the data processing community today. Not the least of which is, the selection of application software. Once a decision in made to purchase a particular package, there are many long term ramifications.

This paper will identify several issues that should be explored before you commit to purchasing your application packages. Your other alternatives are to write your own or possibly convert an old package. Each has its place depending upon the circumstances at your site.

The check list of issues includes:

1). Your Requirements.

2). The Make or Buy Decision.

3). The Software Itself.
- Adaptability
- Interfaces
- Reliability
- Documentation
- Maintainability
- Expandability
- Security
- Overhead
- Checks and Balances
- Recovery
- Limits
- Standards

4). The Vendor.
- Training
- Support
- Enhancements

5). The Software Contract

6). Cost vs. Benefit Analysis

The Life of a Typical System

Requirements

Conceptual Design

Detail Design

Development

Implementation

Review

New Requirements

The Requirements

A definition of the objective - the problem to be solved. Who will use the system? What are the major inputs, outputs? What are the expected volumes of inputs, outputs? What are the interface requirements? What hardware, communications, and software are required to support this application? What are the performance objectives such as on-line response times for file maintenance and data entry. What are the expected costs for development, operation, and maintenance? What are the systems benefits and can they be quantified?

The Conceptual Design

A functional description from the users point of view. This includes: the operational and management functions of the system, the systems inputs, including types of transactions, and the types of edits to be performed, the processing methods and techniques to be used, the data that will be maintained within the data base or file systems, the systems outputs, reports, screens, & purpose of each. A functional description from the D.P. point of view which includes: system flow charts, data flow diagrams, an estimate of input, output, and storage requirements, and other system interfaces.

The Detail Design

A detail description of each transaction including the data elements and the source of each. A description of the processing logic. A description of the file structures to be used (MPE, KSAM, IMAGE), and the access methods along with a volume estimate. A description of the outputs, screens, reports, files, security of access to screens & distribution of reports. A description of the backup, recovery, and audit trails. A description of additional programing required to alter a package or provide custom interfaces to other systems.

The Development Phase

Program specifications. Coding and unit testing. System testing. Documentation - Program, system, operation, user. Training - management, users, programmers, analysts, operators. Acceptance by all involved.

The Implementation Phase

Conversion of old files, or loading new data. On-the-job training and problem solving. Interpretation of data on screens and reports. The first test of your documentation.

The Review Phase

How well is it working? How much did it really cost, in dollars and time? What is the level of satisfaction? What needs to be changed and by when?

The New Requirements Phase

Now with a better understanding of the entire system including a long list of change requests, start the process all over again.

The First Real Issue -- Your Requirements

As described above it is assumed you clearly understand and have listed the minimum requirements.

You probably also have some desired, but not necessary requirements. They should also be listed.

In the process of reviewing software packages, some additional features offered by these packages may be added to your list of required or desired features. Several times a package supplier has encountered problems and developed solutions for them that you had not considered in your initial study.

The Second Real Issue -- Make or Buy

We will assume that after a thorough evaluation of what software is available on the market, you and the potential users have selected a few packages to investigate further that seem to meet the majority of your requirements. If a package meets 80% of your requirements, it is generally considered a good fit. If the package is not a 100% fit, which it rarely is, you must then decide to modify your requirements, have the vendor modify the software, or plan on modifying the software yourself. If you plan to modify the package yourself, you will want system documentation before you begin. Generally the vendor can provide modification quicker, although the charges per hour are higher than most outside contractors, and much higher than your own staff. If your staff will be doing the long term enhancements and maintenance, this is a great opportunity for them to learn the new system. Have the vendors quote to your requirements. A firm fixed price to have all the modifications made. And if they make them, how does that affect your support costs?

Before You Buy -- What about the Software Itself

Adaptability

Will it be adaptable to your changing requirements over its expected life? Is it written in a language that you consider appropriate? How machine dependent is it? Generally performance vs machine independence is a trade off. Can the files and data bases be easily changed (i.e. Cobol copylibs or $include files for all file and dataset layouts? Can your people understand how the code and system is designed? Is the code itself structured to allow easy maintenance? Does the vendor provide you with adequate documentation and or training on such things as the locking strategy employed and the naming conventions used.

Interfaces

Will it have the interfaces you need, and if not, how difficult is it to develop them. Interfaces can be online and/or batch but they should be as independent as possible between modules. This is important so that as your system grows and hardware prices continue to fall, you may want to distribute various applications across several processors. Always keep online interfaces to a minimum.

Reliability

Is the software reliable? How do you find out? Just because it has 200 or more installation, does not mean it is. Don't assume reliability! The biggest flaw of most application software today is the potential of two users trying to update the same record, and while doing so, each believe the software works, but one user overlays the other users transaction. There are several packages on the market that currently do not lock records properly. Check out the locking strategy employed in the code itself before you buy.

Documentation

What does the documentation consist of? Is it enough for your needs? How is it going to be maintained as changes occur? How much you need will vary with each type and your requirements. But don't underestimate your requirements for all the following types of documentation. This is one area where there never seems to be enough or the right type.

Types of Documentation:

User manual, operation manual, system flow charts, data flow charts, program and system manuals including control tables and files, security, cross-references (file to program, file to job stream, element to file, element to program, program to job stream, program to file, calls to program, copy or include files to program), internal audit trails, interfaces to other systems.

Documentation should be online and maintained on your computer system to accommodate changes. All of the above cross-references can be generated and maintained automatically through the use of your computer system and the ROBOT/3000 Automatic Documenter software or manually with HP's Dictionary/3000.

Is the Software Maintainable? And by whom?

The useful life of the software can be increased by several years if it is maintainable at a reasonable cost. The design of the application, the use of control tables, the language used, the documentation available, the amount of structure within the language, and the standards used to develop the entire application will determine the maintainability.

Is the Software Expandable?

Expandable or shrinkable as far as field sizes, file sizes, data sets, data bases, program segments. Can an additional field or element be added to a file or data base without recompiling the entire application, or retesting the entire system? Can an unused field, file, or data set be eliminated, or reused for another purpose?

How does the Level of Security fit your needs?

Is it compatible to your existing systems? Will it make your systems more complex than need be?

Are there any Check and Balance Programs Provided?

What is a check and balance program? These are programs that provide an audit of logical data structures, i.e. in a sales order system, each item on order will have a quantity and price. A good system would also carry a total amount on the order header and a check and balance program that would verify the order detail amounts always add up to the total. Inventory systems may allow items to be in several locations, and also carry a total on the master file. Either an online checking of the correctness of these or a batch program that would either sample or process all records to assure logical integrity.

System overhead should be a major concern!

For online response time, establish an acceptable level before you install any applications. Response time cost money, a slow machine lowers the productivity of all users. It also degrades as you put more applications on any one machine.

How do you Estimate overhead?

First locate, talk to, and preferably visit another site that is using the package you are considering. Make sure your intended use and volumes of transactions are similar. Too many sites that I've visited have real performance problems after using a package for a few months. It is very difficult to guess the actual load any application will take except by using another site as a comparison. Many times to reduce system overhead, or increase the transaction throughput, data bases must be redesigned, programs rewritten, additional programs written to convert data to the new data base. And if you do your development on the same machine that is running your production, it becomes even more difficult to arrive at a valid estimate.

An example of unacceptable overhead!

Installing a new general ledger system went great for the first two months. The updating was done by a batch program that was run nightly and was taking about one hour. Then the previous six months of data was loaded and the nightly one hour update now took twelve(12) hours even though the daily transaction rate had not increased. Obviously this was not acceptable. Since the new system was now fully operational, and the old non HP computer had been sold, the need for immediate action was required. The entire process keep getting longer each day. To make matters worse development and testing had to be done on the now overloaded machine.

After using various tools to monitor the program and the data base, it was determined the access path had to be changed and the summary amount field moved to a new data set. This required a few conversion programs, and all other programs that accessed that data set and data item to be changed. The actual analysis of which programs needed to be altered was accomplished within minutes using ROBOT/3000's online cross-reference capabilities, and a staff of two programmers were given the specification. A few long nights and a very long week end resulted in the newly structured ledger data base being in full production, with the update process time reduced to less than one hour.

You definitely want to avoid this kind of situation, if possible, by ether another users experience, or by carefully analyzing the purchased software first, and if that is not possible, make sure your have the proper tools for changing the software quickly.

Limits of the software

All systems have limits. Find out what they are. How will they affect your intended use? Several limits are imposed with the use of any file system including DBMS systems and only your use and your volumes of data will determine if they will become a problem. Your vendor should know the limits of their package and review them with you.

Recovery

What happens when the machine fails due to hardware or software? Does the application software support automatic recovery such as transaction logging? Maybe you will not want to use it now, but you may in the future. Does it support logical recovery? There are some that have been developed to support rollback recovery for MPE, KSAM, and IMAGE files. If the software does not have a logical recovery scheme, are the users willing to re-enter their lost data?

Standards

Find out and get a copy of the standards used in the design of the software. If you are to maintain it, this could save you many days, weeks, and perhaps months of learning time. Standards are subjective and written to achieve various goals. Find out what the goals were.

The Vendor -- Training, Support, Enhancements

Training

First Determine your training needs. How much, if any is available. Where is the training conducted? What type of areas does the training include (users, operators, analysts, programmers, management)? How much do the various training options cost? Is there any online training available?

Support

Determine what level of support and for what functions do you need support for. Don't buy more than you need. How much support and at what times will support be available? What type of areas does the support cover? Support for users, operators, analysts, programmers, management, or only software bugs.

Enhancements

What is the vendors track record for enhancements to the package? What are the planned enhancements? Do you have to install the enhancements to receive support? How do you install updates, if you have modified the package? A source code comparison tool is a must if that is the case. One strong feature of buying packages is the future enhancements. How much input do you have, as to the direction of these enhancements? How should you communicate these requests to the vendor?

The Software Contract -- Know What you are Signing!

Generally most software contracts have several general items that will protect you from law suits, etc. In addition to those general items, make sure the terms of support are covered. And if possible, get the option to purchase additional copies covered as well. Who knows how many machines will be using it in the years to come. The largest and several times overlooked aspect of a software contract is the liability the purchaser assumes. Although this generally is required to protect your vendor of unauthorized copies being distributed, it is a liability. Your company should have a method of safeguarding the software you buy and clearly explaining to all individuals that have access to it, what your policy is.

COST vs. BENEFIT ANALYSIS

(A MATRIX APPROACH)

Put your spread sheet program to work for you here. List your requirements down the left side of the spread sheet. Start first with your absolute needs. Next add your desired options. Next add the features that make a package more attractive. Include all needs, and don't forget about support. Now add the negative items that you have discovered. Cover all aspects including the items in this paper, and your own specific needs. Now assign a relative number to each item in the list. This could be +100 to -100. This number should represent how important each of these items are to the decision.

Now add each solution across the top of the spread sheet, vendor #1, vendor # 2, vendor #2 with modifications, in house customizing, total in house developed solution.

Now fill in the blanks by rating each solution from 1 to 10 on how well you believe the solution meets the requirements list.

Obviously after multiplying each of these and summing them, you have as nonbiased a rating for each solution as is normally possible. We have found this method a useful tool in making software selections. The main problem is still the items you overlook. Remember the more items generally yields a more accurate picture of reality.

The MATRIX Approach   Solutions for an Order Entry System

	Rel Value +100 to -100	vendor #1	vendor # 2	vendor # 2 Mods	Total in house
Absolute needs:					
allow 200 items per order	100	10	5	10	10
maximum of 25,000 customers	100	10	5	10	10
up to 99 ship to addresses	100	10	10	10	10
back orders per item	100	10	10	10	10
automatic credit approval	100	3	9	10	10
on line maintenance	100	8	10	10	10
inquiry by item	100	10	10	10	10
2500 orders per day	100	10	10	10	10
Desired features:					
ship from multiple warehouse	80	5	9	10	10
requires new HP 3000 48 or >	-50	6	1	1	10
software cost high	-100	5	2	6	10
inquiry response < 1 sec	35	7	5	5	10
online interface to AR	35	3	10	10	10
written in structured Cobol	55	1	10	10	10
proven reliability	90	8	5	1	1
Documentation Online	25	8	9	9	1
Other features:					
Easily Maintained	60	1	8	3	1
Easily Expanded	50	1	8	3	5
Compatible security	20	3	1	10	10
good checks and balances	35	1	6	7	1
design standards available	25	1	3	1	10
local training available	10	6	2	1	10
good enhancements history	35	9	3	1	1
TOTAL RATING PER SOLUTION:		8,630	10,430	10,385	9,595

** Vendor # 2 is the choice with highest rating **

Biography

Roger Olsen is the C.E.O. of Productive Software Systems, Inc (PSS) which develops and markets a family of management and productivity tools for the HP 3000 under the trade name ROBOT/3000. Roger has worked with the HP for over 6 years, and installed several application systems while serving as the director of information systems for a large manufacturing firm.

SM - SYSTEM MANAGEMENT

HOW DISPATCHING QUEUES REALLY WORK

Dave Beasley
Computer Systems Division, Hewlett Packard
Cupertino, California

SUMMARY

Have you ever wondered how the MPE scheduler decides which process should be given the CPU? Have you ever wanted to know why some batch jobs seem to have a disastrous effect on on-line response time? If so, you're not alone. This topic seems to emerge as one of the most debated subjects at every user group meeting that I've attended over the past several years. I think that the confusion about how the dispatcher really works is largely due to the fact that the internal algorithms and policies of the dispatcher/scheduler are not well documented. Based on my experience of working daily with the MPE operating system, focusing primarily on the kernel and I/O system, I would like to share with you what I have learned about this subject over the past few years. This paper will focus primarily on how the dispatcher and interrupt system interact and how the scheduling algorithm works. The performance effects of certain events, such as the execution of batch jobs, resource locking, (SIRS and control blocks), and disc I/O will be mentioned where appropriate.

THE DISPATCHER/SCHEDULER

The MPE dispatcher/scheduler, together with the interrupt system, comprise the heart and soul of the MPE kernel (note that the terms dispatcher and scheduler will be used interchangeably). The dispatcher's major responsibilities include selecting the next process to run, invoking the memory manager when a process needs memory, and keeping track of the CPU time used by each process (rescheduling and adjusting priorities accordingly). The dispatcher also invokes the memory management garbage collection routines when the system would otherwise be idle AND when that action is desireable.

The dispatcher is invoked explicitly by the DISP machine instruction and runs on a special stack called the Interrupt Control Stack, (ICS). If a DISP instruction is executed on the ICS (ie., an interrupt handler is running), the dispatcher is not entered immediately, but is deferred by setting a "dispatch requested" flag. This is to allow all interrupt handlers executing on the ICS to complete (note that we could have stacked interrupts due to one interrupt interrupting another). Each interrupt handler that executes on the ICS must do an IXIT instruction, (a special type of EXIT), when it has finished processing. When all interrupts have been processed, the last IXIT executed must decide whether or not to enter the dispatcher code by "EXIT'ing" through a special "dispatcher stack marker", or to return to whatever was interrupted.

When no process is running, the CPU is either executing in an interrupt handler and the I/O completion routines that it may call, in the dispatcher, or in the memory management routines, (called by the dispatcher). Before proceeding with the algorithm and policies of the scheduler, we must define what a process actually consists of and discuss some key points about the interrupt system.

WHAT IS A PROCESS?

As the well informed HP3000 user, which I'm sure you are, you must have read that "a process is the unique execution of a program at any given time". This statement is absolutely correct, but I'm going to expound on this definition based on the outside chance that even one person reading this may be a little confused. Each time that you logon, a Command Interpreter process is created for you by the system. Each time that you :RUN a program, a new process is created. If an executing process CREATES and ACTIVATES a son program, a new and distinct process is created.

Each process on the system has some unique attributes. Each one has its own unique stack for its own private copy of the program's data. Each process has one or more code segments associated with it. Since the HP3000 allows code sharing, these code segments may be shared between processes running the same program or sharing SL segments. Each process may also have one or more extra data segments associated with it, some of which may be needed by MPE for system tables and file control blocks, and possibly some that are acquired by the program for its own use.

Each process is also given an entry in the Process Control Block Table, (PCB). The PCB table contains all of the status and necessary information that the dispatcher needs to decide which process should be given the CPU next. Each PCB entry contains critical information about the process. For example, the PCB entry informs the dispatcher whether the process is ready to run and only needs the CPU, or if the process is blocked, waiting for some event to occur. Some of the events that a process could be waiting on are I/O waits, memory waits, or critical resource waits, (ie., a SIR or file control block wait). It is the job of the dispatcher to schedule processes to run based on the information found in the PCB table.

THE INTERRUPT SYSTEM

It is well beyond the scope of this paper to fully explore the interrupt system, however it is essential that you understand some of the key points. MPE is an interrupt driven operating system, meaning that MPE does not "go around looking" for things to do, but rather is informed of pending activity and/or the completion of certain events by way of an interrupt from some device or a "message" from some part of the operating system.

There are two basic types of interrupts on the HP3000: internal and external. Internal interrupts are a result of the CPU and its microcode detecting some unusual or abnormal condition. Some examples of internal interrupts are stack overflows, absence traps, (referencing a segment that is not in memory), integer overflows, and bounds violations. There are many others. In most cases, these internal interrupts are handled directly on the user's stack, but some of the interrupts must be handled on the ICS. Additionally, in certain instances, the dispatcher must be invoked on behalf of the process to perform some function, such as initiating and handling a stack overflow or absence trap.

The second type of interrupt is called an external interrupt. These can be thought of as device interrupts. External interrupts are always processed by their interrupt handlers on the ICS. External interrupts usually indicate a change in the status of an I/O device, such as a magnetic tape being placed on-line, or the completion of an I/O operation, such as a disc read or write.

The external interrupt handlers and the I/O completion procedures that they typically call, may call the procedure AWAKE to signify that some event has completed. AWAKE will indicate in the appropriate PCB entry that the event has happened; if it was the event for which the process was waiting, then AWAKE will request that the process be inserted into the dispatcher's list of processes waiting for the CPU. AWAKE may or may not do a DISP instruction at this time. We will discuss AWAKE's role in process preemption and in communication with the dispatcher later.

POLICIES OF THE MPE SCHEDULER

All decisions about CPU scheduling are made by the scheduler. The scheduler maintains a DISPATCH queue of all processes in the PCB table that are READY to run. Being READY to run implies that a process does not need to wait for some event to occur or for some system resource to become available before it is able to execute. All that a READY process needs is to have the CPU allocated to it. Now that I've said this, there are two exceptions to the previous statement about READY processes on the DISPATCH queue. A process that is "short waited" for disc I/O is not taken off of the DISPATCH queue since those waits are usually satisfied so quickly that it is not worth the effort to take them off. Of course, they will not be launched until the I/O is complete. The other exception is that processes waiting on memory resources are also on the DISPATCH queue, since the dispatcher is responsible for initiating memory management activity.

A process which becomes READY, due to the completion of some event or some resource becoming available, is inserted into the DISPATCH queue in priority order. The scheduler will ALWAYS ATTEMPT to allocate the CPU to the highest priority process that is READY to run.

Under certain circumstances, a higher priority process may become READY while a lower priority process is executing. This can occur when an executing process is interrupted, and the reason for the interrupt satisfies the "wait reason" for a higher priority process. After AWAKE requests that the process be inserted into the DISPATCH queue, it must determine if it should take the CPU away from the current process in order to allow the dispatcher to run the higher priority process, or if it should allow the lower priority process to run until it blocks naturally. There are some specific rules that AWAKE must follow with respect to process switching, which we will discuss later. The point here is that if the process switch does not occur, this is the only time that the highest priority process does not run, and it's a very temporary condition.

WHEN DOES A PROCESS GIVE UP THE CPU?

A process will run until it blocks naturally, is preempted by a higher priority process becoming ready for the CPU, or until the dispatcher decides that the process has had the CPU long enough.

BLOCKING NATURALLY

To block naturally simply means that the process requested some resource that wasn't available, such as a SIR, a lock on a file control block, a system buffer,etc. A process will also block naturally when it has requested I/O or if it needs memory resources. A process requests to be blocked by calling the procedure WAIT, specifying what type of event it expects to occur before it can execute again. WAIT stores this reason in the PCB entry for the process. When the procedure AWAKE is called to awaken a process for a particular event, it will update the PCB entry to indicate that the event has occured. If the event corresponds to the reason why the process was waiting, AWAKE will request that the process be placed in the DISPATCH queue of READY processes.

PROCESS PREEMPTION

A process is said to be preempted when the CPU is taken away from it, before it blocks naturally, in order to allocate the CPU to a higher priority process. After verifying that the reason the new process is being awakened corresponds to the reason that the process was originally blocked, AWAKE requests that the process be inserted into the DISPATCH queue in priority order. If AWAKE determines that the new process being awakened is of higher priority than the one that has been interrupted, AND IF the dispatcher has enabled process preemption for that process, AWAKE will do a DISP to initiate the process switch. We will discuss the rules for process preemption and the role of the procedure AWAKE in this effort after we discuss the characteristics of the scheduling queues.

HOW DOES THE DISPATCHER DECIDE HOW MUCH IS TOO MUCH?

Every 100 milliseconds, the system clock generates an interrupt, which causes control of the CPU to be transferred from the executing process to the ICS, where the interrupt handler for the system clock will execute. Each time that a process is launched, it is given three "ticks". In other words, a variable, which we'll call QTIME, gets set to 3, representing 300 milliseconds of CPU time. Each time that the clock interrupt handler executes, it decrements QTIME. Whenever QTIME falls to zero, a DISP instruction is executed to invoke the dispatcher. The dispatcher will then treat this process just as if it had voluntarily given up the CPU or had been preempted. The dispatcher will reschedule this process, adjusting its priority if necessary, and will select the highest priority process on the DISPATCH queue to launch. Note that the new highest priority process may be the same one that the clock interrupt handler preempted because QTIME was zero.

Many people still confuse the above method of allowing the clock interrupt handler to preempt a process with the idea of the process using a "quantum". In fact, under the MPE III operating system, we did call it that! There was even a :QUANTUM command that allowed the system manager to determine how many "ticks" or milliseconds a process would get as its "quantum". However, it is important to note that this method is not used to determine a "quantum" in MPE IV and MPE V! The dispatcher uses different criteria to determine if the process has used more than its "fair share" of the CPU, (or has exceeded its "filter" value). Preempting a process because it has used 300 milliseconds of consecutive CPU time is only done to allow the scheduler to reschedule this CPU intensive process to ensure that it does not exceed the "filter value" for its scheduling queue.

SCHEDULING QUEUE CHARACTERISTICS

There are five distinct scheduling subqueues in MPE. The AS and BS queues are considered to be linear queues, meaning that the scheduling algorithm makes no priority adjustments to processes in these subqueues. The CS, DS, and ES subqueues have always been considered to be "circular" queues. This term originated because of the way that priorities of processes in these queues gradually moved up and down, resembling a "circular" shape, in MPE III. This is not the case in MPE IV and MPE V, however much of the documentation still refers to these queues as "circular" queues, (if you must make a geometric analogy, they probably appear more like "semi-circles"). The scheduler controls the priority of processes in these queues.

The scheduling queue priorities range from 0 to 255, with 255 being the lowest priority. When a process is created, it is scheduled by default into the same scheduling queue as its father. However, the scheduling queue and/or the absolute priority number of the new process may be specified in the CREATE, CREATEPROCESS, or GETPRIORITY intrinsics.

AS AND BS SUBQUEUES

The priority range for the AS subqueue is from 0 to 100. This queue is typically reserved for high priority MPE processes. The priority range for the BS subqueue is from 101 to 149. Some MPE processes run in this queue also. This queue may also be used by high priority user processes if the user specifies. With the proper capabilities, a user can specify any priority for any process. It should be pointed out, however, that it is possible to deadlock the system or to adversely effect system performance by using these subqueues, particularly the AS queue. Once a priority has been assigned to a process in one of these linear queues, the scheduling algorithm does not alter it.

CS SUBQUEUE

The strategy of the CS subqueue scheduling algorithm attempts to favor interactive users. The default range of priorities in this queue is from 152, which is called CBASE, to 202, which is called CLIMIT. These values may be altered by the :TUNE command.

A process' priority is dynamic within this subqueue and is controlled by the scheduler. Whenever a process blocks for a terminal read, it is rescheduled at CBASE priority. Therefore, the first time that a process runs after being awakened from a terminal read, it is launched at CBASE priority. A process may use the CPU and then block, or give it up, several times before it blocks on another terminal read. For example, a process may block for several disc I/O's or it may be impeded for some system resource before it requests another terminal read. We call the CPU time used between terminal reads, a "transaction".

A process is rescheduled each time that it gives up the CPU, (or quiesces). Its priority may be decremented, (incremented numerically), if it exceeds the filter value for the CS subqueue, or if it blocks due to a memory trap. In MPE V, its priority is not decremented due to a memory trap, but only for exceeding the filter value. Additionally, a process in the CS queue is enabled for preemption if it exceeds the filter value. Once the priority of a process is decremented, it is never incremented again until it blocks for a terminal read, (which completes a "transaction"), and it is assigned the CBASE priority at that time. A process' priority may never be decremented below the

CLIMIT priority. There is one exception to the scheduling algorithm which may cause a process' priority to be raised without having blocked for a terminal read. We will discuss this exception after the discussion of each of the scheduling queues.

How does the scheduler determine the filter value for the CS subqueue? The filter value is also called the "average short transaction" time, or AST. The AST is a computed average of the time required for all "transactions" to complete for every interactive process on the system. Remember that a "transaction" is defined to be the amount of CPU time used between terminal reads. Each time that a process blocks, or is preempted, the scheduler keeps track of the CPU time used since the last time the process was awakened from a terminal read, therefore, when the process blocks for a terminal read again, the scheduler knows the time used for the current transaction. A formula is then used to calculate a weighted AST which is effected by each interactive process on the system. Processes in the DS and ES queues which are running interactively, and therefore complete transactions, also contribute to the AST, even though the AST value is not used as the filter value for processes in the DS and ES queues. The AST value will never be allowed to exceed the CQ MINQUANTUM and CS MAXQUANTUM limits, which are set by default to 0 and 300 milliseconds respectively, or which may be set by the :TUNE command.

DS AND ES SUBQUEUES

The DS and ES subqueues are primarily intended for batch jobs, although sessions may run in these queues if so desired. The default range of priorities for the DS queue is from 202, which is called DBASE, to 248, which is called DLIMIT. The default EBASE value is 250, and the default ELIMIT value is 255. These values may be altered by the :TUNE command.

When processes are created in either of these subqueues, they are originally placed on the dispatch queue at their BASE priority. Each time that a process in one of these queues blocks for any reason, if the total CPU time accumulated exceeds the "background filter", its priority is decremented, (numerically incremented by one). In MPE IV, its priority is also decremented, (numerically incremented by one), if it blocks for a memory absence trap, (this does not happen in MPE V). A process' priority will never go below the LIMIT value for its scheduling queue. When a process is not executing interactively, such as a job, once its priority has been lowered it will not be raised again by the scheduling algorithm. Again, the same exception that applies to the other queues exists for the DS and ES queues also. We will discuss the exception shortly. If a process is executing interactively, and it is awakened from a terminal read, it will be placed on the dispatch queue at the BASE priority for its scheduling queue.

How is the "background filter" determined for the DS and ES subqueues? Although the :TUNE command allows you to specify a MINQUANTUM and MAXQUANTUM value to be used as the filter values for these queues, the filter value is constant and is the value specified as the MAXQUANTUM for the DS queue.

THE BIG EXCEPTION

There is one exception to all of the above rules for scheduling processes. A process in any subqueue may have its priority temporarily raised if it is the holder of certain critical resources and another higher priority process wants one or more of those resources. The only two resources for which this exception applies are SIR's and IMAGE data base control blocks.

A SIR, (System Internal Resource), is a semaphore that is used to guard against unwanted concurrent access to certain system resources, such as system tables or the system directory. If one process is the holder of a particular SIR that another process also wants, the second process must wait until the first process releases the SIR. In MPE IV, this "SIR queue" is priority based. For example, if a higher priority process queues up for a SIR which is held by a lower priority process, the lower priority process is given "SIR priority". In other words, its priority is temporarily raised to that of the higher priority process. As soon as the SIR is released, the higher priority process will be given the SIR and the process that was given "SIR priority" is returned to its original priority. Intervening processes in the "SIR queue" will remained queued, but the higher priority process will take precedance in gaining access to the SIR. In MPE V, the "SIR queue" has been changed to a FIFO based queue. If a higher priority process queues for a SIR which is held by a lower priority process, then the process holding the SIR and all intervening processes queued for the SIR will be given "SIR priority". The SIR will then be given to each of the processes in the order that they queued for it.

If a process is the holder of an IMAGE data base control block, and a higher priority process requests it, the holder and all intervening processes that are queued for the control block have their priorities raised to that of the higher priority process requesting the control block. Once each process has released the resource, its priority is lowered back to its original priority. Note that as of MPE V/E, processes in the CS, DS, or ES queues, will not have their priorities raised above the CBASE priority due to impeding a higher priority process for an image control block.

There are many other kinds of resources that processes may queue up for. Some examples of these are RINS, locks for file system control blocks, message buffers, IOQ and DRQ entries if too few are configured, and many more. Note that higher priority processes simply have to wait behind lower priority processes for these types of resources.

WHAT DOES AWAKE DO WHEN A PROCESS BECOMES READY?

If the dispatcher is running and is in PAUSE, meaning that there are no processes currently needing the CPU, AWAKE will do a DISP instruction to restart the dispatcher. The dispatcher will then scan the list of READY processes and launch the one with the highest priority, (which would in this case be the one we have just awakened).

If the dispatcher is running, but is not in PAUSE, it is either preparing a process to launch, it has called the memory management routines on behalf of some process, or it is calling memory garbage collection routines. If this is the case, AWAKE simply places the priority of the process it is trying to awaken in a special memory location that is reserved for communication between AWAKE and the dispatcher. The dispatcher will look at this location at convenient points to determine if there is a more urgent process to launch than the activity that is currently in progress.

If the dispatcher is not running, it means that a process is currently running. Actually, it means that we have interrupted the last process launched by the dispatcher. AWAKE must decide whether or not to preempt this process and initiate a process switch, or to allow the executing process to run until it blocks naturally. The following rules apply to process preemption.

1) Any process in a linear queue may be preempted by any process that has a higher priority, regardless of the priority of the scheduling queue that it is in.

2) Any process in any queue may be preempted by a higher priority process that is also in a higher priority scheduling queue. For example, the CS queue has a higher priority than the DS queue which is higher than the ES queue.

3) A process in the CS queue may be preempted by a higher priority process in the CS, DS, or ES queue if and only if it has been enabled for preemption. The dispatcher enables CS processes for preemption only when they have exceeded the AST, (average short transaction time).

4) Processes in the DS and ES queues are never enabled for preemption which means that they can never be preempted by processes in their same queue, regardless of their priority.

PRIORITIZED DISC I/O

Since the introduction of MPE IV, disc I/O has been prioritized. If a process issues a disc request, the disc request is issued at the priority of the process requesting the I/O. Memory management I/O is also issued at the priority of the process that needs the memory. Background writes, or anticipatory writes, are issued by the memory manager at a background priority, which is 255. Background writes are issued in order to update the copy of a data segment on disc in anticipation of having to swap out the data segment. Since the segment may be "recovered" in memory, therefore invalidating the need for the disc write, these types of writes are performed in the background, so as not to interfere with useful I/O. If the memory manager decides that the memory space is needed, and if the background write has not completed, then the priority of the disc request for the background write is bumped up to the priority of the process that needs the memory space.

PERFORMANCE IMPLICATIONS OF BATCHJOBS

There are several characteristics of the scheduling algorithm and scheduling queues which are quite good at attempting to ensure that batchjobs really do execute in the background. (Although, with the :TUNE command, you can arrange the scheduling queue limits to allow jobs to compete with sessions, or to even be favored). One characteristic in particular is that the scheduler allows higher priority processes in higher priority scheduling queues to preempt lower priority processes as soon as they become ready for the CPU. For example, while one process in the CS queue is waiting for terminal I/O, a CPU intensive batch job in the DS queue may be executing. However, as soon as the terminal read is completed, the higher priority CS queue process will preempt the DS queue process, thus favoring the interactive user. Another mechanism which prevents CPU intensive processes from adversely effecting system performance is the system clock interrupt handler, which prevents a process from using more than 300 milliseconds of CPU time consecutively without being rescheduled. Another characteristic which favors the interactive user is prioritized disc I/O, which tends to allow I/O from interactive sessions to take precedence over I/O from lower priority batch processes. However, in spite of the attempts to ensure that sessions are favored over batch jobs in scheduling, low priority processes that compete for the same resources as higher priority processes can seriously effect performance on busy systems.

Once a batch job, or low priority process, acquires a critical resource that is needed by a higher priority process, the higher priority process must simply wait until the resource is released. If there are a large number of processes on the dispatch queue ahead of the lower priority process, obviously it cannot run in order to release the resource. To complicate the scenario, if the low priority process has outstanding disc I/O, and if there is a significant amount of I/O being issued at higher priorities, the low priority process that holds the resource may have to wait a considerable amount of time for the I/O to complete. In the mean time, the higher priority process that wants the resource will simply have to wait, as well as any other process that queues for the resource.

As we have previously discussed, an attempt was made by the scheduling algorithm to give low priority processes a higher priority, temporarily, when they are impeding higher priority processes. This was only done for two resources, however; SIR's and IMAGE control blocks. Although this works well in most cases, if the lower priority process had previously issued disc I/O which was not complete, the priority of the outstanding disc I/O is not bumped up in priority. Therefore, if a considerable amount of disc I/O is being queued up at a higher priority, the I/O request for the lower priority process that holds the resource may be delayed, in spite of the fact that its CPU priority had been temporarily increased.

CONCLUSION

As you can plainly see, there are many variables in determining how one or more batch jobs will effect the performance of a system. The most obvious ones are the total number of processes on the system and the scheduling queues that they execute in, and the amount of disc I/O that is queued up at any one time. What is not so obvious is what effect competition for the same resources, by sessions and batch jobs, have on the scheduling algorithm.

I hope that this paper has helped you understand this to some degree.

It is almost impossible to recommend "rules" and "guidelines" about what is the ideal mix of batch jobs and sessions. The only recommendation that I will make is that you attempt to minimize resource competition between jobs and sessions. One way in which this can be done is by minimizing, or not running, jobs during peak hours that utilize the same files and data bases as online users. Of course, it will be impossible to eliminate competition unless you simply defer all jobs to off hours. This assumes that you have "off hours" in your shop! Good luck and happy tuning!

Dave Beasley
is a Systems Support Engineer for Computer Systems Division of Hewlett Packard in Cupertino, California. Dave has worked for Hewlett Packard for 6 years and has been a user of HP3000's for 8 years.

Interfacing the MPE Man with the UNIX* Machine

Sam Boles, Member Technical Staff

HEWLETT PACKARD

With UNIX™ evolving as the de facto "industry standard" operating system, Hewlett-Packard now includes this important dimension in its array of computer technology. The Series 200 and 500 of the HP9000 family currently support HP-UX, a powerful dialect of UNIX™, with more under development.

The charter of the HP Productivity Center in Cupertino, California, includes the integration of HP3000 and HP9000 computers. In the light of this experience you can get a view of UNIX™ thru the eyes of MPE. The friendly vernacular of MPE -- second language to HP3000 users around the world -- becomes the familiar basis in terms of which those new to UNIX™ can acquaint themselves with the terse power of this operating system.

Starting with fundamentals that map one-to-one, you'll see some MPE UDC's used with HP summer students to accelerate their productivity by providing a transitional mechanism from their UNIX™ background. From there you'll move into the more complex facilities that the UNIX™ productivity engine gives to both programmers and end-users alike.

... the Hewlett-Packard Productivity Center: getting Performance to the Bottom Line ...

At the Hewlett-Packard Productivity Center in Cupertino, California, we deal with a ***wide range*** of hardware and software products. This is part of getting the right tool to do the right job and to interact with the other tools in the right way. That's our job: ***Productivity: Getting Performance to the Bottom Line***.

In the course of doing this, we integrate into our Productivity Network a wide range of Hewlett-Packard computers. A component of this task is to make the HP3000 and the HP9000 play together. This means that the engineers working in the Productivity Center need to be ***multi-lingual***: conversant in ***MPE, UNIX™*** and other operating systems.

Much of our HP9000 work at the Productivity Center uses the ***HP-UX*** operating system. Both the 16-bit Series 200 and the 32-bit Series 500 support HP-UX. HP-UX is a powerful UNIX™ dialect that is based on the Bell Labs ***System III***, enhanced with features from the ***Berkeley 4.1*** and the Bell ***System V***, plus some Hewlett-Packard contributions to the UNIX™ evolution.

The Productivity Center leverages the ***SEED*** (Student Employment and Educational Development) program that Hewlett-Packard has sponsored for many years. The SEED program is designed for the mutual benefit of Hewlett-Packard and outstanding students in universities around the world.

**UNIX™ is a trademark of AT&T Bell Laboratories.*

The student gains the benefit of experiencing a real-world job doing real-world work on a real-world project. The company gains the benefit of bringing bright fresh creativity to its product development from youngsters who've not yet learned that ***a given task is impossible***. Since they don't have the experience to know that it's impossible they sometimes actually ***get it done***. Or come up with a work-around we hadn't thought of before.

Sure, we ***throw a lot of stuff away***. But once in a while we get ***some super products***. Probably at about the standard R&D hit ratio.

. . . MPE UDC's to leverage the SEED student's knowledge of UNIX™ . . .

Coming from the ***university community*** our SEED students typically have UNIX™ experience. UNIX™ was born in the quasi-university environment of Bell Labs, and some of its great evolutionary contributions have come from universities such as the University of California at Berkeley.

In order to ***leverage this knowledge*** and to ***expedite SEED productivity*** in the MPE environment, we have some MPE UDC's (User Defined Commands) that map fairly closely to what the SEED student is accustomed to under UNIX™. This provides a transitional mechanism that enables the SEED student to ***focus more on*** the meat of ***the project than*** on some of the accidental properties of ***the operating environment***.

This same transitional vehicle, if we shift it into reverse gear, can give an experienced MPE user a glimpse of UNIX™. That's what we do in this paper.

The initial emphasis is on the functional similarities of the two operating systems. However you can't deal with the similarities for very long without touching on the differences.

Here you get a superficial introduction to some of the powerful UNIX™ features like ***piping, redirection and the UNIX™ file system***; and, to give equal time, some of the MPE features missing from some of the popular shells (command interpreters), such as the MPE REDO (extensively used by some of us heavy-fingered folks whose data-processing lives tend to be as redundant as they are heavy-fingered.)

. . . since 1969 . . . a little UNIX™ lore, mystique and culture . . .

Before we get into the UDC's let's look at a little UNIX™ lore, mystique and culture. Anything that's survived in this business since 1969 (for the children among us, that's the year IBM "unbundled" in recognition of the fact that software was no longer just the packing that came free with the hardware to keep it from rattling around in the carton) has some lore and deserves a little nostalgia.

First of all, it's an operating system ***of the programmers, for the programmers, by the programmers***. That's why we ***byte hacks love it***. And that's why ***civilized people spend large sums of money for*** commercial ***shells to cover*** the lean terse power of UNIX™.

Some say it's ***unfriendly***. But you have to remember that one man's "friendly" may be another man's "verbose." It's the ***classic issue of efficiency vs friendliness***. Do you have an elaborate high-overhead ritual to establish friendliness or do you get right to the point? Do you ask for a confirmation that the user really wants to purge every file in his group, or do you assume that if he's smart enough to ask for it you'll do it for him?

You can look at it this way: like a lot of development nodes, in my section in Cupertino, we save trees by running OUTFENCE high, going to hardcopy on only a small portion of our output. That means to get hardcopy you need

```
ALTSPOOLFILE #Onnn;PRI=nn
```

Now that's ***mnemonic and intuitive***, probably. It's maybe what you'd call ***friendly***. But about the third time you do it, you decide it's worth the trouble of updating your UDC's with something like

```
ASF SPL=0,PRI=1,COPIES=1
OPTION LIST
ALTSPOOLFILE #O!SPL;PRI=!PRI;COPIES=!COPIES
```

that you invoke with

```
ASF nnn nn
```

That's the UNIX™ style. It's ***terse***. All right, ***cryptic***. Powerful. A rich repertoire of commands and options to do the kinds of things programmers do to build and document software.

. . . maybe something of a misnomer: today, there's little UNI in UNIX™ beyond UNIfying . . .

Next, UNIX™ may be a ***misnomer***. Legend has it that when the brilliant Ken Thompson named his brilliant child, he did it in counterpoint to the multi-tasking multi-user MULTICS at MIT. His was a single-user system for a single-engineer work station. Today there is ***little UNI*** in UNIX™ beyond the fact that it may be the single most ***UNI***fying element across the wide variety of hardware architectures and configurations in the industry today. Beyond that great ***UNI***fying attribute and signal contribution, UNIX™ is ***MULTItasking, MULTI-user, MULTI-noded, MULTI-shelled, even MULTI-processed*** on the HP9000/500 with its tri-CPU design.

One aspect of the "non-UNI" of UNIX™ is its ***multiplicity of dialects***. This is probably good and bad at the same time. Like any worthwhile software system, UNIX™ is evolving. It's inevitable that such a magnificent theme have myriad variations. The Bell Labs UNIX™, perhaps the seat of orthodoxy, has a System III and a System V. There's the Berkeley 4.1. And HP-UX, XENIX, VENIX, QNX, UNI-plus, -star, -plex and, of course, the powerful NIX of the anti-UNIX™ clingons.

Now no one disputes the value to mankind of the ***c-shell*** re-do (painfully missing in some Shells), but when you transition from Brand X to Brand Y, is it there or isn't it? Not a big deal. But neither is Life, for that matter. Just a fabric (or hodge-podge) of little deals -- but if they gang up on you, you're in trouble.

Now we can't hold back tomorrow. We don't even want to. Just ***don't be deluded*** into thinking that the UNIX™ "industry standard" is going to get you entirely out of the ***technological retread business*** we've been confronted with for generations (computer, that is), with all its learning curve entropy and proactive inhibitions.

As Churchill once said about Democracy: ***It's not perfect by any means; it's just the best we've been able to come up with.***

Of course, you could see it coming. Back in ancient times, on the 1401, you could put a complete program on a single 80-column card to read-and-list cards. It started out

```
,008015 . . .
```

The comma set a word-mark to give you a "variable word-length" machine. You'd slap that one card on the front of your "source deck" to do a

```
1,$p
```

(That's UNIX™ editor for "/LIST ALL.") When the 360 arrived it had a thing called an "operating system" and you couldn't do that with just one card anymore. The beginning of the end.

So much for yesterday. Let's take a look at tomorrow.

... UNIX™ and MPE side by side, going thru a few ordinary everyday commands ...

First a few words about the examples you see here. We all know the frustration of the ***example that doesn't work***. One way to reduce that problem is to capture the example right at the screen ***in vivo***.

Now to do this for the MPE part is a fairly straightforward exercise if you have an old 2647 like mine. You work the example on the terminal on-line to the computer, then position the cursor at the start of the example. You go into local command mode to do a

```
COPY ALL FROM DISPLAY TO LEFT TAPE
```

When you've gotten the latest batch of examples on tape, you do a

```
MARK LEFT TAPE

REWIND LEFT TAPE
```

Then you get into TDP (Text & Document Processor), find the place where you want to splice in the example, do an

```
/A nnn.nn
```

Then when you get the line number prompt, touch the READ key to get the cartridge tape contents spliced into your text file.

Getting the UNIX™ examples is a little different. If you're using an HP9000/520, you've got an integrated 5" floppy disc drive. As you do the examples you precede the example with an `echo` or a `cat >>` to the disc file where you're collecting your actual examples. For example, for the `ps` (**p**rocess **s**how) command:

```
echo $ ps -e >> seb
```

(The `>>` means to append or concatenate the string after `echo` to the target file `seb`.) Then you actually do the command but redirect the output to the same file:

```
ps -e >> seb
```

This gives you what would have been on the screen in your disc file. Then, if you haven't bothered to engineer any better datacom, you can

```
lifcp seb /dev/rfd:SEB
```

to get your examples onto a floppy in LIF (Logical Interchange Format), take it over to your HP9000/236, do a virtual terminal file transfer to the HP3000 where the main body of your text is for doing your laser typesetting via TDP, and join the examples into the appropriate spot.

In the examples you see the HP-UX form, then the MPE form with an HP-UX-like UDC. In the UDC there's an `option list` to show you how the UDC gets expanded and executed. Imagine a Carriage/Cursor Return at the end of each line unless specified otherwise. If there's a ***Control-D*** you'll see `[ctl-D]`.

So much for the logistics. Let's get started. First, get on the system:

HP-UX:

```
login: boles

Welcome to Hewlett-Packard System 9000 HP-UX
```

MPE:

```
:hello boles.cad
HP3000 / MPE V  G.B0.00 (BASE G.B0.00).  MON, DEC 24, 1984,  4:24 PM
```

Accounting in HP-UX is done generally at the ***user*** level, as opposed to ***user.account*** in MPE. There are some other differences, too. For example, if you have a password and key it wrong, MPE asks you several times to try again; HP-UX doesn't tell you whether it's the user or the password or a backspace that's the problem, but asks for everything again.

HP-UX:

```
$ who am i
boles    console Dec 24 13:26
```

MPE:

```
:whoami
SHOWME
USER: #S85,BOLES.CAD,UNIX      (NOT IN BREAK)
MPE VERSION: HP32033G.B0.00.  (BASE G.B0.00).
CURRENT: MON, DEC 24, 1984,  4:26 PM
LOGON:   MON, DEC 24, 1984,  4:24 PM
CPU SECONDS: 5          CONNECT MINUTES: 2
$STDIN LDEV: 22          $STDLIST LDEV: 22
```

Notice the blanks are suppressed in the UDC to get the `showme`.

HP-UX:

```
$ date
Mon Dec 24 13:48:48 PST 1984
```

MPE:

```
:date
SHOWTIME
MON, DEC 24, 1984,  4:26 PM
```

Basically the same but a little more time granularity in HP-UX and a GMT (Greenwich Mean Time) basis.

HP-UX:

```
$ ps -e
   PID TTY  TIME COMMAND
 27335  co  0:00 ps
 27279  co  0:04 sh
    35   ?  0:01 getty
    34  a2  0:02 getty
    33  a1  0:01 getty
    32  a0  0:01 getty
     1   ?  0:01 init
```

MPE:

```
:pse
SHOWJOB

JOBNUM  STATE IPRI JIN  JLIST    INTRODUCED  JOB NAME

#S69    EXEC        20  20       FRI  8:54A  OPERATOR.SYS
#S85    EXEC        22  22       MON  4:24P  BOLES.CAD

2 JOBS:
    0 INTRO
    0 WAIT; INCL 0 DEFERRED
    2 EXEC; INCL 2 SESSIONS
    0 SUSP
JOBFENCE= 0; JLIMIT= 5; SLIMIT= 60
```

The `ps`, like `showjob`, tells you what's running in the system. Some differences are cosmetic: syntax, format, nomenclature; but CPU consumption, state and start time are all useful but not available in both systems with these comparable commands.

HP-UX:

```
$ ls
seb
sebb
```

MPE:

```
:ls
LISTF @

FILENAME

SEBUNX
```

Again, mostly cosmetic differences.

HP-UX:

```
$ ll
total 3
-rw-rw-rw-   1 boles    101         370 Dec 24 13:51 seb
-rw-rw-rw-   1 boles    101         100 Dec 24 13:48 sebb
-rw-rw-rw-   1 boles    101         365 Dec 24 13:51 sebc
```

MPE:

```
:ll
LISTF @,2
ACCOUNT=  CAD          GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

SEBUNX  *          72B  FA          48         48   7       16  1  1
```

The HP-UX "long" file list gives security, date and owner. The *-rw-rw-rw-* means it's an ordinary data file (not a directory nor a device special file), the owner of the file has read and write access but not execute permission, as do the user's group and the public in general. The listing also includes number of directory links, owner, group code, size in bytes, date and time of last modification.

This touches on a major difference: the file system. The UNIX™ directory structure and file concepts are a major transitional consideration, and beyond the scope of this paper. You get glimpses here in the links information and in the `mkdir` ***and*** `cd` ***examples below. But remember this is only the tip -- there's a real iceberg there.***

HP-UX:

```
$ cat
This is to show cat with no parms.
This is to show cat with no parms.
This is line 2 of show cat.
This is line 2 of show cat.
[ctl-D]
```

MPE:

```
:cat
FCOPY FROM=;TO=
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983


This is to show cat with no parms.
This is to show cat with no parms.
This is line 2 of show cat.
This is line 2 of show cat.
 < CONTROL Y >

2 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

This is the `cat` (for con***cat***enate) form without parameters. It's basically input from $STDIN and output to $STDLIST -- the CRT in this case.

HP-UX:

```
$ cat > file1
This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
[ctl-D]
```

MPE:

```
:catt file1
FCOPY FROM=;TO=file1;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
 < CONTROL Y >

3 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

This is the concatenation of CRT input to a new or replaced file on disc. An easy way to build a file without getting into the editor -- but you give up the more powerful edits. Notice the >. That's UNIX™ redirection from the default CRT to the named file. Be careful: UNIX™ has high regard for your presence of mind. If it finds a file out there already by that name, it doesn't ask as MPE does whether you're sure you want to purge it (unless you've removed the write permission with a chmod) -- it just writes over the old file.

HP-UX:

```
$ cat file1
This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
```

MPE:

```
:catf file1
FCOPY FROM=file1;TO=
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983


This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
EOF FOUND IN FROMFILE AFTER RECORD 2

3 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

Here with an implicit < redirection of input, we concatenate from the named file to the CRT.

HP-UX:

```
$ cp file1 file2
$ ll file*
-rw-rw-rw-   1 boles    101          121 Dec 24 14:01 file1
-rw-rw-rw-   1 boles    101          121 Dec 25 20:44 file2
$ cat file2
This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
```

MPE:

```
:cp file1 file2
FCOPY FROM=file1;TO=file2;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

EOF FOUND IN FROMFILE AFTER RECORD 2

3 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
:ll file@
LISTF file@,2
ACCOUNT=  CAD         GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

FILE1              80B  FA           3       1023   1      128  1  8
FILE2              80B  FA           3       1023   1      128  1  8


:catf file2
FCOPY FROM=file2;TO=
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983


This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
EOF FOUND IN FROMFILE AFTER RECORD 2

3 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

This is a simple file copy with no changes as you can see from the `ll` and `cat` listings. Note the wild card * that gives you all files starting with "file".

HP-UX:

```
$ cat >> file2
This is some more text to illustrate the
concatenation facility of UNIX in this game
of "Follow the Leader" with MPE.
[ctl-D]
```

MPE:

```
:cattt file2
FILE file2,OLD;ACC=APPEND
FCOPY FROM=;TO=*file2
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

This is some more text to illustrate the
concatenation facility of UNIX in this game
of "Follow the Leader" with MPE.
 < CONTROL Y >

3 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
:catf file2
FCOPY FROM=file2;TO=
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983


This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
This is some more text to illustrate the
concatenation facility of UNIX in this game
of "Follow the Leader" with MPE.
EOF FOUND IN FROMFILE AFTER RECORD 5

6 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

Here you see a concatenation with the append redirection instead of the replace. Control-D signals End of Data.

HP-UX:

```
$ cp file2 file3
$ cp file2 file3b
$ cp file2 file3c
$ ll file3*
-rw-rw-rw-   1 boles    101          247 Dec 25 20:55 file3
-rw-rw-rw-   1 boles    101          247 Dec 25 20:55 file3b
-rw-rw-rw-   1 boles    101          247 Dec 25 20:55 file3c
$ rm file3*
$ ll file3*
file3* not found
```

MPE:

```
:cp file2 file3
FCOPY FROM=file2;TO=file3;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

EOF FOUND IN FROMFILE AFTER RECORD 5

6 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
:cp file2 file3b
FCOPY FROM=file2;TO=file3b;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

EOF FOUND IN FROMFILE AFTER RECORD 5

6 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
:cp file2 file3c
FCOPY FROM=file2;TO=file3c;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

EOF FOUND IN FROMFILE AFTER RECORD 5

6 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
:ll file3@
LISTF file3@,2
ACCOUNT=  CAD         GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

FILE3             80B  FA           6       1023   1       128  1  8
FILE3B            80B  FA           6       1023   1       128  1  8
FILE3C            80B  FA           6       1023   1       128  1  8
:rm file3
PURGE file3
:rm file3b
PURGE file3b
:rm file3c
PURGE file3c
:ll file3@
LISTF file3@,2
NO FILES FOUND IN FILE-SET (CIWARN 431)
```

Note here the generic purge, representative of the UNIX™ respect for the programmer's presence of mind. (Some of us who only marginally deserve that respect do a lot more back-ups under UNIX™.) You can protect yourself on sensitive files by removing write permission and thereby getting UNIX™ to prompt for confirmation of purge.

HP-UX:

```
$ ll file1
-rw-rw-rw-   1 boles    101          121 Dec 24 14:01 file1
$ chmod 600 file1
-rw-------   1 boles    101          121 Dec 24 14:01 file1
```

MPE:

```
:ll file1
LISTF file1,2
ACCOUNT=  CAD          GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

FILE1              80B  FA           3       1023   1      128  1  8


:chmod600 file1
SECURE file1
```

Here's an example of changing access permissions on a file. We don't have a simple parallel in MPE. This now disallows group and public users to access the file. Note that the UNIX™ granularity of control could be approximated by a combination of the `secure` you see here and the `altgroup xxx; access=` facilities in MPE.

HP-UX:

```
$ cp file2 file3
$ ll file*
-rw-------   1 boles    101          121 Dec 24 14:01 file1
-rw-rw-rw-   1 boles    101          247 Dec 25 20:53 file2
-rw-rw-rw-   1 boles    101          247 Dec 25 21:06 file3
$ mv file3 file4
$ ll file*
-rw-------   1 boles    101          121 Dec 24 14:01 file1
-rw-rw-rw-   1 boles    101          247 Dec 25 20:53 file2
-rw-rw-rw-   1 boles    101          247 Dec 25 21:06 file4
```

MPE:

```
:cp file2 file3
FCOPY FROM=file2;TO=file3;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

EOF FOUND IN FROMFILE AFTER RECORD 5

6 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

```
:ll file@
LISTF file@,2
ACCOUNT=  CAD         GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

FILE1              80B  FA           3       1023   1      128  1  8
FILE2              80B  FA           6       1023   1      128  1  8
FILE3              80B  FA           6       1023   1      128  1  8


:mv file3 file4
RENAME file3,file4
:ll file@
LISTF file@,2
ACCOUNT=  CAD         GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

FILE1              80B  FA           3       1023   1      128  1  8
FILE2              80B  FA           6       1023   1      128  1  8
FILE4              80B  FA           6       1023   1      128  1  8
```

Here you see the ***move*** or rename facility in action.

HP-UX:

```
$ pwd
/users/boles
$ mkdir dir2
$ ll
total 15
drwxrwxrwx   1 boles    101          0 Dec 25 21:26 dir2
-rw-------   1 boles    101        121 Dec 24 14:01 file1
-rw-rw-rw-   1 boles    101        247 Dec 25 20:53 file2
-rw-rw-rw-   1 boles    101        247 Dec 25 21:06 file4
-rw-rw-rw-   1 boles    101       2536 Dec 25 21:28 seb
-rw-rw-rw-   1 boles    101       1055 Dec 24 14:07 sebe
-rw-rw-rw-   1 boles    101       1055 Dec 25 20:43 sebf
-rw-rw-rw-   1 boles    101       1607 Dec 25 20:54 sebg
-rw-rw-rw-   1 boles    101       2059 Dec 25 21:04 sebh
$ cd dir2
$ pwd
/users/boles/dir2
$ cp ../file1 file1dir2
$ ll
total 1
-rw-------   1 boles    101        121 Dec 25 21:35 file1dir2
$ cd
$ pwd
/users/boles
```

MPE:

Here we don't have an MPE analog closer than `hello` with a new group specified. Here while in the home directory, you create a new directory with `mkdir`. The directory file (initial "d" in the `ll` listing) now appears in its parent directory. A `cd` (**c**hange **d**irectory) to the subdirectory `dir2` is confirmed with a `pwd` showing the path name up the directory chain. The cp uses a `../` to indicate the parent directory of the current working directory. A `cd` without an explicit directory gets us back to the home directory, which the `pwd` confirms. This is just a quick dip in the deep end of the pool. Don't worry about this for the brief glimpse of UNIX™ you get here. But do be aware that the UNIX™ file system is different from MPE.

```
$ who > file4
$ cat file4
boles    console Dec 24 13:26
$ date >> file4
$ cat file4
boles    console Dec 24 13:26
Tue Dec 25 21:42:10 PST 1984
$ ll file4
-rw-rw-rw-   1 boles    101          59 Dec 25 21:42 file4
$ wc file4
      2     11     59 file4
$ grep 'MPE' file1
the UNIX vernacular to an MPE environment.
$ ll file1 > temp1; wc file1 > temp2; cat temp1 temp2 | grep 'file'
-rw-------   1 boles    101         121 Dec 24 14:01 file1
      5     22    121 file1
```

MPE:

Don't try to map this one-for-one to MPE. You see some redirection and a new counter command, `wc`, to set up an illustration of ***piping*** (the | operand) and the string finder, `grep`. The `wc` counts lines, words and characters. The `grep` lists the lines that contain the search string argument. Here the `cat` has 2 input files that it ***pipes*** to `grep` which outputs the two lines containing the search string 'file'.

Before wrapping up, let's scratch the surface of the UNIX™ ***shell***. First some simple shell scripts. Suppose you want your UNIX™ to speak MPE. Here you see a file called `listf` that contains `ls`. At first it won't execute but the `chmod` fixes that.

```
$ cat > listf
ls
[ctl-D]
$ listf
listf: cannot execute
$ chmod 777 listf
$ listf
dir2    file2 . . . temp1    temp3
file1   file4 . . . temp2
```

Here you see a file called `purge` with a `$1`, the symbol for the first argument, which is the name of the file you want to purge. The `chmod` makes it executable.

```
$ cat > purge
rm $1
[ctl-D]
$ chmod 777 purge
$ purge temp4
```

Next you see a shell control loop that edits all the files starting with "file", using the commands in `sebmod`.

```
$ cat sebsh
for i in file*
  do ed $i < sebmod
  done
```

Here's what `sebmod` looks like. It gives the file name, lists the first record, inserts a new line, then lists the first three lines, then quits.

```
$ cat sebmod
f
1
i
Begin with Apr 1985 HP3000 IUG Conference . . .
.
1,3p
q!
```

Here's an execution:

```
$ sebsh
107
file1
This is to show the translation of
Begin with Apr 1985 HP3000 IUG Conference . . .
This is to show the translation of
the UNIX vernacular to an MPE environment.

233
file2
This is to show the translation of
Begin with Apr 1985 HP3000 IUG Conference . . .
This is to show the translation of
the UNIX vernacular to an MPE environment.

59
file4
boles    console Dec 24 13:26
Begin with Apr 1985 HP3000 IUG Conference . . .
boles    console Dec 24 13:26
Tue Dec 25 21:42:10 PST 1984
```

There you have it: a glimpse thru the Looking Glass from MPE-land into the Land of UNIX™. You've seen our "MPENIX": some of the elementary functions in our quasi-UNIX™ UDC's that we built to help our SEED students. From there you sampled some of the UNIX power that enables a computer user to reach new levels of productivity. You have seen some of the features that have enabled UNIX™ to establish a good track record as the common link that lets us move with reasonable gracefulness across a substantial portion of the computer world today.

About the Author ....

Sam Boles *is a Member Technical Staff in the Productivity Center at the Hewlett-Packard computer facility in Cupertino, California. With HP since 1976, his computer experience started back in the AUTOCODER days of the 1401/1410, migrated thru the 360/370 era, and now focuses on networking HP productivity technology. Sam received his MS at UCLA in Information Systems.*

sebiug48 1010 28dec84

THE POOR MAN'S PERFORMANCE MEASUREMENT.

Rob van Dijk
Performing Systems, Montfort, The Netherlands

Summary

In the Contributed Library are quite a few entries which give us some information about the performance of our HP3000-system. The trouble is that the given information is often presented in such a way that it is difficult to interprete the given numbers or rates. Moreover, the introduction of MPE-VE has made most of these programs obsolete, because of the radical changes in the structure of the tables and their approach.
This paper tries to define what information is needed for a performance measurement and what tools are available to get at this information. It also states some of the restrictions of these tools, as far as this limits the usefulness of the given information.

It does not make the attenders performance specialists, but tries to make them aware of possible pitfalls in the use and evaluation of the given information.

The problem

There are a number of reasons why anybody would want to do a performance measurement:

- Long range capacity planning.
- Long response times at the terminal.
- Assessment of the load generated by a specific application.
- Assessment of the available capacity for adding new applications.

While all of these would require a different point of view on system performance, it is possible to define a superset of what information is needed. In a given situation it could very well be sufficient to get only a part of it, dependent on the specific requirements.
For this moment we therefore state our problem as:

- What part of the capacity of the system is in use
- Which processes cause which part of the load
- What is the relation between this load and the throughput of the system.

To get meaningfull answers to these questions we also should define the resources which possibly limit the capacity of our system:

- Type of the CPU
- Memory size and possible expansion
- Disc configuration and possible expansion
- Terminal configuration and possible expansion

but also:

- Available staff (EDP and End-users)
- Working hours
- Availability of personnel outside working hours

Each of these gives not only a limit to a partial capacity but also a unit to measure so that we are able to use quantities as well as rates to define the load of our system.

<u>What part of the capacity of the system is in use</u>

By answering this question we get an overall picture about the use of our system.
First of all is it necessary to obtain a detailed insight in the capacity that is available.

- Overall time that the system is available
- Working hours, pauses for lunch etc.
- CPU-type
- Available memory
- Disc configuration and from that the number of physical I/O's that can be expected

The measurement itself should of course fulfill the following conditions:

- The measured load should be typical for the normal system load.
- It should last long enough to exclude any random fluctuations.
- It should be detailed enough to show all significant fluctuations during the measured period.
- It should capture any structural load-variations during a meaningfull period.

We should measure the following items:

- Occupation of the CPU
- Memoryload
- I/O-load

<u>Occupation of the CPU</u>

The CPU-use that is generated in our system can be measured in CPU-seconds (Its speed is a constant). In the most ideal case we should become aware what part of the time the CPU is doing what.

- Busy on processes
- Busy for memory management (dispatching, memory allocating, disc caching)
- Busy on ICS
- Waiting for disc-I/O (User-I/O)
- Waiting for swap (Memory-I/O)
- Idle

Only the time that it is busy on processes is effectively used for user-oriented processing, but all other busy-states are directly connected to this processing and as such unavoidable. Idle-time is of course directly available for extra load, but also during the time in the Waiting-states isn't there any CPU-activity.

As long as our only goal is to know whether there is any free CPU-capacity, would it be sufficient to measure the Idle- and Wait-times together.
When it is necessary to see whether any optimisation is possible, then it would be necessary to separate the Idle-time from the Wait-time.
Separation of the Busy-states would be necessary if we would like to see how much time the CPU is busy on ICS. This could indicate high system-overhead because of terminal-I/O or type of CPU (Series III).

Tools in this area are:

- IDLE : gives rate of Idle + Wait states reports per hour.
- PORPOISE : gives exact distribution of CPU-states comes in BOEING library, not sure whether this is generally available.
- SURVEYOR : gives exact distribution of CPU-states. works through the Measurement Interface. reliability dependent on version.

Memory-load

As the HP3000 tends to fill its memory as far as possible, it is not correct to deduct the memory-load from the amount in use on a given moment.
Neither is the amount of virtual memory a reliable source of information, as this is determined by the MAXDATA-parameter, which can be substantially bigger than the actual segment-size.
We have to determine the load from the percentage of times that the Memory-Manager finds a segment (code or data) absent and has to swap it in from virtual memory.
An indication is also given by the amount of disc-I/O that is generated by the Memory-Manager.

The only tool in this area is:

- IOSTAT : gives distribution of I/O to all devices and total of Memory-Management I/O (MAM). not generally available.

I/O-load

Because of the restricted number of Disc-I/O's that are possible this is often the factor that limits the overall capacity of the system. Exactly how many I/O's are possible is dependent on

- Type CPU : The setup of a disc-I/O takes a varying amount of time on each CPU-family
- Number of GIC's : Each GIC is limited in the amount of data it can transfer.
- Number of Master-disc's : Each controller can handle only one request at the time
- Version of MPE : Which tricks are available:

Disc caching
Seek/Look-ahead
Rotational position sensing

- Locality of data: How far must the arm move
- Size of blocks : How much data is to be transferred

As you see, not all of these factors are hardware-dependent. Locality of data can be controlled by selective restoring data, while the blocksize is decided by your application.

Tools which count the amount of disc-I/O are widely available

- DISKIO
- FILERPT
- FILUSE
- JOBACCT (part of DREEACT-accounting system)

As these programs use the system logfiles, they can't obtain the I/O-rate, so their value in this environment is restricted. Programs that give I/O-rates are:

- SURVEYOR : gives I/O-rate per device
- IOSTAT : gives I/O's per device and Memory-Management I/O in a specified interval. Can select per PIN, device, Type(Read, Write) not generally available.

<u>Which processes cause which part of the load</u>

The optimal answer to this question would give us the distribution of the systemload for each process on each moment. Only then could an exact answer be given as to which process is the most critical to our system, this being processes which cause a given resource to be in use near its capacity. Because of the fast altering of the list of these processes, this will cause quite an amount of overhead by the measurement, a small measuring interval being necessary. In many cases this is not acceptable, the main reason for this kind of actions being an allready overloaded system. Because of this a less accurate way of measurement is generally used. This gives us the amount of use of each system-resource (CPU-load, Memory-load and I/O-load) for each process without the direct connection with the level of activity systemwide (Or only partially), in most cases also at bigger measuring intervals. This will cause inaccuracies, but they will have the largest influence on the short-lived processes, which in their turn have only a small influence on the overall system-performance.
At a minimum a measurement should produce the following figures:

- CPU-seconds per process
- Maximum memory-load
- I/O per process

For a more detailed look at the working of each process it would be nice if these figures would get supplemented with:

- Maximum memory-load differentiated for code, stack and each XDS
- I/O per file per process
- Number and size of terminal-I/O's

- Disc-I/O per terminal-I/O

If at a later stage the relations between the processes on the system should be examined, then the next items would also be necessary:

- Time waiting for locking
- Time between terminal-I/O's

We should keep in mind that a process which uses a lot of system-resources does not necessarily be a bad program. This naturally depends on the functions that are to be executed. On the other hand such a process is naturally first choice for optimisation.
It does also not mean that a process which doesn't use much resources is therefore a fast-running program. Waiting-times for locking or such could very well ruin the response-times of these processes.

It is in this area that our tools lack. Available are the SOO-family the logfile analyzers and SURVEYOR.
What can they offer:

- SOO-family : This type of program monitors the PCB-table to obtain information on CPU-load and Memory-load of the current processes. We can find the accumulated CPU-time and Memory-load differentiated to code, stack and the total of XDS-space. This comes from the swap-table and gives only the situation at sampling time.
- Using the system-logfiles could we get information about the CPU- and Memory-load of jobs and sessions. The easiest way to this info is using one of the accounting packages as for example DREEACTG. These give total amount of CPU, number of swapped sectors, number of disc-I/O's and number of terminal I/O's. The problem is that these figures are not on process-level, but on job/session level.
  Also can we distill information on file-use on a systemwide base (Top-100) or per job/session.
  for example by using:
    - DISKIO
    - FILERPT

  or some program like these.
- SURVEYOR : This program gives approximately the same information as the SOO-family, but it uses the Measurement Interface. Its reliability depends on the version.

<u>What is the relation between this load and the system-throughput</u>

This is the most difficult question of all three, because it can only get answered with a wide knowledge of the application.

The relation does not have to exist. If there is no overload on the system, the only relation could be interaction of specific processes, which could be different programs, but also more occurrences of the same. The point to consider is that such interaction between processes is mostly through files. Not only file locking could drastically reduce the throughput of the system, also the mechanism that is used by IMAGE to serialize the I/O to a database could very well be a limiting factor.

If some systemresources are loaded in excess of their capacity, then another interaction arises. Queuing for disc-I/O could very easily multiply the response-time by factors. Also Memory shortage can have devastating effects, far larger than directly shows from the figures.

Apart from common sense, a wide knowledge of the application and of the HP file-access methods there are no tools for this kind of job. It would be nice to have some of the figures mentioned in the previous section available though.

It seems to me that here is a lack in the Contibuted Library that is asking to get filled by one of these marvellous men that know so much about our HP3000.

Biography

Rob van Dijk
has worked with HP3000-computers since 1979 at one of the biggest Dutch softwarehousès. He is now working as an independent consultant for system management, system performance and data communications.

PERFORMANCE SELF-ANALYSIS

Brian Duncombe

CAROLIAN SYSTEMS INC.

Toronto, Ontario, Canada

INTRODUCTION

This paper deals with the subject of system performance - what it is, how we can measure it and what we can do to improve it.

As a consultant specializing in performance of the HP3000 computer system, I have had experience looking at a wide variety of system configurations and application loads. From this perspective, I have come to the conclusion that much of the written material in this area is too analytical in nature and not enough of the focus is on the practical things that anyone can do.

In this paper, I try to provide a technique for analysing any system. The treatment is that of a common sense approach rather than an in-depth technical discussion and as such should be applicable to both the technical as well as the management-oriented user. The material presented requires neither great knowledge of MPE internals nor of the details of how to go about monitoring them. It shows that the subject is a simple example of supply and demand.

WHAT IS GOOD PERFORMANCE ?

In order to discuss a subject, we must first agree on its definition. One definition of performance is "What you want, when you expect it." Most of us could agree with this but we could probably discuss the CAUSE and CURE of the problem forever. In discussing the subject, many approaches and ideas should come to light but the most important thing that should emerge is a firm understanding of the variables of performance and a general idea of how they interact.

WHAT CAUSES DELAY?

GETTING THE COMPUTERS ATTENTION (priority)

With a multi-user system such as the HP3000, many users are often competing for the available processing power. Since there is really only one central processor, each user must compete with every other user for this sometimes scarce resource. In order to provide a reasonably equitable sharing of the CPU resource, MPE uses a system of priorities. These are adjusted according to a set of parameters specified by the system manager with the TUNE command and are dependent upon the scheduling queue that a particular user has been assigned to. In this manner, competing users are favoured or penalized depending on the queue that they are running in (BS, CS, DS or ES) as well as how much CPU resource they have consumed. By applying these rules, MPE may greatly favour one user over another so that even though one user is getting quite good response times, another user is being almost totally ignored. Of course the total resource available will also delay some or all users if the demand exceeds the supply.

PROCESSING INSTRUCTIONS (central processor speed)

Even when a particular user is able to get the attention of the central processor, the instructions that form the program that is helping him solve his particular task require some finite time to execute. We all hear the claims of computer manufacturers regarding the thousands or millions of instructions that their processor can execute in a second of time (MIPS - millions of instructions per second). A sobering fact is that what appears to be a simple task to a user at a terminal can actually require millions of instructions to be executed. The traditional HP3000 application is usually considered to be low on CPU resource and high on disc I/O requirements but many tasks are processor intensive. In addition, when many users are sharing the system, their relatively small individual requirements can quickly add up.

GETTING MAIN MEMORY SPACE (memory management)

Just as all users must compete for the CPU, all users must also compete for the available main memory resource. Operating systems such as MPE recognize that main memory can be used more efficiently if the main memory size is made to appear very much larger by a system of allocating the less-used memory areas to disc memory (virtual memory management). The theory here is that at any point in time, only a small portion of users are really doing anything and therefore requiring real main memory resource space. This mapping is of course made almost invisible to the various users of the system and it is only when the demands for <u>actively</u> used memory space exceed the available real memory that the scheme breaks down due to the very much slower access time associated with disc storage. When this happens, the delays associated with getting data from and putting data to the virtual memory appear as sluggish performance. At this point, users that make efficient use of main memory and follow some simple guidelines may very well compete unfairly with the other users of the system. They may not suffer noticable delays even when the majority of the system community does. This should point out the possibility that if more users followed the guidelines, they and possibly the whole community might gain similar improvements.

WAITING FOR I/O TO COMPLETE (disc & terminal speeds)

HP3000 applications are normally characterized as I/O intensive. By this, we mean that the applications access large volumes of data while requiring a relatively low volume of processing of the data. The most popular devices involved in data transfer are terminals and disc drives.

With terminals, data can be transmitted at rates of up to several thousand characters per second. While this might seem adequate, many applications use block mode and large formatted screens containing several thousands of characters of data to be transmitted. What this means is that a user might very likely grow impatient during the second or so that the data takes to move between the terminal and the computer. Depending on the nature of the user and his expectations, he may see the transmission time as part of the overall response delay or he might consider that once the data begins to move, he is seeing progress and is therefore satisfied. How the user views this transmission time can have a great effect on perceived response. Of course, slowing down the data flow with modems and sharing the transmission capacity by using line-sharing (multiplexing) can worsen this perception.

Disc I/O is often the single largest bottleneck on the HP3000 computer system. The data transmission rate between disc and the computer is in the range of one million characters per second. In addition, some hardware allows simultaneous transmission of data between the computer and two or more disc drives. What slows down this whole process is the mechanics of disc accessing. The physics of the matter is that data is stored on circular tracks on a spinning disc and depending upon where the platter is in its rotation, it can take a significant time to wait for the media to rotate into position for transmission. The construction of most disc drives also provides for moving heads to allow more than one circle of data to be stored on concentric rings of the disc platter. The requirement for head movement also slows the process down. The sad fact is that the average transmission is about 500 bytes of data and the transmission time for this is insignificant when compared to the rotation delay (latency) and head movement time (seek). The net result is that a single disc drive can average only about 25 physical accesses per second.

Viewed in the light of data transfer limitations, the typical HP3000 application can present a real problem to overall system performance.

INDICATORS OF PERFORMANCE

Based on the definition of performance presented earlier, the signs that indicate relative performance to us are elementary. Since it all reduces to users getting what they want when they expect it, the meters of performance are time measurements and expectations.

RESPONSE TIME AT THE TERMINALS

Terminal reponse time is usually what an end-user is looking at in attempting to measure performance. As discussed previously, there is some room for interpretation as to what components are actually part of the response delay. Generally response delay is measured from the time the user presses the return/enter key until the first character of data is returned to the terminal. This measure is usually adequate for non-key-entry users of a system but for heads down data entry, the measure is usually considered to be from the time the return/entry key is pressed until all data has been returned to the screen and a new transaction can be begun. On a system such as the HP3000, this terminal response can vary dramatically during the day depending on what else is happening on the system. For this reason, it can sometimes be frustrating to investigate a reported slow response and get there just as the system suddenly picks up speed again.

RUN TIME OF BATCH JOBS

As far as the operations personnel of a computer centre are concerned, the time it takes to run the various batch jobs that must be run is the real measure of performance. It is one thing to have some interactive users whining about response times but when the batch processing components of a system start to run behind, things become very hectic. In the event that nightly batch runs are not able to be completed by the next morning, the system is usually not available to the interactive users until later in the day. The ultimate in poor performance is not being able to access the system at all. Unfortunately, batch processing performance problems seem to creep up on you slowly until all of a sudden you are faced with a severe problem.

FREQUENCY OF PHONE CALLS

An excellent early indicator of deteriorating system performance is an increase in the phone calls that operations receives. These usually start out with the less popular users calling to mention that things are slow and the situation worsens until users start to call to ask if the system has crashed. By developing a habit of "managing by wandering around" the computer room area, you can often tune in on the user community feelings about system performance.

APPROACH TO IMPROVEMENT

THERE WILL ALWAYS BE A BOTTLENECK

No matter what system you look at, there is always a bottleneck that is limiting the performance of the system. No matter how many bottlenecks you relieve, another one will appear to take its place. Realizing this, you can approach the subject as a never-ending one. It is up to you to determine how far you will go to "solve" the performance problems of your system. At some point, the gain will not warrant the effort involved and at this point you should stop for a while. The definition of this point is a subjective one that only you can determine.

APPLY EFFORTS TO AREAS OF MAXIMUM RETURN

Since the resource involved in the current bottleneck is the one that will yield performance improvement when the demand for it is reduced, it makes sense to work on this as the primary target. If performance is currently limited by the processor speed, reducing demand for main memory will have little effect on improving performance. Even within this scope, there will be several areas that you can work on to reduce demand for the critical resource. There will always be areas for which a relatively small effort will yield large results while other areas will exist for which it seems that no amount of work will yield a noticeable result. If you apply some common sense to choosing the area to work on, you should be able to maximize the return on your effort. Clearly, it would be better to tune a program that accounts for 2 percent of all run time on the system than to chose a program that is only run at month end.

KEEP IT SIMPLE

Naturally, you will usually find that there are more areas to improve than there are resources to work on them. In my experience, the techniques that have consistently yielded the best long term benefits have been those that involved a <u>simple</u> change in approach to the data processing problem. As the tuning becomes more and more "elegant", the possibility of introducing a new problem or complicating future maintenance efforts rises dramatically.

ENCOURAGE GOOD HABITS

Most tuning effort is focused on the study of existing systems to see how they can be improved. This is a never-ending task but one that is unavoidable. If in addition to this, you begin a campaign to develop the use of more efficient techniques in the development effort, you will get it right the first time on new system development and reduce the effort required to tune it in the future. By doing this, you not only get helpers working for you in your tuning effort but also you will break the circle of generating new systems to be tuned at a later date. By instilling in everyone the habits of doing it the right way, you have not increased their workload but you have gone a long way towards reducing the number of obvious contributors to poor performance.

YOU ARE NEVER FINISHED

As I mentioned earlier, there will always be a bottleneck to be removed. At some point, you must at least rest from this tuning effort. Only you can define the point at which additonal effort is not worthwhile. Even when you feel you have reached this level, you should plan to periodically review the situation to see if you have slipped below this line. The line itself will usually move depending on whether you have more processing resources than you really need (or more dollars to spend!) or whether additional applications have been pushing your resources to the point where you must do something to carry you over until more processing resources can be acquired. Remember, it is much less painful to spend some time for periodic review than to be suddenly faced with a serious problem that appears from nowhere.

INDICATORS OF A PROCESSOR BOTTLENECK

CURRENT INSTRUCTION REGISTER / ACTIVITY LIGHT

All of the HP3000 central processors provide a visible indication of CPU acivity. In the case of the Series II/III/64/68 processors, there is a 16 LED display that normally shows the bit pattern for the machine instruction currently being executed. If this instruction is not changing very rapidly or there is a visible fixed pattern to the display, the system is not busy all of the time. If the pattern is constantly bright and changing, then the processor is probably running at or near capacity. The current instruction register (where available) is the first thing that I look at upon entering a computer room.

With the Series 39/40/42/44/48 processors, there is no current instruction register display. If you open up the front panel of the processor, you will find an activity light which is a single LED that flashes as the processor is executing instructions. While you can't get as good an indication of what is going on, you can easily spot a very busy or very idle system.

BATCH JOBS GETTING NOWHERE

If your installation regularly runs batch jobs during the daytime, the ability of these jobs to make progress is a good gauge of the reserve capacity of your CPU. Since batch jobs will by default reside in the DS subqueue, their priority will be less than that of all interactive processes. If the CPU becomes saturated, the dispatcher function of MPE will totally ignore these less important batch jobs and they will make little or no progress. If you see this happening at particular periods of the day, it is a very good indication that the CPU resource is saturated.

INDICATORS OF A DISC BOTTLENECK

DISC ACCESS LIGHTS

With the exception of the HP7911/12/14 disc drives, all of the disc drives contain an LED that shows when the disc is physically being accessed. By observing this light on each of the drives, you can not only see how busy the various disc drives are but also how evenly the disc accessing activity is spread across the available disc drives (assuming you have more than one). This is one of the first things a seasoned performance specialist will look at upon entering the computer room.

FALLING PACK COVERS / TAPE SEALS

Many users store the empty pack covers for the various disc packs in use on the top of the disc drive that the pack is currently in. In addition, many people place tape seals on top of disc drives while the tape is mounted on the tape drive. If your operations people are frequently retrieving pack covers and tape seals from the floor then there is a good chance that the disc drive is shaking as it is being accessed (a hand placed on top of the disc drive would give you a clue also!). This shaking is caused by the movement of the arm-access mechanism as the read/write heads are rapidly moved across the surface of the disc. Since this seek movement is a major contributor to slow disc access times, the vibrations might point to a situation where two files on the same disc are being accessed as a set. If one of these files could be moved to a drive with less activity, there is a real potential for disc subsystem performance improvement.

DRIVES THAT MOVE

When the vibration of the arm-access mechanism movement becomes very pronounced, it is common for the disc drive to actually move slightly on the floor. If the feet of the drive are not screwed down to take the weight off the casters, this movement can happen quite easily. If you are experiencing disc drives that "ramble", you should first check to see that the feet are screwed down and if this is the case, you have a very severe disc subsystem access problem. If the seek activity has caused this much vibration, it is undoubtedly also causing slow data access on the drive where the vibration is happening.

TIME TO TEXT IN A GIVEN FILE

A very simple test to measure the capacity of the disc subsystem to handle access requests is the time it takes to TEXT a file into the editor. If you build (or choose) a file containing 50 or 100 records and measure the time it takes from when you hit ENTER for the text command until you get the next editor prompt, you have a simple test. By recording the time when you are the only one on the system, you get the optimum value for your system. When you repeat the test during normal operations, the resulting time when compared to your stand-alone timing gives you an indication of disc subsystem performance. This test can be extended by creating an identical file on each disc drive and repeating the test on each drive.

EXCESS ACTIVITY ON VIRTUAL MEMORY DRIVE(S)

When the system is having trouble keeping the required segments in main memory for all active users of the system, it begins to swap segments for one user out in order to make room for the next user even though the segments will be needed very soon. The state of rapidly swapping segments in this manner is referred to as THRASHING. Once this state occurs, performance of the system deteriorates very rapidly since you are replacing main memory access times with the relatively slow disc access times. If virtual memory is configured on only some of your disc drives, the excess activity on these drives can signal a main memory shortage.

VIOLENT VIBRATION OF VIRTUAL MEMORY DRIVES

Even when the accessing activity is fairly well balanced on the drives, if the drives containing virtual memory are shaking more than the others, you may have main memory shortages. When you start your system and load it initially, disc space is allocated first to system overhead including the directory and then to virtual memory space. After this space is allocated, your own data files are positioned on the disc media. This results in the directory and virtual memory being positioned on the edge of the disc and your data spread across the surface of the platters. When excessive directory or virtual memory accessing begins, the read/write heads must be continually moved from your data files to the directory/virtual memory and back again. This usually results in much longer than average seek distances and more pronounced shaking of the disc devices.

VERY SLOW RESPONSE IN THE EDITOR

A simple test that can highlight main memory shortages involves making simple changes to a few lines in a text file using the editor. To do this, text in a file and then modify several lines at least 50 lines apart. This will require very little CPU resource and only one or two disc accesses. If a noticable delay occurs between the two changes, the cause is likely to be that between the two activities much of your working set was swapped out to disc and the delay was in making it present again.

THE BIG 10 PERFORMANCE KILLERS

#1 - EXCESSIVE FILE OPENS AND CLOSES

Perhaps the most common cause of poor performance is the situation where programs are opening and closing files frequently during their execution. The overhead of this activity is very heavy and for much of the process it requires exclusive access to system resources such as the directory which locks other users out. Many programs exist in which file opens and closes are part of a commonly used subroutine or procedure and it is not apparent that this activity is going on.

#2 - INSUFFICIENT IMAGE BUFFERS

IMAGE/3000 by default assigns a number of buffers to be shared by all users of a database. By default this number varies according to the number of users who currently have the database open. In the first place, the defaults are usually not high enough and IMAGE performance suffers because of this. In the second place, as the number of buffers varies, the data segment containing the buffers will require a change in size which often involves swapping it out and back in again. My recommendation is to specify a fixed number of buffers for all numbers of users. This can be done by using DBUTIL and specifying "SET basename BUFFSPECS=32(1/120)". This suggestion should be tempered by the number of databases, the amount of available real main memory and the number of users sharing the database.

#3 - NOT BLOCKING SERIALLY ACCESSED FILES

A very common habit on the HP3000 is to allow the file system to assign the default block size to a file. In reality, the file system does not do a particularly good job of this and you should always specify it explicitly. A good rule of thumb is to block all files so that the block size is about 4K bytes. By doing this, you gain a large blocking factor and at the same time do not create buffer segments that are unusually large and therefore difficult for the memory manager to keep in main memory.

#4 - NOT ALLOCATING COMMONLY USED PROGRAMS

One of the most disruptive tasks that the operating system is required to perform is that of making a program file ready to be executed. The many facilities of MPE that make code sharing so convenient also extract a high toll at the time that you RUN the program. The work required to setup the segments and resolve external subprogram linkages is a monumental task and its affect on the system is magnified by the requirement to "own" many of the MPE system resources during the effort. By ALLOCATING the programs that are often run but seldom shared concurrently, you can drastically reduce both the delay to the user trying to invoke the program as well as the other people trying to get work done on the system.

#5 - EXCESSIVE USE OF UDCs

The use of UDCs is very much a mixed blessing. When used in moderation, they provide a great convenience that makes the system much easier to use. Many installations, however, get carried away with this facility and implement their own operating system interface. In addition, many of the commands defined for the benefit of development staff find their way into the end-user UDCs. Whenever a command is issued, it must first be searched for in all active UDCs for the user. If it is not found there then it is passed to the system command processor. By having a long string of UDCs, you make this scanning more complex than necessary. In addition, many shops develop a standard of a system wide UDC file, an account UDC file and possibly a user UDC file. For each file, an MPE file is kept open for the life of the job/session. This is made worse by naming more than one file at a particular level.

#6 - NOT USING "OUTPUT DEFFERRED" WHERE POSSIBLE

This capability of IMAGE/3000 is not very well publicized. What it does is turn off the facility of IMAGE that always keeps disc files current with what is in the IMAGE buffers in main memory. By selecting deferred output, you tell IMAGE that disc writes should only be done when absolutely necessary. By doing this, you can decrease the disc accessing required for a given task by a factor of 2 in many cases. There is a cost for this, though. In the first place, you must be the only accessor of the database (open mode=3). In the second place, you should be prepared to recover and rebuild your database since in the instance of a program or system failure, the data base is guaranteed to be not only logically incorrect but also structurally unsound. For these reasons, the technique is usually reserved for batch processing when you have a good backup and can save all transaction input.

#7 - EXCESSIVE LOGONs

The hardest task that can be requested of MPE is to logon a new user. The process makes exclusive use of many of the system tables and resources thereby preventing others from using them for much of the duration of the logon. While people must log on in order to use the system, it is not at all uncommon to see a user logging on to one group or account and then off and onto another repeatedly. When you investigate the reason for this, you often find that by changing the security on some files or accounts you could eliminate this logon activity or at least reduce it substantially.

#8 - BACKUP

With a few exceptions, system backup is a function for which the system must be used exclusively. The time that it takes is lost time as far as the users of the system are concerned. Even when the system is backed up late at night when no users desire to use it, if batch processing is to be done it will be held up. Any steps that can narrow the window during which the system is off the air for backup will tend to improve perceived system performance.

#9 - DATABASE DESIGN

The design of the databases that are used by applications can have a serious impact on the overall performance of the system. When chained paths are not provided for commonly utilized searches, the resulting serial scans of the database place an onerous load on the system resources. The pitfalls of database design are a topic unto themselves and you would be well advised to spend time doing a periodic checkup of your data base designs and their usage.

#10 - GETTING ON THE WRONG SIDE OF THE USERS

As our definition of performance points out, the topic is very subjective. If a user is unhappy with the services provided by the computer system, there is probably nothing that can be done to eliminate the comments directed at the system performance. Cases where the user expected a 'keypunch like' response and was delivered ENTRY/3000 to batch files is usually a no-win situation. By setting expectations up front and making a long term commitment to work at user satisfaction, this situation can be largely avoided.

SOFTWARE MONITORS

While there are a great many tools available as part of MPE and many performance measuring techniques that can be used to evaluate your system's performance, there comes a time when you want more. The most obvious tools that come to mind are the so-called software monitors. These tools are programs that have been designed to sample data relevant to some aspect of system performance and report it to you in some fashion. These programs fall into two categories that are differentiated most obviously by their cost. In most instances, increased cost includes support, better documentation and regular enhancements.

CONTRIBUTED

Many programs are available on a contributed basis from either the IUG contributed library tape or the HP TELESUP account. Some of the more popular ones are listed below.

SOO This program exists in many forms (SOO4, MOO, GOO, etc.). In general, the program scans the various processes active on the system and reports on CPU usage for each.

TUNER This program reports on the configured size of many of the system tables and in addition shows the current and maximum utilizations.

LOGREPT This program actually consists of several programs that analyse the file close entries in MPE log files and reports on the files that have been accessed with some statistics.

SURVEYOR This program monitors the system using the MPE measurement interface and reports various statistics concerned mainly with process level data. In addition, it reports much of the data that TUNER does.

OPT/3000 (Hewlett-Packard) This is the tool that all the others are traditionally compared to. It provides a great deal of data about both overall system data as well as process level data. It includes many different display screens designed to highlight specific areas of performance concern.

APS/3000 (Hewlett-Packard) This tool is aimed at analysing use of the CPU from the viewpoint of the code segment containing the instructions. It does this by using a statistical sampling technique to investigate the code in use at any point in time. It is useful for determining what code is the most frequently used and therefore the best candidate for tuning efforts.

TINGLER (TEXET) This tool is an excellent tool for analysing where a program is spending its time. On the surface it appears to serve the same function as APS/3000. It uses, however, a technique of modifying a temporary copy of the program to be studied so that it can trap all calls. In practice it is probably the best tool for the serious investigation of code utilization and resegmentation.

LOOK/3000 (Tymlabs) This tool is aimed at the COBOL user. It attempts to provide a tool similar to APS/3000 and TINGLER. It provides a very good feedback facility between the code being executed and the source program and makes it easy to highlight the busy code in a program.

SYSVIEW (Carolian) This program provides facilities very much like OPT/3000. It is oriented towards displays that are aimed at studying various aspects of system performance. It has been developed from the performance specialist viewpoint to make the displays as convenient as possible and to highlight relevant data easily. Its ASYST function allows automatic system monitoring and reports any deviations from user-set permissable ranges.

The Twilight Zone ....Between MPE Capabilities.

By Jelle Grim,

Holland House, AALST, Holland.

Abstract.

Within the MPE Operating System a number of so-called Capabilities are recognized which define the actions a user (or a program) can perform on the HP3000 system. These capabilities can be used to implement and maintain the security and the integrity of the system. The most critical capabilities are SM (System Manager), OP (Operator/Supervisor) and PM (Privileged Mode).

The traditional HP3000 systems always had assigned to them a System Manager who, by means of his so-called Capability Set, had access to all resources of the system. However, with the introduction of the lower-cost HP3000 systems and the ongoing hardware decentralization, it will not always be desirable and/or feasible to let a full-time user of a decentralized system have all the capabilities. In other words : These systems will not have a full-time system manager and/or console operator.

This paper will describe a method, that can be used to assign capabilities to certain users dynamically instead of statically, i.e. only those capabilities and only for so long as is necessary to perform a specific task. Needless to say that this must be accomplished while retaining security and including all required measures to avoid undesirable adventures. As the "carrier" of this method a menu processor can be used, whilst also some intrinsic substitution will have to take place. These items will be discussed also.

Contents.

1. Introduction.
2. Design Considerations.
3. DCAPS, the Dynamic Capability Switch.
4. RESCOM, the Restrictive Command Intrinsic.
5. MENUPROC, the Alternative Command Interpreter.
6. Tying it All Together.

1. Introduction.

This presentation will deal with the requirements for a more dynamic definition of the so-called "capabilities", that are recognized within the MPE operating system of the HP3000 series of computers. Currently these capabilities are supported in a (too) static way. If we go back in the history of the HP3000 series of computers, we will find the cause of this situation.

When the first types of computers (the CX, the II, III, 44 and 64) of the HP3000 series were introduced and sold, the investment for a complete installation normally exceeded $150.000. The machines were used in a data processing environment, whereby a number of special functions were recognized. In connection with the HP3000 machines the function of System Manager, Operator and often also Account Manager were special and mostly required some extra training or the hiring of specialized personnel. In comparison with the investments in hardware and system software these specializations were justified.

The MPE operating system also recognizes these special functions by defining special capabilities that were assigned to these functions, enabling them to perform their specific, specialized tasks. The capabilities referred here are SM, OP and AM. Within the MPE security these capabilities give the user a number of special commands, like ALTACCT (SM only), ALLOCATE (OP only) and ALTUSER (AM or SM), and widen the scope of a number of other commands, like REPORT (especially SM and AM) and STORE (especially OP).

Lately Hewlett-Packard has introduced more and more low cost HP3000 systems, like the 39 and recently the 37. Currently is is possible to acquire a complete HP3000 system for less than $30.000. Now the picture changes, because it is no longer feasible to appoint expensive System Managers and/or operators full time for these small installations. In the best case, if these small installations are part of a multi-machine user, a specialist of the central DP department will dedicate some time to these mini-mini's now and then, thereby acting as a part-time or ad hoc System Manager.

However, some tasks involving special capabilities have to be done on a regular basis or have to be done on an ad hoc basis, when a specialist is not available. This can be solved by assigning one or more of these special capabilities to one or more of the regular users of these little systems. Now here's the catch. The special capabilities OP and even to a greater extend SM give the user, who has these capabilities assigned to him, almost unlimited power within the MPE operating system and its security. Obviously this situation is not very nice, either because of educational reasons (playing system manager or operator just requires some specialist knowledge), or because of security reasons (often it is not advisable to let a "regular" user have access to all information stored within a machine).

Currently there is no "in-between", i.c. there is no such thing as a sort-of System Manager or a partly Operator or a pseudo Account Manager, in other words :

There is no twilight zone between MPE capabilities.

Thinking of this problem, it seems that a lot of things will be solved, if we would have a mechanism for dynamic capability switching. Dynamic Capability Switching, in this context, would mean, that a "normal" user gets a special capability (OP, SM, AM, etc.) assigned to him for so long a time as he needs it to perform a certain task. After that task is finished, he has to return to the status of "normal" user again. Furthermore, once we would have this mechanism we could also use it for the following, additional operations that we have been thinking about, but that were impossible to implement under the standard MPE operating system :

- A user with temporary AM capabilities could be allowed to change his own password, to perform some file management within his account, BUT he must NOT be allowed to perform a PURGEUSER or PURGEGROUP command.

- A user with temporary OP capabilities could be allowed to use commands like ALLOCATE and SWITCHLOG, to alter the JOBPRI to be able to submit a job to the CS queue, BUT he must NOT be allowed to perform a STORE command for files outside the logon group.

And, of course, once the problem of dynamic capability switching is licked, it will also be applicable to other capabilities like SF (save files), PH (process handling), DS (extra data segments) and PM (Privileged Mode !!!).

Although it seems that the only thing, that is needed now, is some sort of capability switching utility, there is more to this alteration of the MPE capability principle than meets the eye.

Therefore, this paper will elaborate on a number of design considerations with regard to the Dynamic Capability Switch, with rough designs of the components that, together, will form a complete dynamic capability switching mechanism. The concluding chapter will describe a way to integrate all components as described above into one unit, whereby an applications manager will be the end result.

2. Design Considerations.

In this chapter we will try our hand on describing a number of features and other assorted items with regard to the design of a dynamic capability switching system. As with most utility-type software, it is possible to turn out a quick-and-dirty routine, that performs the capability switching, without looking at the implications on system security, system integrity, etc.

However, thinking about the potential of a capability switching utility, everyone will agree that at least some caution will have to be applied when designing, building and using this software. This paper will not try to present an all watertight design for the capability switch, but it will merely point out a number of pitfalls, temptations and dangers, that have to be avoided at all costs.

In the following pages a number of items with regard to the design of the dynamic capability switch will be discussed.

Stand-alone or Routine.

The choice that has to be made here is between supporting the capability switch as a stand-alone program and supporting the switch as a routine/procedure. Defining it as a program enables the usage of the switch from within a UDC. On the other hand, when the capability switch is designed as a routine, it can also be used from within programs.
If the switch must be used from a menu processor either method will suffice, because most menu processors support either the execution of programs, or the execution of external routines or both.

Simple Operation.

It seems best to keep the capability switch as simple as possible. The basic function of the capability switch must consist of :

- the ability to switch one or more MPE capabilities on,
- the ability to switch one or more MPE capabilities off,
- the ability to report on the current setting of the MPE capabilities, and
- any combination of the above.

The actual capability switching will be done on the MPE Job Information Table (JIT). No additional functions should be incorporated in the switch.
The specification of capabilities that have to be switched ON or OFF and/or the command needed to verify the settings can be passed to the program or routine using a character string.

Usage of Privileged Mode.

It is obvious, that it is necessary for DCAPS, to be able to perform its feats, to use the much feared Privileged Mode capability.
Although the PM usage for this purpose is no direct danger to the MPE integrity, indirectly breaches of integrity/security can occur when the real hackers enter the game.

The most stringent security measures must therefore be observed when determining the location of the program, determining who is going to use the program and how the program will be used.

Access Security / Location.

As the capability switching utility can be a dangerous weapon in the hands of an end-user, or worse, in the hands of an evil minded end-user a number of precautions must be taken to ensure, that the capability switch can and will only be used in a controlled environment. It doesn't do to leave the possibility open for just anybody to say "please run DCAPS and give me all capabilities".

Therefore both the location and the access of the software must be controlled. If the switch will be developed as a program, the best place for it to reside is within a group with the PM capability within the SYS account, like PRIV.SYS or UTIL.SYS. Furthermore, the program should be protected by a lockword.
If the switch is developed as a routine in an SL, then this SL must reside in the same group as the program that calls it, for instance a menu processor. It could be placed in the system SL, but that would give too easy an access.

Restricted Usage.

When a user has certain special capabilities assigned to him it is no good, to tempt him by letting him have unlimited access to the MPE operating system. As discussed in the Introduction, special capabilities will allow the user to use certain special MPE commands and/or widen the scope of certain other MPE commands. It is therefore mandatory to use the switch always in combination with some kind of restrictive user interface like User Defined Commands (logon UDC's) or a menu processor like HPMENU, HELLO-3000 or UNIMAN, making it impossible for the user to abuse the power vested in him by the capability switch.

The Danger of the MPE COMMAND Intrinsic.

Even if a restrictive user interface is used, the structure of the MPE subsystems sometimes offer the possibility to access MPE commands in a more indirect way. Subsystems like SPOOK, FCOPY, EDITOR and also user designed software can offer access to the MPE commands via the so-called COMMAND Intrinsic. If the user has some additional capabilities assigned to him by the capability switch at that moment, he can use the MPE commands belonging to that capability.

A second security enforcer must therefore be implemented in the form of a restrictive command intrinsic, i.c. a custom made command intrinsic, that captures all calls to the "official" MPE COMMAND intrinsic and that only transmits commands, that cannot be used for less legal purposes, to the MPE COMMAND intrinsic. Again, to make the complete capability switching system work, a restrictive user interface is necessary.

The design considerations as discussed above show, that it is not good practice to just write a little PM program, that will perform your capability switching. On the contrary, after some thought anyone will agree that, to support a real dynamic capability switching system, the following components are necessary and should always be user together :

- The actual Dynamic Capability Switch, or DCAPS.

- The Restrictive Command Intrinsic, or RESCOM.

- The Restrictive User Interface, or MENUPROC.

The following chapters will deal with each of these components in detail.

3. DCAPS, the Dynamic Capability Switch.

To get the logical placement of DCAPS into perspective, the diagram below shows how it can help a "special" user to access the MPE special capabilities.

Three types of users will be recognized :

- The standard user, who has normal access to all standard capabilities of the MPE operating system.
- The special user, who is just a normal user that now and then gets one or more special capabilities to perform a certain task.
- The system management that normally has all capabilities at its disposal.

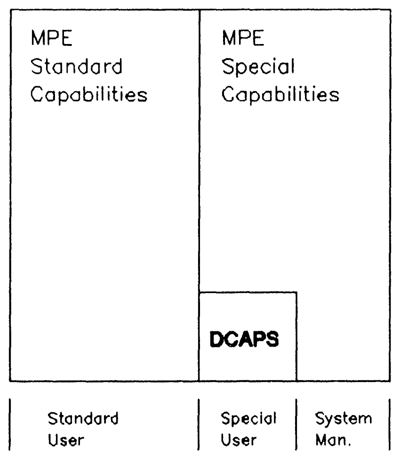


Figure 1. DCAPS, the Dynamic Capability Switch.

The Dynamic Capability Switch or DCAPS must be capable to perform at least the following activities :

- Switch one or more of the MPE capabilities ON.
- Switch one or more of the MPE capabilities OFF.
- Report on the current settings of the MPE capabilities (ON or OFF).

Some observations can be made. If DCAPS is envisaged as a stand-alone program, the actual capability switching will have to be performed on the MPE Job Information Table (JIT), which means, that the capability switch will be valid until it is switched off again or until the end of the current session/job.

It is therefore of utmost importance to keep track of which capabilities have been assigned and to switch the capabilities off when they are not needed anymore. The information required for the actual capability switching can be transmitted to the DCAPS program by means of the ;INFO= parameter of the :RUN command. The following rules can be implemented :

- If a capability is specified in the INFO-string either as it is or preceded by a plus sign (+) the capability will be switched ON.
- If a capability is specified in the INFO-string preceded by a minus sign the capability will be switched OFF.
- Multiple capabilities specified in the INFO-string must be separated by commas (,).
- The occurrence of the word VERIFY in the INFO-string, separated from the other information by commas, will cause DCAPS to display the status of the capabilities (ON or OFF) after all capability switches have taken place.

The following examples would be valid commands to run DCAPS according to the rules as laid out above :

```
:RUN DCAPS.PRIV.SYS;INFO="AM,PH"
:RUN DCAPS.PRIV.SYS;INFO="VERIFY"
:RUN DCAPS.PRIV.SYS;INFO="-SM,AM,+OP,+PH,VERIFY"
```

Another approach is to design DCAPS as a routine, residing in a privileged SL. For purposes of clarity, this routine will be viewed as accepting as a parameter one character string with the same layout as the INFO-strings as described above. A valid call to the routine would then be :

```
DECAPS (INFO); or
CALL DCAPS (INFO) or
CALL DCAPS USING INFO
```

whereby INFO would be a character string with the contents "AM,PH", "-SM,AM,+OP,+PH,VERIFY", etc.

An advantage of this method of parameter passing is, that to the user nothing changes in the case that Hewlett-Packard decides to add a number of new capabilities, change the effect of one or more capabilities, etc. Only DCAPS must then be changed to support the new (changed) possibilities.

4. RESCOM, the Restrictive Command Intrinsic.

As can be seen in the diagram below, the RESCOM idea does not have to be restricted to working together with the DCAPS utility. RESCOM can also be used to restrict a "normal" users in the MPE commands, that can be executed programmatically.

Now four types of users can be recognized :

- The standard user, who has the standard MPE possibilities at his disposal.
- The restricted user, who can only perform programmatically the commands, that are accepted by RESCOM.
- The special user, who will now and then get permission to use one or more special capabilities, but who can only execute programmatically the commands, that are accepted by RESCOM.
- The System Management that normally has all capabilities at its disposal.

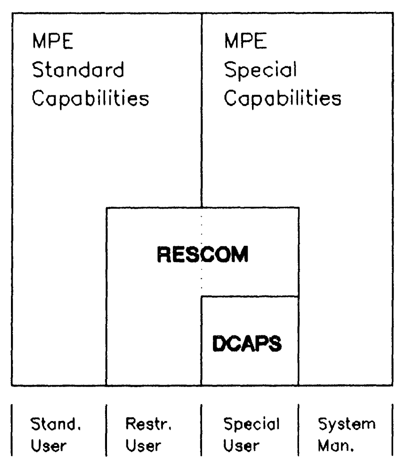


Figure 2. RESCOM, the Restrictive Command Intrinsic.

In order to intercept calls to the MPE command intrinsic, screen the contents of the call and to decide whether the call should be rejected or submitted to the MPE command intrinsic, an alternate command intrinsic, or RESCOM, must reside in the SL, that is attached to the program calling the MPE command intrinsic.

This alternate command intrinsic or RESCOM must recognize the following authorities, possessed by the user calling the intrinsic :

- The authority to execute all MPE commands.
- No authority to execute any MPE command.
- The authority to execute a number of MPE commands.

The first two possibilities will not present any real problems when designing RESCOM. The last one, however, is more tricky and it requires at least some kind of list of authorized or non-authorized commands. The choice must be made by storing these restrictions either in a file or in an extra data segment. For performance reasons the extra data segment method will be preferable.

An ideal situation would be if RESCOM could be used for both "normal" MPE commands and programmatically executed MPE commands. However, this excludes MPE itself (via its UDC structure) as the user interface. In this case a menu processor like HPMENU, MenuProcessor or UNIMAN must be used.

Just to illustrate how a menu processor can be used to build an alternate (restricted) MPE command processor the following will show a part of a UNIMAN command section doing just that. As the author is familiar with the UNIMAN menu processor, this package will be used in all examples pertaining to the menu processor. See the next chapters for some more information on UNIMAN.

```
*
* EXAMPLE COMMAND INTERPRETER FOR THE AMSTERDAM CONFERENCE
*
COMMAND COM06,CONTROLLED
SET UPSHIFT
*
LABEL LOOP
DISPLAY :
ACCEPT MPE
IF PARM MPE EQ BYE
   STOP
ENDIF
IF PARM MPE EQ MENU
   GOTO END
ENDIF
:!MPE!
IF ERROR
   DISPLAY !ERROR!
   DISPLAY ENTER "MENU" TO GET BACK TO THE UNIMAN MENU
ENDIF
GOTO LOOP
*
LABEL END
END
*
*
```

Of course, when executing this file, RESCOM must be activated by running UNIMAN with the ;LIB= parameter, indicating the SL that contains the RESCOM routine. Furthermore, RESCOM must have the information on the commands that are authorized and/or the commands that are not authorized, at its disposal.

As soon as this part of the UNIMAN designer file is executed, the screen will be cleared and a normal MPE prompt (:) will be displayed. From this point onwards, any MPE command entered will be executed, if it is OK with RESCOM.
The BYE command will terminate the execution of UNIMAN. If UNIMAN is part of a UDC ending with a BYE command, the user will be logged off automatically. If the command MENU is entered, control will be returned to the UNIMAN MENU-section, that called this COMMAND-section COM06.

Special considerations have to be taken into account when determining the location of the SL containing the RESCOM intrinsic.
For all accounts outside the SYS account, that have to run programs to be restricted by RESCOM, the SL containing RESCOM must reside in the PUB group of that account and the programs must be run with the ;LIB=G or ;LIB=P parameter.

For programs in the group PUB.SYS, there is a catch, because PUB.SYS already contains an SL (the system SL). It is therefore necessary to migrate all programs in SYS, to be restricted by RESCOM, to a group like, for instance, UTIL.SYS or LIB.SYS. The SL containing RESCOM must also reside in that group and all programs must be run with the ;LIB=G parameter.

5. MENUPROC, the Alternative Command Interpreter.

Because DCAPS, the Dynamic Capability Switch, uses some special capabilities itself (especially the PM capability), it is mandatory that the regular user or even anybody not being the System Manager has no access to this program. Also the possibilities created using DCAPS should be shielded from direct access by the user. It is therefore, that some form of alternative command interpreter or menu processor should be used to restrict the activities of the users outside the application that they must process.

Enter MENUPROC. For the purpose of this presentation MENUPROC is not an existing system, but more a name to be given to the idea of having an intermediate layer between the user and the power of MPE. MENUPROC can be anything from a shrewd application of the UDC structure, using logon UDC's up to a more developed menu processor like HELLO-3000 or UNIMAN.

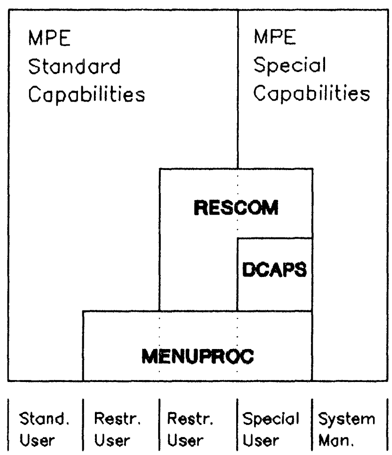


Figure 3. MENUPROC, the Universal Menu Processor

A MENUPROC system to be used for this purpose should be designed along the following reasonable criteria :

- MENUPROC should guide the user from the moment he logs on to the machine until he logs off again. It should be impossible for the user to access MPE directly. The logical way to solve this one is start MENUPROC from a logon, non-breakable UDC.

- MENUPROC must at least be capable of
  . Running programs.
  . Streaming jobs.
  . Executing MPE commands.
  . Detecting errors.

- It would be an advantage if MENUPROC would possess its own security layer, enabling, for instance, extra passwords for certain users or for certain sensitive actions.

- It would be an advantage if MENUPROC would have the possibility for the conditional execution of certain actions based on, for instance, errors detected, the current logon device, the time of day, etc.

Looking at this list it will be obvious that rather simple systems such as the MPE UDC's and HPMENU will have much difficulties in meeting the design criteria for MENUPROC. However, numerous menu processing systems are on the market or can even be found on the Interex Contributed Software Library. It is the responsibility of the system management to select and test one or more of these systems to be used in combination with DCAPS and RESCOM. It is not advisable to start building one's own MENUPROC because the market offers systems for any purpose at reasonable prices.

In order to see, how the combination of DCAPS, RESCOM and MENUPROC can work, the UNIMAN package will be used to fulfill the tasks of MENUPROC. More information on UNIMAN can be found in the UNIMAN User Manual. For the time being if suffices to say, that UNIMAN uses a designer file, consisting of MENU sections, describing screen layouts, the function key labels, and the names of the COMMAND sections, that have to be executed when a certain key is pressed, and COMMAND sections, describing the actions to be performed. The UNIMAN language is rather selfdescribing. In the following examples detailed explanations will be provided in those cases that are not directly clear on reading.

```
*
* DEMONSTRATION UNIMAN DESIGNER FILE FOR AMSTERDAM CONFERENCE
*
* INITIALIZATION, UNIMAN PASSWORD CHECK
*
COMMAND INITIAL
CLEAR
DISPLAY AT 1010,PLEASE ENTER YOUR UNIMAN PASSWORD :
GETPASS EXPASS,CONSOLE
LOAD DEMO1
END
*
*
*
```

```
*
*
* SIMPLIFIED USER MENU DEFINITION
*
MENU DEMO1
DISPLAY AT 1010,DEMONSTRATION MENU 1
DISPLAY AT 1110,===================
KEY 1,CHANGE\PASSWORD,COM01
KEY 2,TDP\,COM02
KEY 3,FCOPY\,COM03
KEY 4,STREAM\CS QUEUE,COM04
KEY 8,EXIT
END
*
*
* CHANGE USER MPE PASSWORD USING AM CAPABILITY
*
COMMAND COM01
CLEAR
DISPLAY AT 1010,PLEASE ENTER NEW PASSWORD :
ACCEPT PASSWORD
:RUN DCAPS.PRIV.SYS;INFO="+AM"
:ALTUSER !USER!;PASS=!PASSWORD!
:RUN DCAPS.PRIV.SYS;INFO="-AM"
END
*
*
* TEXT AND DOCUMENT PROCESSOR USED WITH STANDARD CAPABILITIES
*
COMMAND COM02,CONTROLLED
:RUN TDP.PUB.SYS
IF ERROR
   DISPLAY UNABLE TO RUN TDP
   DISPLAY !ERROR!
ENDIF
END
*
* FCOPY USED FOR FILE MANAGEMENT WITH AM CAPABILITIES
* MAKE SURE RESCOM IS ACTIVATED
*
COMMAND COM03,CONTROLLED
:RUN DCAPS.PRIV.SYS;INFO="+AM"
:RUN FCOPY.UTIL.SYS;LIB=G
:RUN DCAPS.PRIV.SYS;INFO="-AM"
END
*
*
```

```
*
*
* STREAM A JOB IN THE CS QUEUE USING OP CAPABILITIES
*
COMMAND COM04,CONTROLLED
DISPLAY AT 1010,ENTER NAME OF JOBFILE :
ACCEPT JOBFILE
:RUN DCAPS.PRIV.SYS;INFO="+OP"
:JOBPRI CS
:STREAM !JOBFILE!
:JOBPRI DS
:RUN DCAPS.PRIV.SYS;INFO="-OP"
END
*
*
* EXIT UNIMAN, PREFERABLY FOLLOWED BY A BYE IN LOGON UDC
*
COMMAND COM08
STOP
END
*
*
```

Remarks.

- The CONSOLE keyword of the GETPASS statement indicates, that all password violations must be logged onto the system console.
- The backslash (\) in the KEY statement causes the text in the function key labels to be centered.
- At the location where a parameter name is enclosed in exclamation marks (!), the value of that parameter is substituted.
- The keyword CONTROLLED of the COMMAND statement causes an automatic CLEAR and DISABLE at the start of the section and a REFRESH at the end of the section.
- The UNIMAN parameter ERROR always contains the last MPE error encountered, if any.

6. Tying it all together.

The three systems, that are defined so far, and their relations can be depicted by the following diagram :

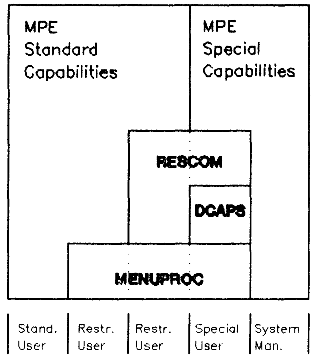


Figure 4. The Components Needed for the Solution.

However, the three systems still are very separate and are just a number of tools to get to the goal as described in the introductory chapter. The last step is to integrate DCAPS and RESCOM into MENUPROC in order to get a real universal applications manager. The following diagram shows the new situation as viewed by the designers of the user menus.

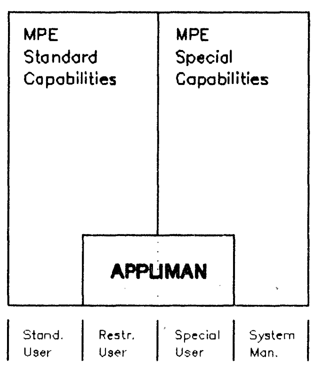


Figure 5. The Integrated Applications Manager.

The twilight zone .....

The thing to decide upon now is how to achieve this integration. Just combining the three programs and routines into one big program is not the solution, because that would require the complete program to run in Privileged Mode.
No, the best thing to do would be to design a number of commands to add to MENUPROC, that can describe the designer's requirements and that interface to the DCAPS and RESCOM subsystems. Every MENUPROC system will have its own particulars, so this presentation will just limit itself to additions that will have to be made to UNIMAN in order to get something like APPLIMAN.

As the UNIMAN designer file must be compiled before it can be executed, a new compiler command $PM will be introduced. $PM indicates, that the designer file may contain statements requiring access to Privileged Mode routines, in this case DCAPS. Default will be $NOPM and the UNIMAN password file SKPASS will contain a list of users, authorized to use the $PM compiler command.

The next command to be added for usage within COMMAND-sections is SWITCHCAP, that will be used to switch capabilities. SWITCHCAP accepts as a parameter a character string as described in the chapter on DCAPS with the exception of the VERIFY keyword, so the following UNIMAN statements are perfectly valid :

```
SWITCHCAP AM,PH
SWITCHCAP -SM,AM,+OP,+PH
```

The UNIMAN VERIFY statement will be extended with the keyword CAP that will cause the display of the status (ON or OFF) of the MPE capabilities. VERIFY ALL will also include this display.

The integration of the restrictive command interpreter will be performed using an extra data segment to store all commands with their ON/OFF indicators. To initialize the status of the commands, two compiler commands have to be added : $ALLOW to initialize all commands as being allowed and $DISALLOW to initialize all commands as being disallowed.

The COMMAND-section statements ALLOW and DISALLOW must be added to allow or disallow commands on an individual basis. Both commands accept as a parameter one or more commands, separated by commas. An ALLOW for a specific command stays in force until it is disallowed by a DISALLOW command.

The UNIMAN VERIFY statement will be extended with the keyword ALLOW that will cause the display of the status (ALLOWed or DISALLOWed) of the MPE commands. VERIFY ALL will also include this display. The UNIMAN designer file as described in the chapter on MENUPROC will look like this after the addition of the new commands : (see next pages)

Using this setup, an extra security measure can be achieved. Instead of changing the capabilities in the Job Information Table (JIT), it is now possible to change most capabilities on the stack of the UNIMAN process. The advantage is that, when UNIMAN is aborted for whatever reason, the special capabilities will be gone also, so the user can never end up within MPE with some special capabilities left.

```
*
* DEMONSTRATION UNIMAN DESIGNER FILE FOR AMSTERDAM CONFERENCE
*
$PM
$DISALLOW
*
* INITIALIZATION, UNIMAN PASSWORD CHECK
*
COMMAND INITIAL
CLEAR
DISPLAY AT 1010,PLEASE ENTER YOUR UNIMAN PASSWORD :
GETPASS EXPASS,CONSOLE
*
*
ALLOW BUILD,FILE,HELP,LISTF,PURGE,RELEASE,RENAME,REPORT,RESET
ALLOW RESTORE,SAVE,SECURE,SHOWJOB,SHOWME,SHOWOUT,SHOWTIME,STORE
ALLOW STREAM,TELL,TELLOP
*
*
LOAD DEMO1
END
*
*
* SIMPLIFIED USER MENU DEFINITION
*
MENU DEMO1
DISPLAY AT 1010,DEMONSTRATION MENU 2
DISPLAY AT 1110,===================
KEY 1,CHANGE\PASSWORD,COM01
KEY 2,TDP\,COM02
KEY 3,FCOPY\,COM03
KEY 4,STREAM\CS QUEUE,COM04
KEY 8,EXIT
END
*
*
* CHANGE USER MPE PASSWORD USING AM CAPABILITY
*
COMMAND COM01
CLEAR
DISPLAY AT 1010,PLEASE ENTER NEW PASSWORD :
ACCEPT PASSWORD
SWITCHCAP +AM
:ALTUSER !USER!;PASS=!PASSWORD!
SWITCHCAP -AM
END
*
*
```

The twilight zone .....

```
*
*
* TEXT AND DOCUMENT PROCESSOR USED WITH STANDARD CAPABILITIES
*
COMMAND COM02,CONTROLLED
:RUN TDP.PUB.SYS
IF ERROR
   DISPLAY UNABLE TO RUN TDP
   DISPLAY !ERROR!
ENDIF
END
*
*
* FCOPY USED FOR FILE MANAGEMENT WITH AM CAPABILITIES
* MAKE SURE RESCOM IS ACTIVATED
*
COMMAND COM03,CONTROLLED
SWITCHCAP +AM
:RUN FCOPY.UTIL.SYS;LIB=G
SWITCHCAP -AM
END
*
*
* STREAM A JOB IN THE CS QUEUE USING OP CAPABILITIES
*
COMMAND COM04,CONTROLLED
DISPLAY AT 1010,ENTER NAME OF JOBFILE :
ACCEPT JOBFILE
SWITCHCAP +OP
:JOBPRI CS
:STREAM !JOBFILE!
:JOBPRI DS
SWITCHCAP -OP
END
*
*
* EXIT UNIMAN, PREFERABLY FOLLOWED BY A BYE IN LOGON UDC
*
COMMAND COM08
STOP
END
*
*
```

The twilight zone .....

Biography.

Jelle Grim worked for the same company, the contractor Royal Boskalis Westminster from 1966 to April 1984. Starting as a civil engineer in the technical area he almost immediately switched over to the computer section. The Boskalis automation between 1968 and 1984 changed from in-house IBM S/3, through external data processing at a CDC service bureau using local Datapoint mini's, to in-house HP3000 equipment from 1978 onwards. When Jelle left Boskalis he was Information Network Manager in charge of a dual HP3000 network serving approximately 200 terminals and microcomputers both in Holland and abroad.
In April 1984 Jelle and his partner Rene van Geesbergen together founded Holland House, a company specializing in HP3000 system management consultancy and software products.
Jelle is secretary of the Dutch Users Group HP3000 (DUG) and a member of the Amsterdam 1985 Host Committee.

The twilight zone .....

Towards a Better Utilization of Print Resources.

By Jelle Grim,

Holland House, AALST, Holland.

Abstract.

The output spooling system, that is part of the MPE operating system, was designed a fair number of years ago for the HP3000 configurations of that time.
However, the development of this MPE spooler has somewhat fallen behind compared to the developments of HP hardware, communications and to the way, the users want to have processed their printed output.
These limitations of the MPE spooler become particularly apparent in the following area's :

- printing of output on printers, that are connected to other (remote) HP3000 systems in a network;
- a limit of 16 remote spooled printers per HP3000;
- difficulties when spooling to non-HP equipment;
- a rather crude interface between the console operator and the spooling system.

In this paper, a number of idealized configurations will be discussed, followed by a number of design criteria for a (supplementary) spooling system as derived from these configurations.
From these design criteria some actual design considerations, such as the various device types to be supported, the impact of a new spooling system on the various types of users and perhaps even some futuristic spooler developments will be discussed.

Contents.

1. Introduction.
2. Configurations of this time.
3. Design criteria.
4. Device types to be supported.
5. Impact on the users.
6. The future.

1. Introduction.

Looking at the history of printing output from computers it is evident, that this subject always has been very important to all users of computer systems. It is and always has been unthinkable to consider a computer system without the means for hardcopy output.

The early systems always had a printer attached to them, that could be accessed from one program at the time. Other programs wanting to use the printer just had to wait until the program, occupying the printer, had finished. However, as the computers became more powerful and were able to run a larger number of programs concurrently, it was necessary to design a method whereby the various programs could produce output independent on the availability of the printer.

As in so many other fields IBM lead the way by designing the principle of Simultaneous Peripheral Operations On Line, the acronym SPOOL of which will always be associated with the buffered printing of output or SPOOLing. The principle of spooling is rather straightforward. All programs wanting to use a printer resource "print" their output to a disc file. After completion this "spoolfile" is queued for printing. A separate spooler process controls the actual printing of the spoolfiles on the print resource(s).

Looking back in history at the control of printing operations, the keyword has always been "centralized". Not only the relative high price of the printers, but also technical considerations such as the maximum distance of the printer to the CPU have been the cause of this development. Names like "system printer", "central lineprinter" are still being used by all people involved in data processing.

For the last couple of years, however, a trend can be recognize towards decentralized printing. The improvement of communications and the price decreases for printing equipment have helped a lot in this area. Also the fact, that people want their output quicker and do not want to wait until centrally printed output is distributed, has contributed to the development of decentralized printing.

Originally this decentralized printing was performed in a non-spooled way, i.e. like the system printers in the first section of this introduction. However, with the formation of remote clusters of users and the availability of fair priced, medium speed, high quality print resources there is a requirement for remote spooled print operations.

The spooling system that is part of the MPE operating system for the HP3000 series of computers was designed in the early seventies, along with the MPE operating system itself. Since that time, a number of extensions and improvements have been implemented, but still the effectiveness of the MPE spooling system has fallen behind the development of printer hardware, of communications and of the way, that the users want to use the print resources.

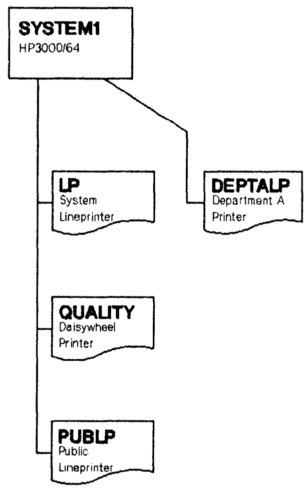
This paper will try to define the shortcomings of the MPE spooling system by means of the discussion of a number of HP3000 configurations that can be found in data centers all over the world. Although the MPE spooling system can cope with a large number of requirements from these configurations, the addition of a number of features would greatly enhance the efficiency of the usage of the print resources, the satisfaction of the users of these configurations and the ability for the operations staff to really control the print operations of these data centers.

Based on the above a number of design criteria for a more universal spooling system for the HP3000 series of computers will be defined and discussed.

The last part of this paper will go deeper into the design of an alternative spooling system in terms of the definition of the various device types that have to be supported in order to comply with the design criteria, by discussing the impact of the alternative spooling system on the various types of HP3000 users, and by trying to look in the future to discuss even more advanced spooling matters.

2. Configurations of this time.

This section will deal with a number of HP3000 configurations, that will illustrate if and how the MPE spooling system can accommodate the requirements of the modern HP3000 user. Configuration example 1 shows the classic stand-alone HP3000 system with a number of printers connected to it.



Example 1. One HP3000 spooling output to multiple printers.

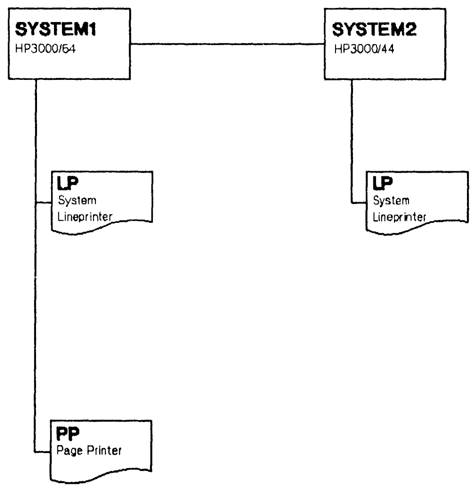
The various printers of this configuration are used in the following way :

* LP is the system lineprinter, located in the computerroom under the control of the Console Operator. The printer is emptied once every hour during office hours. After office hours the printer is switched off; it is then not available to the users.
* QUALITY is a daisywheel printer, located in the computerroom, and used to produce letter quality output from TDP. It is subject to the same rules as the system lineprinter.
* In order to serve those users, who are working after office hours or who do not want to wait for the hourly emptying of the central printers, the spooled printer PUBLP (an HP2631B medium speed serial matrix printer) is located outside the computer room in a public space. Every user can send output to this printer, but neither the System Management nor the Console Operator has the responsibility for the emptying of the printer, for the distribution of the output, etc.
* Department A, located in the next town in a small subsidiary office, uses the printer DEPTALP, an HP2631B matrix printer through a leased line, modem and multiplexer combination.

At first sight, this configuration will not present any real problems to the MPE spooling system, but some slight modifications to the requirements will change the picture completely. For instance, what if :

- The QUALITY printer is a non-HP device, that must be used by other applications than TDP alone ?
- The PUBLP printer is not an HP2631B printer, but an HP2932 printer, connected to an HP150 microcomputer, the combination working as a remote printing station ?
- The DEPTALP printer must be connected through a X.25 public data network by means of an HP2334 cluster controller ?
- The HP3000 configuration becomes so big, that another 20 remote spooled printers are required ?

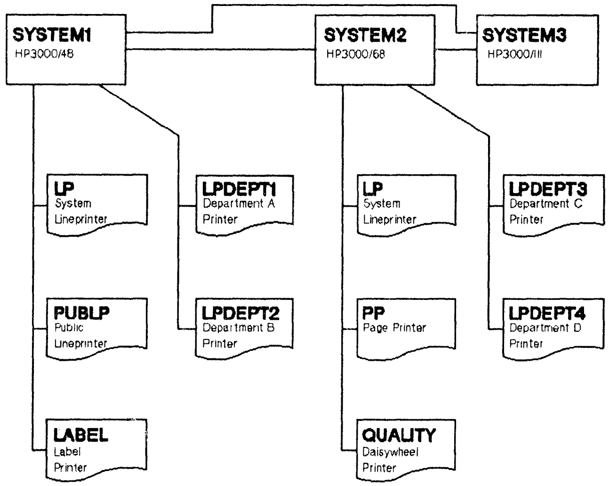
The next configuration shows a simplified example of a shared printer resource.



Example 2. Multiple HP3000 systems sharing a printer

The one thing that really would be nice in this situation is the possibility to share an expensive printer resource like PP, the HP2680A Page Printer, with two or more HP3000 systems. Hewlett-Packard are offering a number of solutions like remote file equations, HP-IB switches, storing/restoring of spoolfiles using the SPOOK program, and there are some unsupported products such as SPOOK2, RSPOOL, etc. However, neither of these solutions are quite satisfactory to the user of the system. Almost all solutions require either intervention of the Console Operator or unnatural actions of the End User like starting remote sessions, etc. The ideal situation would be for the user to just specify, that his output must be sent to the device PP, and let the system worry on how this must be accomplished, on where the device PP is located and on how to get the output there.

The third and last configuration shows an amalgamation of the previous two with some additional niceties thrown in for fun.



Example 3. Multiple machines spooling to multiple printers

Now this is a situation the MPE spooling system cannot cope with, especially if the following requirements are added to the already impressive configuration :

* SYSTEM3 was added to the configuration just to provide some extra processing power. It must use the print resources, that are already available.
* All three HP3000 systems will normally send output to one or more printers, connected to one of the other machines.
* The printer LABEL is used as a special purpose printer. In this case it is a low cost, medium speed matrix printer, always loaded with three-up mailing labels. This eliminates the need for changing forms and cluttering up the other printers with labels, whilst offering all network users the possibility to print labels on a standard format anytime.
* Often, the same output is sent to the four departmental printers LDEPT1, LDEPT2, LDEPT3 and LDEPT4. It would be very useful, if these printers could be addressed by a common name, like ALLDEPT, whereby all output, directed to ALLDEPT, would be printed on all four printers.
* Sometimes very confidential information is generated on SYSTEM3 and sent to the page printer PP on SYSTEM2. This output must be recognizable from the normal output by adding a banner page before and after the spoolfile, containing the blown-up text "CONFIDENTIAL". The operator will have special instructions with regard to the distribution of this output.
* In view of the variety of printers and the dependency of many users of their printed output, it would be nice if routing could be performed between all printers in the network. Especially in the case of maintenance, breakdown, special forms handling, etc. this is almost a must.

3. Design Criteria.

The configurations as described in the previous section have shown both a number of situations the MPE spooling system can handle and a number of situations, that this spooler cannot handle. Based on these observations, a number of design criteria for a more universal spooling system can be defined. These criteria are :

- The possibility to spool output to remote machines.
- The possibility to overcome a number of MPE limits.
- Transparency to the users.
- Spoolfile compatibility with the MPE spooler.
- Compatibility with the MPE operating system.
- Improved device and spoolfile control.
- Minimal overhead and usage of system resources.

These design criteria will be discussed in detail in the following pages.

Spooling output to remote machines.

The standard HP3000 computer system offers a number of methods to transfer spoolfiles from one system to a printer connected to another system. The most widely used methods are either the application of an HP-IB switch, which limits the number of systems sharing a print device to three, and the usage of one of the unsupported programs SPOOK2, DSPRINT, RSPOOL , with all disadvantages attached to them like Privileged Mode usage, no maintenance, etc.
Other solutions, equally unattractive, are the usage of the SPOOK program to store and restore spoolfiles to and from tape, or the application of remote file equations. The first method necessitates heavy operator intervention, the second one forces all users to initiate a remote session, thereby increasing the load on the DS-connection.

What really would be needed is a system, whereby one process is controlling all transfers of spoolfiles in one direction of a DS-connection, thereby using only one remote session. Spoolfiles would be transferred using a store-and-forward method, preferably under complete control of the Console Operator. The Console Operator must have the power to set an OUTFENCE on the DS-transfer and to open and close the DS-connection at will. Even a time table for opening and closing the DS-connection for spoolfile transfer could be considered.
Spooling output to more than one remote system must be possible.

Overcoming the MPE limits.

The current MPE spooling system imposes a number of limitations on the users, the most important of which are :

- The maximum number of remote spooled printers is 16.
  Although most sites do not consider this limitation a big handicap, those configurations with a big terminal network will get into trouble at some point in time. Especially the bigger machines like the model 68 do have the potential for connecting 20, 30 or even more remote printers. Therefore this limitation must be circumvented one way or another.

- Only certain types of printers are supported.
  Historically seen only the HP2631B printer was supported by the MPE spooling system as a remote spooled device. The CIPER MIT added the the HP2608S using MTS to the list and lately the HP293x (SULLIVAN) and HP2687 were added to the list. However, the common denominator of these printers are the letters HP. Remote spooling of non-HP printers is not supported by MPE.
  Now, although HP printers are famous for their quality, it is not unthinkable that people would like to connect non-HP printer equipment to their HP3000 network, based on considerations like a better price/performance, special printer features, etc. Any universal spooling system should therefore offer the possibility to use non-HP equipment as (remote) spooled printers.

- No HP2334 X.25 Cluster Controller support.
  As the Public Data Networks are gaining rapidly in importance in the field of data communication between machines, but also for terminal-to-computer traffic, it is of utmost importance, that all facilities offered on the "normal" asynchronous transmission methods will also be available for X.25 users. Of course this includes the application of remote spooled printing.

Transparency to the users.

The MPE spooling system is transparent to the user insofar, that the user only needs to specify the name (class) of the printer in the file equation of his output file in order to submit his output to the spooler, which in its turn will take care, that the output ends up at the right place. The majority of standard software packages automatically interface to the MPE spooling system.

Any spooling system to be built should be designed along these lines. The user does not want to see a fancy banner telling him, that he is using the so-and-so spooler of that-and-that softwarehouse, because it is not relevant. The user even does not have to know that an additional spooling system is being used.

The only thing the user has to know is the name of the printer he wants to send his output to and let the spooler worry as to how to find that specific device and how to submit the spoolfile to it. Also, all software sending output to certain spooled devices, should continue to do so. This also eliminates the need for heavy conversion activities.

Spoolfile compatibility with the MPE spooler.

Surprisingly, not very many companies have tried to introduce alternative spooling systems onto the HP3000 market. The spooling systems, that have surfaced, normally address one of the limitations of MPE or one specific situation, requiring a special solution. However, what almost all these systems have in common is the fact, that they use their own "spoolfiles", mostly normal CCTL ASCII files. Sometimes, these files and/or their attributes and/or device information are stored in a SPOOLER Database.

The disadvantage of this method is evident. The processing of spoolfiles in these situations will always take place either by MPE or by the alternative spooling system. There is no way in which the two systems can cooperate. Even some HP subsystems have introduced individual methods of output spooling. TDP, for instance, employs a spooling system for serial printers that uses a special spoolfile format. HPWORD uses a queuing system, whereby files to be printed are queued and are not accessible as long as they are not printed.

The only sensible thing to do when designing an alternative spooling system is to hold on to the MPE standard spoolfile format. Any system built along this line will have the possibility to work with the MPE spooler instead of next to the MPE spooler. A free exchange of spoolfiles between the two systems is possible, thereby offering the possibility to let each system concentrate on those things it is good at.

Compatibility with MPE.

In addition to the spoolfile compatibility between the MPE spooler and an alternative system, also the command compatibility is important. It is not very smart to try to design one's own commands, because the only result will be confusion, especially on those sites where both spooling systems are used together.

Therefore, all existing spooler commands like STARTSPOOL, STOPSPOOL, RESUMESPOOL, UP, DOWN, OUTFENCE, HEADON, HEADOFF, SHOWOUT, ALTSPOOLFILE and DELETESPOOLFILE should be implemented and offer at least the same possibilities as their MPE counterparts. However, an effort can be made to actually improve some of the MPE commands by offering more power and/or more features than MPE does.

Improved device and spoolfile control.

If all goodies and features as described above must be implemented in a new universal spooling system, it follows naturally that a number of new commands will have to be defined in order to cope with these things. In addition, the interface between the MPE spooling system and the Operator has become a bit old-fashioned. (Ever tried to purge 15 deferred spoolfiles with the DELETESPOOLFILE command?)

The new commands and improved commands can be recognized in the field of :

* Control over the transfer of spoolfiles over a DS-connection, f.i. the commands DSOPEN and DSCLOSE and an improved version of the OUTFENCE command enabling the setting of an outfence for a remote system.

* More insight in the status of the spooled devices by implementation of a SHOWSYS command to show the status of the remote systems and an improved SHOWDEV command, both showing not only static information but also dynamic information.

* New commands STARTROUTE, STOPROUTE and SHOWROUTE to control the routing activities between the spooled devices in the case of breakdowns, maintenance, etc.

* A better spoolfile control by means of improved implementations of the SHOWOUT, ALTSPOOLFILE and DELETESPOOLFILE commands. Minimum requirement for the DELETESPOOLFILE and ALTSPOOLFILE commands is the ability to process groups of spoolfiles, based on any selection, with a single command.

* Improved special forms handling by means of the new commands SHOWFORMS and ALTFORM. This enables the Console Operator to really, dynamically control the special forms handling.

* And lots more, thereby offering the user a possibility to abbreviate commands according to his own taste similar to the MPE User Defined Command structure.

Minimal overhead and system resource usage.

Obviously, this design criteria is one, that will be on any designers list, because nobody wants a heavy package, that eats all system resources including tons and tons of CPU seconds.

Still, for this particular subject some observations can be made. The MPE spooling system normally runs in the BS and sometimes even in the AS priority subqueue. This means, that on a very busy system the spooler can easily try to monopolize the system resources. Chance is, that this situation is exactly the cause of the rather rigid MPE limitation of 16 remote spooled printers.

It could prove wise, therefore, to design an alternative spooling system in such a way , that it runs in the CS or even the DS subqueue, thereby competing in a healthy way with the normal workload on a specific machine.

This would mean, that on a very busy system also the spooling activities would slow down.

So what !!

4. Device types to be supported.

It is obvious that, to fulfill the various types of tasks as outlined in the design criteria, a number of different device types can be recognized, that will help in building a more universal spooling system. At this point in the presentation it is time to try our hand on the definition of these device types. This definition will be based on :

- The design criteria as specified in the previous chapter.
- A good knowledge of the HP3000 system and its MPE operating system.
- A generous portion of common sense.

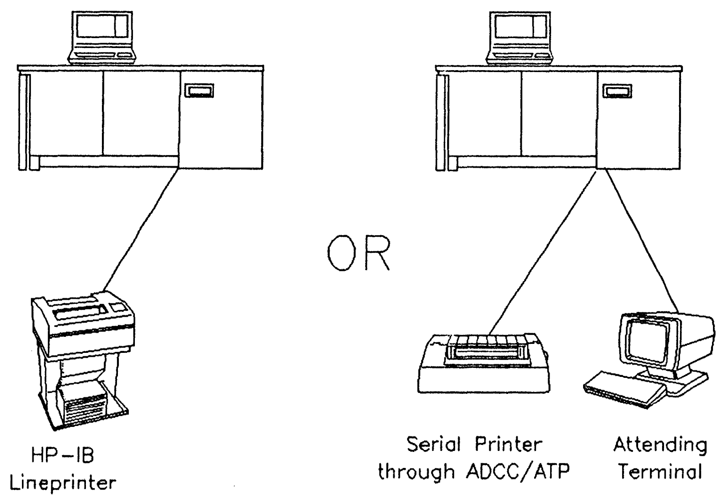
For the time being, the following device types will be defined :

- Devices spooled by the MPE operating system.
- Print devices, that can log on as a terminal.
- Print devices, that are permanently connected to the HP3000.
- Print devices, connected to a remote machine.
- Distribution devices.
- Banner devices.

These six device types, the first three of which are physical devices and the last three of which are, more or less, logical devices, will be discussed in the following pages.
It should be noted that, by cleverly combining two or more of these device types, even more device types can be designed.

Devices spooled by the MPE operating system.

In spite of a number of disadvantages, the MPE spooler does have a large quantity of good things. In view of this there is no need to try and improve on the MPE spooler on those things, that it is good at. This means that the MPE spooler will keep spooling output, even if an additional spooler system will be implemented. The trick is, to let the two systems work together as harmoniously as possible, thereby enabling each spooler to concentrate on the things it is good at.
However, this could require the new spooler to know, which tasks are performed by the MPE spooler, or more specific, which devices are serviced by the MPE spooler. This way, these devices can still be included in the new goodies that will be part of the new spooler like routing, improved device control, broadcasting of spoolfiles etc. Schematically, this type of device looks like :

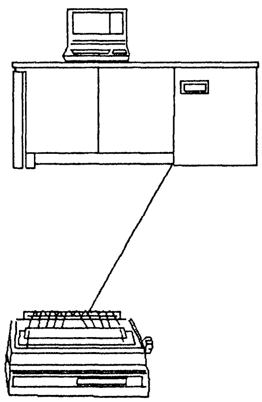


Devices Spooled by MPE

It is essential, that the new spooler does not interfere with the actual output spooling activities to these devices. Only specific new activities like customized banners, broadcasting, routing will be handled by the new spooler.
It is even thinkable, that the new spooler will transport a spoolfile over a DS-line and than submit it again to the MPE spooler to have it printed on, say, a page printer.
In short, no harm will be done, if these devices are included in the configuration of the new spooling system; only benefits will be derived from it.

Print devices, that can logon as a terminal.

In most HP3000 configurations a large variety of print devices is used. Apart from the conventional printers like system lineprinters and serial printers, directly connected to a port, more and more printer combinations are used. These printer combinations can be printing terminals, like the HP2635, printers connected to terminals through the terminal printer port, printers connected to micro-computers. Yes, even printers connected to HP250 computers are included in this category. These printer combinations can be connected to the HP3000 system directly, through modem/multiplexer combinations, through X.25 cluster controllers, etc.



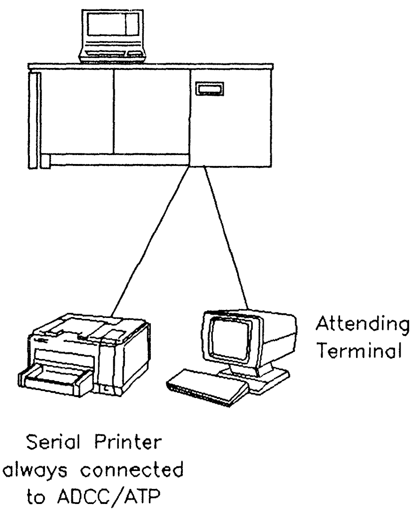
Serial Printer
with Keyboard
through ADCC/ATP

Log on Device

The MPE spooler normally does not support these printing devices. As these 'logon devices' are used more and more, the new spooling system should support them. From a design point of view, this device type is rather easy to implement. The device can logon as a terminal, start a normal session and can initiate the printing of output on the attached printer. Communication between the Remote Operator and the spooling system can take place through the keyboard, if required after interrupting the printing process by, for instance, a 'Ctrl-Y'.
When there are no spoolfiles left to be printed, the printer combination can be used for any type of work, because the Remote Operator decides, when the spoolfiles waiting for the device will be printed.

Print devices, permanently connected to the HP3000.

Printers without a keyboard, like the HP2631 or the HP2932, are normally attached to a computer port on a permanent basis, albeit sometimes over a leased line or over an X.25 data network. As long as these printers are switched online, they are ready to receive output. From a spool/print technical point of view, these devices can be served by a driver similar to the 'logon devices'.

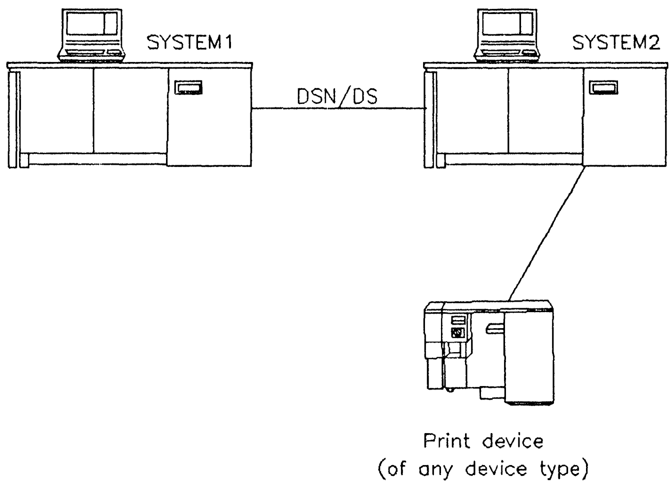


Permanently Connected Device

Because printers of this type do not have a keyboard, it is not possible either to start a session or to communicate with the spooler through the keyboard. Therefore it is necessary to assign an attending terminal, that can be used to control the spooling/printing process, similar to the MPE remote spooled printers.
However, it will be an advantage if just any terminal in the neighborhood of the printer can be used as an attending terminal. When this terminal is not communicating with the spooler, it must be available to perform other activities. These devices are also normally under the control of a Remote Operator.

Print devices, connected to a remote machine.

Seen from one HP3000 system, all printers connected to another HP3000 system in a network will be regarded as remote devices. Therefore a remote device is a logical instead of a physical devices. On the remote machine, the device on hand can be one of the other types of devices, i.e. an MPE spooled device, a logon device, a distribution device, etc. It is even possible, that the device is again a remote device, so that a spoolfile will pass several HP3000 systems on its way to its final destination.
Schematically this looks as follows :



Remote Device

Of course, the processing of spoolfiles for remote devices will be dependent on the status of the connection between the various HP3000 systems in the network. Even more control can be supplied by offering the possibility to set an OUTFENCE on a connection, so that the actual transfer will take place at the discretion of the Console Operator, or to include the possibility to define a time table for spoolfile transfer, similar to the HPDESK time table for mail transfer.

As far as the actual communications link between the HP3000 systems is concerned, all Hewlett Packard options as well as some alternative options should be supported. This means, that the spooler must support DSN/DS on BI-SYNC or X.25 lines, but also for instance the cheap UNIPORT port-to-port communication system.

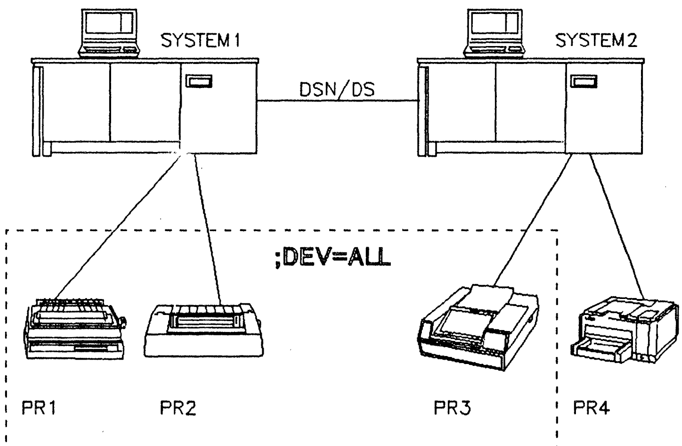
Distribution devices.

A special device class will have to be formed by the so-called distribution devices. A distribution device is a logical devices,

that can best be looked upon as a collection of one or more printers with a group name. In other words, a distribution device can contain one or more destination devices. Two types of distribution can be recognized :

- The spoolfile must be printed on all destination devices. This is particularly useful for bulletins, newsletters, etc.
- The spoolfile must be printed on only one of the destination devices. This type of distribution can be used to distribute the workload evenly over two or more printers.

The distribution device looks like this :



Broadcast Device

Note, that the destination devices must also be accessible through their own name. As can be seen, the destination devices in their turn can be any one of the other device types. A destination device can be another distribution device. However, care should be taken with the configuration of distribution devices, because a distribution device may never distribute spoolfiles to itself, either directly or indirectly. The result would be a very fast and powerful spoolfile generator.

Banner devices.

In order to be able to add customized banners to spoolfiles, the banner device will be defined. A banner device also is a logical device, with a destination device attached to it. The destination device in its turn can be any device type. Two types of banners must be supported :

- An alternative banner page, that resides as a normal file in the HP3000. This is a banner with a fixed content for each banner device. Per device a header and trailer banner can be defined. When processing a spoolfile for a banner device, the spooler will have to add a header banner in front of the spoolfile, or a trailer banner following the spoolfile, or both and then direct the new spoolfile to the destination device.
- In order to support also customized banners with dynamic contents, the new spooler must supply user exits, transferring control to user procedures in the SL, that supply the alternative header and/or trailer banner.

Combined with the HEADON/HEADOFF commands, the complete principle of banner printing will be enhanced greatly through the banner device. Obvious applications of the banner device are :

- To aid the operator with the distribution of output to the end user.
- To supply the users with more data about the spoolfile that was printed.
- To notify the Console Operator of special actions that have to be taken, for instance, with regard to the distribution of the printed output, like the addition of the blown-up text "CONFIDENTIAL".

5. Impact on the users.

The implementation of an alternative output spooling system has an impact on almost all users of an HP3000 system. Both End Users and System Management will benefit from the features of the new system. The impact of the new spooler will be discussed for the types of users who will be involved mostly :

- The End User

- The Remote Operator

- The Console Operator

- The System Manager

The following pages will concentrate on the changes in possibilities and procedures as will be experienced by these user types.

The End User.

Ultimately any alternative spooling system will be built to satisfy the requirements of this person. The spooler will especially have to satisfy the design criteria of user transparency. The ideal situation will be where the End User only has to know the name of the printer he wants his output on. An obvious implementation would be where this device name is specified in the ;DEV= parameter of the :FILE command, in the :OUTCLASS= parameter of the :JOB command, in the /FINAL command of TDP etc.

This way, the user even does not have to know that the alternative spooler exists, or whether his output is processed by the MPE spooler or by an alternative spooler. It will be the task of the spooler to locate the printer in the network and to take care of the transmission of the output to that printer.

The Remote Operator.

Just like the situation where remote MPE spooled printers are being used, the new spooling system also requires the assignment of people to attend to the printers, that are outside the control of the Console Operator. These printer attenders will be called Remote Operators.
The spooling system must supply to these Remote Operators the means to perform, amongst others, the following actions :

- start and stop the printing of spoolfiles on the device
- inspect the queue of spoolfiles waiting to be printed on the device
- setting an outfence for the device
- inspect the status of the device
- switch the printing of MPE banner pages on or off
- delete one or more of the spoolfiles waiting to be printed
- alter the attributes of one or more of the spoolfiles waiting to be printed

Furthermore, the printing of spoolfiles with a special forms message must be executed as efficient as possible.
Communication between the Remote Operator and the spooler can take place either through the device itself (in the case of a log on device) or through an attending terminal (in the case of a permanently connected device).

Note, that all actions to be performed by a Remote Operator pertain only to the device he is attending and to the spoolfiles associated with that specific device. One Remote Operator can attend to more than one remote device. The Console Operator can act as the Remote Operator for remote spooled printers in his vicinity.

The Console Operator.

The Console Operator ultimately controls the day-to-day operation of the output spooling, both in the case of MPE and of an alternative spooling system. It is therefore obvious, that much effort must be put into designing and implementing the interface between the Console Operator and the spooling system in the most efficient and user-friendly way.

Apart from all actions, that can be performed by a Remote Operator, the Console Operator will have the tools to perform the following activities at his disposal :

- starting and stopping the complete output spooling system
- opening and closing the spoolfile transfer links to remote systems in the network
- starting and stopping the routing of output between the various printers in the network
- storing and restoring spoolfiles to and from tape
- recovering the spooling system after a system crash
- controlling the activity logging of the spooling system

In order to perform his task as efficient as possible, powerful commands are required for the Console Operator to be able to DELETE or ALTER groups of spoolfiles, to be able to dynamically handle special forms, to be able to easily detect errors, etc.
Furthermore, the Console Operator can overrule any actions performed by the Remote Operators and he can act as a Remote Operator for remote spooled devices in the neighborhood of the computerroom.

The System Manager.

The System Manager should not have an operational task in the spooling activities with the exception of one. It should be the responsibility of the System Manager to configure the spooling system, i.e. to decide which printers will be spooled, which remote systems will be included in the spooling configuration, which people will be assigned the task of Remote Operator, etc. In addition, the System Manager naturally can overrule any action as performed by the Console Operator and by the Remote Operators.

6. The Future.

Looking in the not too distant future, some other enhancements to the whole spooling activity come to mind. Future releases of either the MPE spooler or an alternative spooling system should offer one or, preferably, more of the following features :

* As Hewlett-Packard is offering a number of plotters with automatic paper feed, the need for some form of spooled plotter operation will manifest itself. Would not it be nice if standard graphics packages like EZCHART, DSG and HPDRAW could spool their output to a plotter device, instead of addressing the plotters as 'hot' devices.

* In view of the great increase in the number of HP microcomputers that are used to also communicate with HP3000 systems the following proposition could be of interest. Using the alternative spooling system as described in this paper it is already possible to use a microcomputer with attached printer as a remote spooled printer station. However, it would be nice to also have the possibility to submit files from the microcomputer directly to the HP3000 spooler to be printed on one of the other spooled printers.

* When the new spooling system is built, the problem of store-and-forward of spoolfiles is licked. Why not extend this method to other types of files, like MPE-files, KSAM-files and perhaps even to IMAGE-files. It is an interesting thought to envisage a network file transfer system, transporting any HP3000 file to any place in a network using any inter-machine connection.

Biography.

Jelle Grim worked for the same company, the contractor Royal Boskalis Westminster from 1966 to April 1984. Starting as a civil engineer in the technical area he almost immediately switched over to the computer section. The Boskalis automation between 1968 and 1984 changed from in-house IBM S/3, through external data processing at a CDC service bureau using local Datapoint mini's, to in-house HP3000 equipment from 1978 onwards. When Jelle left Boskalis he was Information Network Manager in charge of a dual HP3000 network serving approximately 200 terminals and microcomputers both in Holland and abroad.
In April 1984 Jelle and his partner Rene van Geesbergen together founded Holland House, a company specializing in HP3000 system management consultancy and software products.
Jelle is secretary of the Dutch Users Group HP3000 (DUG) and a member of the Amsterdam 1985 Host Committee.

THE WORLD ACCORDING TO GREP*
(IS UNIX** IN YOUR FUTURE?)

Amy Heidner, John Fluke Mfg. Co., Everett, WA, USA
Dennis Heidner, Boeing Aerospace Co., Kent, WA, USA

Abstract

The UNIX operating system has been a favorite of programmers of Digital Equipment Corporation (DEC) computers for some time. More recently, a number of UNIX-like operating systems have become available for various other non-DEC mini's and micro-computers. Even HP has released its version, HP-UX, for the 9000 series workstations and (probably) the forthcoming SPECTRUM machine. With the increasing emphasis on UNIX and UNIX-like systems, HP3000 users should be aware of some of the major characteristics of that operating system. This paper discusses UNIX and MPE, showing the strengths and weaknesses of each operating system from a user's point of view.

* A UNIX search utility program.
** UNIX is a trademark of Bell Laboratories.

Introduction

Every experienced computer user has definite opinions about operating systems, which usually can be boiled down to enthusiasm for their own familiar setup, and suspicion of most others. UNIX has been a favorite of programmers and users of DEC computers, and is gaining popularity among programmers and users of other machines as well. Although it isn't available on the HP3000, it is an important trend of which HP3000 users should be cognizant.

UNIX and MPE sprang from entirely different backgrounds. UNIX is the child of a programmers' environment, developed to make their tasks easier. MPE grew from a security-conscious business environment. Thus early versions of the two represent roughly opposite ends of the spectrum of software philosophy. However under the real world pressure of computer users, the two are moving toward a common center, UNIX acquiring some facilities required by business applications and MPE adopting features to facilitate program development.

Some of the confusion about UNIX is due to the fact that there is no single UNIX. There are two major versions, one from American Telephone and Telegraph (AT&T) and one from the University of California at Berkeley (often simply called "Berkeley"), several major revisions (releases) of each, plus scores of derivatives of one or the other (e.g., HP-UX). Contrast this with MPE, which was defined and developed by a single manufacturer, HP. There are advantages to each lineage - UNIX has enjoyed a cross-fertilization from the work of many groups while suffering from the lack of cohesiveness and strict documentation standards expected of a single-source product such as MPE.

Notation

UNIX uses some standard notation. "!" is a special character referring to the shell's history mechanism. "$n" refers to argument number n, "*" refers to all, and "$" refers to end or last. The backslash " " preceeding a character negates the special meaning of that character, allowing it to be used in an ordinary sense.

One note about UNIX - it is case sensitive. All commands (with the possible exception of user-defined commands) are lower case. In this article, we have sometimes capitalized for emphasis or clarity (e.g., C-shell, whose command name is "csh").

Origins

UNIX was developed in 1969-70 at AT&T's Bell Laboratories to run on a very small machine. It was derived from the MULTICS operating system developed jointly by MIT and Bell Labs on a General Electric computer. The first version was written in assembly language, but was rewritten in C, a derivative of B, which in turn was a derivative of the BCPL language. UNIX has since grown more capable and more portable, and is gradually shedding anacronisms left over from its days on a 12-bit computer.

Bell Labs began licensing UNIX in 1973, at a modest price to universities and a much higher price to commercial institutions. Bell was prohibited by law from competing in the computer market, so no support was available for the system. The software development group at University of California Berkeley added a number of new features, and began licensing their version. Since the divestiture of AT&T, Bell is supporting their version of UNIX. A more recent development is the announcement of a UNIX operating system (ULTRIX) from DEC; although most UNIX machines are DEC hardware, DEC was not recommending or supporting any operating system except their own. Market pressure, however, has brought about the change.

Despite their radically different philosophical origins, UNIX and MPE do have some commonality. They were among the first operating systems written largely in some higher-level language. [1] The use of a higher-level language coupled with the relatively small amount of memory required for the basic kernel makes UNIX portable to a large number of machines. [2] This portability has led to both widespread use of the operating system and proliferation of versions with non-standard enhancements.

Structure

The UNIX kernel is exactly what its name implies - the very heart of the operating system, containing the intrinsic commands. The shell is the command interpreter, which is what the user "sees" as UNIX. The shell also contains the facilities to configure the environment, pipe or redirect I/O, execute processes in foreground and background, and write and execute shell scripts, which are user-defined commands. Also available to the user and included in the overall UNIX system are utilities such as text processors, compilers, file and string manipulators, source control and file dependency operators, and communications utilities.

The shell may also be invoked like any other program, thus creating a subshell which operates on its own input and output. A feature of UNIX is the ability to specify the shell on a user-by-user basis. Thus some users may elect to use the Bourne shell and others the C-shell. This feature also allows the possibility of providing security by setting special restricted shells for certain users (e.g., contractors).

The history mechanism is a unique feature of UNIX. It allows users to re-execute previously-executed commands, edit and re-execute, or simply see what commands have been done during the session. For example, if we had logged in, editted a file containing this paper, in background mode sent it through the document formatter to the printer, sent mail to a fellow user, checked the line printer queue to see if the document was finished, and then done a history command, we would get the following display on the terminal:

```
1  vi grep.world
2  nroff -me grep.world | lpr &
3  mail jeff
4  lpq
5  history
```

Now if we wanted to edit the file again, we could type

```
!v
```

to re-execute the last command beginning with "v". Alternately, we could have typed

```
!1
```

to re-execute the command with history number 1.

If we must repeat similar commands, or commands with similar arguments, history editing is convenient. For example, to list the files beginning with "fc" plus the file "rcd", we type

```
ls fc* rcd
```

Now if we wish to delete those files, we can substitute the remove command for the list command, keeping the same arguments, by typing

```
^ls^rm
```

If we then type

```
history
```

we get the display

```
1  vi grep.world
2  nroff -me grep.world   lpr &
3  mail jeff
4  lpq
5  history
6  ls fc* rcd
7  rm fc* rcd
```

MPE allows the display, editting, and re-execution of only the last command, using the :REDO command.

Pipes, I/O Redirection, and Filters

Most UNIX programs read and write from a pair of special files, stdin and stdout. (In MPE these are $STDIN AND $STDLIST.) These are ordinarily assigned to the user's terminal. However, they can be reassigned on the command line by use of the special I/O redirection characters. " " indicates the argument following it is the replacement for stdin, and " " indicates the replacement for stdout. For instance, an engineering application program called SPICE requires input from a specified input file or stdin, and puts its output on stdout. The output may be redirected by the user in the following manner:

```
spice inputfile > outputfile
```

Filters are programs which read from stdin and write to stdout.

MPE provides message files (IPC) which serve the same function as pipes; the system-defined files such as $STDIN and $STDLIST can be redirected through the use of these files. For example:

```
:FILE PIPEIN=PIPEIN;CODE=MSG
:FILE PIPEOUT=PIPEOUT;CODE=MSG
:RUN XPROG; STDIN=*PIPEIN;STDLIST=*PIPEOUT
```

Pipes are mechanisms whereby the output of one program is sent directly to the input of another. Both programs run concurrently; the first will die when it finishes, then the second when it has completed. For instance, the output of program SPICE may be piped to a terminal paging program, instead of being directed to stdin (the terminal, where it will go whizzing by) or redirected to a file. This is achieved by:

```
spice inputfile | more
```

Directories, Files and Devices

Like MPE, UNIX sees everything as files - directories are only a specific type of file, and I/O devices are merely "special" files. Devices such as printers and terminal monitors are treated like write-only files, while keyboards are read-only files. This greatly simplifies redirection of program input and output.

Under the main directory of a UNIX account, files and subdirectories may be created; subdirectories in turn may contain other files and subdirectories. Each directory has at least two entries "..", referring to its parent, and ".", referring to itself.

A simple filename references a file in the current directory; files in subdirectories may be accessed by a longer format of the filename, subdirectoryname/filename. The slash is the separator, and subdirectory names may be prepended for as many levels as necessary. To change the current directory, one connects to a different one using the "cd", change directory, command. Figure 1 shows a sample UNIX directory tree. The number of layers is determined by each individual user.

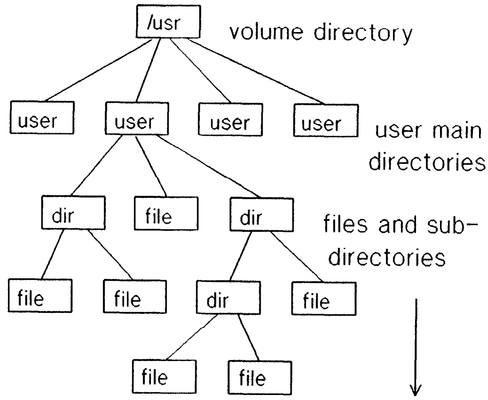


Figure 1. Sample UNIX Directory Tree

Groups in UNIX are collections of users, such as all the users working on one project. Groups are not necessarily permanent; when that project ends and users begin new tasks, they may be assigned to be members of new groups.

Filenames in UNIX can be specified with wildcard characters or regular expressions, as well as explicitly spelled out. For example,

```
mc*         matches all filenames in the current directory
                 starting with "mc"
rnl/mc*k    matches all filenames in the rnl subdirectory
                 starting with "mc" and ending with "k", no
                 matter how long
mc[1ab]      matches "mc1", "mca", and "mcb" in the current
                 directory
../mc?       matches any 3-character filename in the parent
                 directory starting with "mc"
mc[a-d]1     matches "mca1", "mcb1", "mcc1", and "mcd1" in the
                 current directory
```

There are no restrictions on extensions; they may be any length, or quantity.

MPE does not provide a tree-like directory such as available under UNIX. Instead MPE has one common directory, with files and users assigned to accounts and file groups. Users locate files by using the :LISTF command; like UNIX, wildcards may be used to select files. For instance,

```
:LISTF Q@.@.@,2
```

will locate all files on the system that begin with "Q".

Security

Each user of the UNIX operating system has a separate account and password. These passwords are stored in encrypted form in a system password file, and are initially assigned by the superuser, but a user may subsequently alter the password on the account. The right to alter the password is verified by demanding that the person requesting the change know the present password.

Under the main directory of the account, a user may build files and subdirectories, which in turn may contain other files and subdirectories. Each UNIX file has a protection mode, which controls read, write, and execute access for that file for each of 3 classes of users -- the owner, the group, and the world (everyone else). When a file is created it takes a default protection mode set by the superuser (system manager); this is often mode 644 (octal), allowing read and write by owner, read by group, and read by world. However, the mode of any individual file may be changed by its owner, using the chmod command. For instance, to change the file containing this paper to read-only for all classes of users, one would use

```
chmod -w grep.world
```

MPE provides security at account, group, and file levels as well as an optional lockword. Users are classified into seven different categories: system manager, account manager, account librarian, group librarian, account user, group user, and file creator. The MPE file system allows the user to build a matrix describing who can read, write, lock, append, execute, or save files in specific accounts and groups.

For example, the security on a typical file in the PUBLIC group of the system account is:

```
R,S:ANY;W,A,L:AC
```

Environment

UNIX allows the user to tailor the environment to suit by setting some variables, defining aliases (synonyms or shorthand) for commands or groups of commands, and writing shell scripts for user-defined commands. For example, an alias definition is shown below.

```
alias printdbl "cat !*   awk ' print $0;print '   lpr &
```

It defines an alias "printdbl" which operates in background mode to send a double-spaced version of the files specified as arguments to the line printer; the files themselves are unchanged.

Two shell scripts are shown below, a simple one which prints a formatted phone list, and a more complex one which runs a series of simulation programs, does some processing of the output, and prints the result. The lines (except for the first) beginning with "#" are comments.

```
#! /bin/csh -f
#'newphone'
#this  shell script takes a file "phone" of name and phone number
#pairs, residing in the user's main directory,
#sorts it alphabetically,  saves the sorted file and prints it in
#two columns on a 132-column printer
sort -df /phone -o /phone
pr -w132 -2 /phone | lpr
```

```
#! /bin/csh -f
```

```
# 'do circuitname [circuitname] ...'
#this script extracts a schematic from each specified circuit
#layout, prints some of the resulting files, & if the .cap output
#file is non-zero, invokes the shell script "listcap" to do some
#text processing on that file and print the results.
cd ca
@i = 1
while ($i <= $#argv)
     rm $argv[$i].cif
     echo "ed $argv[$i]; cif -p 100; q" | xcaesar -n
     mextra -o $argv[$i]
     rm $argv[$i].cif
     print
     mv $argv[$i].nodes $argv[$i].log $argv[$i].al ../rnl
     mv $argv[$i].sim ../rnl
     cd ../rnl
     presim $argv[$i].sim $argv[$i].bin ../cmos.config
          -c$argv[$i].cap,0
     if (-e $argv[$i].cap) then
          if ! (-z $argv[$i].cap) listcap $argv[$i]
          endif
     endif
     @i = i + 1
end
```

MPE also provides the capability to write "programs" in the command language. For more information on how to use MPE as a program language, see "MPE Programming" by Eugene Volokh, presented at the 1983 HPIUG in Montreal, Canada. [3]

Communication

Several types of communication are included in the UNIX system. The mail program allows users to send messages to other users. These messages are queued until the recipient reads them. If there is mail awaiting them, users are notified when they log on.

Write allows a user to send a message directly to another user's terminal. This is similar to MPE's :TELL command.

Cu allows a user to force the computer to call up another machine on which the user has an account. Cu will handle file transfers in both directions, as well as execution of commands on either machine. The command sequence

```
cu phonenumber ; login
```

is equivalent to the MPE commands:

```
:DSLINE
:REMOTE HELLO
```

Uucp (UNIX-to-UNIX copy program) allows machines to call each other to pass information.

Text processing

UNIX comes with a standard text editor, ex (a superset of the editor ed), which has a visual (screen-oriented) entry-point called vi. The post-processor-style document formatter nroff is provided, along with some special-purpose formatters for tables, equations, etc. In addition, there are a number of text-processing filters, including sed, a stream editor, and awk, a string manipulator. MPE provides only the line-oriented EDITOR.

Comparison of Commands, Utilities, Etc.

Appendices A and B show a comparison of some commonly used commands, with one column for each operating system. These translations are for major function only, as there are nuances in the operation of each system which give it a unique flavor. We have tried to restrict this list to "vanilla" systems of each (i.e., MPE's Fundamental Operating System (FOS) and UNIX as supplied directly from AT&T or Berkeley), although there are many common, useful additions available. The appendices are split roughly by MPE's notion of commands and utilities; there isn't such a division in UNIX. All appear as commands to the user although they may be translated by the shell to a string of commands to the kernel or to a separate program.

User-Interface Issues

It has been said that UNIX is not very user-friendly, and there is a great deal of truth in this assertion. Most UNIX commands are short, abbreviations or acronyms. This is great for the experienced user who hates to type more than necessary, but frustrating to the novice or occasional user. MPE commands, on the other hand, tend to be verbose, e.g., :SHOWJOB or :SHOWLOGSTATUS.

For the occasional user or "permanent novice", someone who does one type of task and has no need (or desire) to learn more about the operating system, the drawbacks are not so great. This is due to the chameleon capabilities of the shell; UNIX can be made to look like practically anything desired, including other machines more familiar to the user. This adaptation can be done by an experienced, but not necessarily expert, user. A similar function, called user-definable commands, is available to MPE users.

What about windows? Windows are not an integral part of UNIX, but there are an increasing number of programs which can be used as overlays to provide windowing. Many UNIX-based engineering workstations have integrated the windowing capability.

Summary

UNIX and MPE are operating systems developed in very different environments, and yet they are not as dissimilar as one might assume. Each has strengths and weaknesses; evolution in real-world use is forcing them to progress toward a common center. HP3000 programmers and users will probably see more UNIX-like features in future releases of MPE and more vendor programs to add other UNIX-style tools.

Notes and References

[1] In fact, the languages SPL and C are similar in some aspects, both are structured languages. Many of the intrinsics are also similar, due to the fact that the authors of both were familiar with MULTICS, and both languages were developed in roughly the same time period. (MPE was released in the early 1970's.)

[2] Portability is relative; it does not imply a trivial migration process. Basically something is portable if it "can be transported with less effort than writing it anew." Jalics & Heines, "Transport-

ing a Portable Operating System: UNIX to an IBM Minicomputer", Communications of the ACM, Dec. 1983, p.1070.

[3] Volokh, Eugene; "MPE Programming"; Proceedings, 1983 HPIUG, Volume 2, pp.90-1 through 90-10.

Bibliography

Bell Telephone Laboratories; UNIX Programmer's Manual, Volumes 1 and 2; Holt, Rhinehart & Winston; 1983.

Halamka, John; Real World UNIX; Sybex; 1984.

Ritchie, Dennis and Thompson, Ken; "The UNIX Time-Sharing System"; Communications of the ACM; July 1974; pp.365-375. (Reprinted Jan. 1983; pp.84-89.)

Yates, Jean and Thomas, Rebecca; A User Guide to the UNIX System; McGraw-Hill; 1982.

Appendix A Translation of Some Common Commands

MPE	UNIX	Function
:ABORT	kill	abort a program
:ALTACCT,:ALTUSER	ed,mkdir,chown,passwd	set up a new user, change password
:ALTSEC	chmod	change read/write access
:APL	no equivalent	language APL
:BASIC	bas	BASIC language
:BUILD	any UNIX command with output to a file	create a new file
:BYE	logout	log off
:COBOL	no equivalent	COBOL language
:COMMENT	#	comment; no action
:CONTINUE		
:DEBUG	abd	debugger
:DSLINE	cu	connect to (call up) another computer
:DSTAT	df	status of disc drives
:EDITOR	ex (ed) vi nroff	line-oriented editor screen-oriented editor text formatter (post processor for editor files
:FCOPY	cp	file copy routine
:FILE	< > | (pipe)	redirect input redirect output send output of one program directly to another
:FORTRAN	f77	Fortran language compiler
:HELLO	login	log on
:HELP	man apropos	interactive help (on-line manual pages) lists manual pages applicable to a keyword
:IF,:ELSE,:ENDIF	if() then, else, endif	conditionally perform statements

:JOB/:STREAM	&	submit work to background
:LISTF	ls	list files
:NEWACCT/NEWUSER	see :ALTACCT	
:PREP	ld	link program files
:PURGE	rm	purge (remove) a file
:REDO	!!	redo last command line note: UNIX history mechanism is much more powerful
:RELEASE	chmod	unsecure a file
:REMOTE HELLO (after :DSLINE)	login (after cu)	log on to another (remote) computer
:RENAME	mv	rename (move) a file
:RESTORE	restor	restore backup file
:RPG	no equivalent	language RPG
:RUN	not applicable	run a program; UNIX executable files may be run simply by typing their names
:SAVE	mv	save a file
:SECURE	chmod	re-establish security rules for a file
:SEGMENTER	make, ld	library maintenance program
:SETCATALOG	shell configura- tion file	establish user-defined commands
:SHOWJOB	ps -au	show all jobs running on system (ps can also show all jobs of a single user, including background jobs
:SHOWME	finger	show information about the user
:SPL	cc	system programming language
:STORE	dump	make a backup copy
:SYSDUMP	dump, dumpdir	backup system; change configuration
:TELL	write	send a message to another user who is currently logged on

Appendix B Translation of Some Common Utilities

MPE	UNIX	Function
no equivalent	awk	string manipulation, pattern scanning
no equivalent	calendar	send reminder mail/memo to user at specified date/time
use EDITOR or FCOPY	cat	type files; multiple files are catenated
no equivalent (must use FCOPY)	chown, chgrp	change creator (owner) or group of file
FCOPY (;COMPARE)	cmp	compare two files
FCOPY	cp	file copy routine
no equivalent	crypt	encrypt, decrypt file
SHUTDOWN, WARMSTART (only supported method)	date	set date, time
no equivalent	dc	desk calculator
FCOPY	dd	copy and convert files
DSTAT, FREE2.PUB.SYS	df	show disk volumes, devices and free space
no equivalent	eqn; neqn; checkeq	typeset (format) mathematics
no equivalent	factor	factor number into primes
no equivalent	graph; plot	draw graphs
no equivalent	grep; fgrep; egrep	pattern search a file or stdin
RECOVER LOST DISK SPACE	icheck; dcheck	verify and check disk free space and directories
no equivalent	iostat	report I/O status
IMAGE/QUERY	join	relational database operator
no equivalent	lex	generates programs to perform lexical analysis
contained in SPL compiler	lint	verifies and detects bugs in C programs
no equivalent	m4	macro pre-processor, front end

		for cc, f77, etc.
no equivalent	mail	send & receive mail
compilers have masterfile, change-file capabilities (TOOLSET from HP)	make	update dependent files based on changes to master file
:JOB ... PRI=ES (batch only)	nice	process command at a low priority
:ALTUSER (requires AM capability)	passwd	change user password
FCOPY, EDITOR	pr	print & paginate a file, add headers (default = filename, print date, time)
SYSINFO, TUNER, SOO (from CSL)	pstat	system status information
no equivalent	ptt	generate a permutation index
no equivalent	pubindex	make a bibliographic index
no equivalent	roff; nroff; troff	text formatter
:REPORT	sa, arctan	report system usage
:SEGMENTER	size	show object program size
SORT	sort	sort file
no equivalent	spell	check spelling
FCOPY	split	break files into smaller pieces
compilers do not normally include this information, you must explicitly request it with ;FPMAP	strip	remove symbol table and relocation information from object file
unfortunately there is no equivalent, not even on the Workstation configuration from HP	stty; tabs; termcap file	specify terminal capabilities, speed, cursor addressing, etc. UNIX programs can use any terminal which can be described by its capabilities
not applicable	sync	force an update to disk of directory cache domain
FCOPY, EDITOR	tail	extract specified number of

		lines from end of file
no equivalent	tbl	format tables
no equivalent	tc	photo typesetter simulator
no equivalent except using FCOPY twice	tee	intercept I/O going through a pipe and make another copy
:SHOWTIME (accurate to 1-2 minutes)	time (accurate to 1/60 second)	time a command
global rins	wait	wait for a process or job to to complete
:TELL, :WARN	write; wall	send a message to a user, all users

About the Authors:

Amy Heidner is a licensed professional engineer with a degree in electrical engineering. Ms. Heidner has programmed on various computers including the HP2100 series, forerunners of the present HP1000 computer family, and a number of computers made by the Digital Equipment Corporation (DEC). Ms. Heidner is an experienced UNIX user. She was involved in the editing of "The IMAGE/3000 Handbook", published first quarter 1984 by WordWare. Ms. Heidner is currently employed as a design engineer at John Fluke Manufacturing Company, makers of electronic test equipment, including the 1720 series of instrument controllers with touch-sensitive screens.

Dennis Heidner received the BSEE degree from Montana State University, Bozeman, Montana. He joined the Boeing Aerospace Company in 1978 on a special project to study techniques for management of general-purpose electronic test equipment. Based in part on that review of management methods used throughout industry, the Boeing Aerospace Company Test Equipment Management group received approval to buy an HP3000 computer system. Mr. Heidner was responsible for the system requirements, planning, design, and program implementation. In May 1980, the Test Equipment Inventory Management System (TEIMS) became functional. Mr. Heidner has written "Transaction Logging and Its Uses", presented at the 1982 HPIUG. He co-authored two papers, "Transaction Logging Tips" and "IMAGE/3000 Performance Planning and Testing", which were presented at the 1983 HPIUG in Montreal. He presented the paper "Disaster Planning and Recovery" at the 1984 HPIUG conference in Anaheim. Mr. Heidner is a co-author of "The IMAGE/3000 Handbook", published by WordWare, Seattle, Washington. He has also written technical articles which have been published in several magazines.

Structured Tuning: Man's Interface to the Machine's Performance Components

Mark Muntean, Performance Center Manager
Sam Boles, Member Technical Staff

HEWLETT PACKARD

You've seen the benefits of Structurism: Structured Design, Structured Programming, Structured Implementation have enabled significant gains in the quality and timeliness of computer systems. Here you'll see the Top-Down principle extended to the perennial subject of System Performance in the form of Structured Tuning.

You'll get an update on the tool set used to examine the performance components of your system and to interface with the comprehensive performance instrumentation that has evolved as an integral part of MPE. Structured Tuning steps thru the several Layers of your system, starting at the topmost global Layers, and maps the tools to performance-sensitive factors like CPU, main memory and disc I/O.

You'll see the interface in action in performance benchmarking and in dissecting the biopsies from performance-troubled production systems. And you'll see the results of this exercise lead naturally to the next step of predictive modeling for capacity planning and early warning on potential bottlenecks.

. . . if you solve the wrong problem . . . your real problem doesn't go away . . .

Once in another time and another place, there was a man who had two problems. One of his problems was that he wanted to play a tune on his fiddle. He thought if he solved one of his problems, then he'd have one less problem. So he played a tune on his fiddle, and, sure enough, he got ***that*** problem solved.

However, in the meantime, ***Rome burned***.

He'd solved a problem -- but it was the ***wrong problem***.

We all know there's no such thing as a ***free lunch***. Maybe there's no such thing as a ***free anything***. To do anything has a ***cost***. In fact, it has ***two costs***. One cost is what it costs you to do it. The other cost is what it costs you not to do something else. You call that the ***opportunity cost***.

A lot of the time it's the opportunity cost that really hurts. You've got a limited budget and limited resources. If you use your limited (and expensive) performance engineering resources to solve the ***wrong problem***, the opportunity cost may be that your ***real problem*** doesn't go away.

Structured Tuning is a methodology that helps you identify the ***right problem*** to work on. Structured Tuning decomposes the complexity of a computer system into a series of hierarchical layers that can be traversed from the top down.

. . . Structured Tuning . . . a top-down discipline for global perspective . . .

This top-down discipline works against ***Premature Optimization***, one of the costliest parts of the Wrong Problem Syndrome in performance tuning. It forces you to look at the ***global issues first***, before getting down to the local issues. If you start at the top and look at the layers in order, you see things in the context of the "Big Picture." This helps you identify the high-yield projects and get your ***priorities straight***.

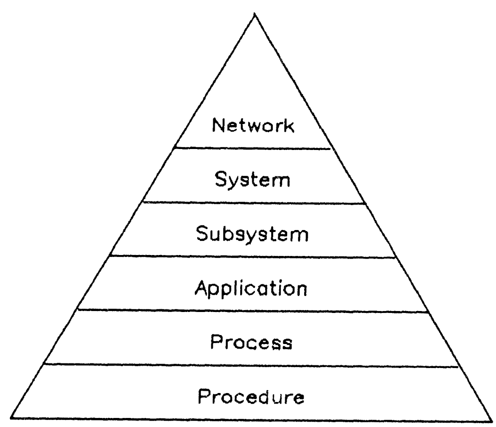
Structured Tuning does for performance what Structured Implementation does for a project: You identify the main line and build a prototype, fast and cheap. You try it. If it doesn't work because the technology can't stand the strain of the application, you throw it away. If it's not what the user needs, you throw it away.

You use the prototype to solve the ***right problem***: ***Can it be done? Should it be done?*** . . . and if either question yields a negative, you throw away less code and minimize your losses. If the prototype registers positive, you get into the inevitable rewrite faster and on firmer ground. The Structurism of Structured Implementation helps you ask the ***first questions first***; and the same thing happens in Structured Tuning.

In a step-wise traversal of the hierarchy, if you see at the System Layer that you're bottlenecked on disc I/O, you don't jump right in and rewrite that simulation algorithm (that you ***know*** is killing you) to cut its CPU consumption in half. That might not be a bad idea . . . eventually . . . but not now. This discipline has Pareto's 80-20 Rule built into it, and gives you a first-order prioritization of your tasks.

. . . topology . . . tools . . . techniques . . . for Structured Tuning . . .

A dissection of the subject that seems to yield a workable structure for Structured Tuning has the Network as its top Layer and decomposes down thru System and Program Layers to the Procedure Layer:



The Structure of Structured Tuning

The "bottom line" is ***productivity***. The metrics we use are transaction throughput ("how many orders can I enter in a hour?") and response time ("when I key in 'how much is 2 + 2' how soon do I get the answer after I touch RETURN?")

To get these metrics in this range of layers there's a range of tools. Some are software products you can buy. Some are contributed. Some are part of your HP Performance Specialist's bag of tricks. These tools for the most part interact directly or indirectly with the comprehensive performance measurement instrumentation that is part of MPE.

A key measurement tool in the HP3000 domain is ***OPT*** (On-line Performance Tool). This tool is a supported product and provides a variable-interval, variable-level window into the system.

Complementing OPT is a consulting tool ***MPEDCP*** (MPE Data Collection Program) which collects performance statistics similar to those in OPT, but with a somewhat different focus and format. ***MPEDRP*** is its ancillary data reduction program.

For finer granularity at the file/process level, ***IODCP*** is a consulting tool that gives statistics to help with mid-range and low-level tuning. ***IODRP*** is its ancillary data reduction program.

At the Process and Procedure Layer the ***APS*** (Application Program Sampler) system provides insight into CPU utilization down to the level of lines of code.

In addition to these tools, there are MPE and subsystem commands and utilities that enable views of how the system's running and parameter settings, eg SHOWCACHE, DSCONTROL, DSDUMP.

Supported Products

OPT
APS
SHOWCACHE, DSCONTROL, DSDUMP

Other Tools

MPEDCP
MPEDRP
IODCP
IODRP

. . . models of the real world . . . capacity planning . . . performance management . . .

Complementing the performance monitoring tools are the models of the "real world" that can give valuable support to Capacity and Configuration Planning and Performance Management: ***Benchmarking***, ***Simulation***, ***Analytic Modeling***.

While the monitoring tools tend to be reactive to problems that already exist and need to be corrected, models are more predictive and proactive. With predictive power, models can provide the information necessary to make a good decision at the beginning that can prevent a problem rather than fixing it after it's happened. This ***Ounce of Prevention Mode*** can save money, time and human resources.

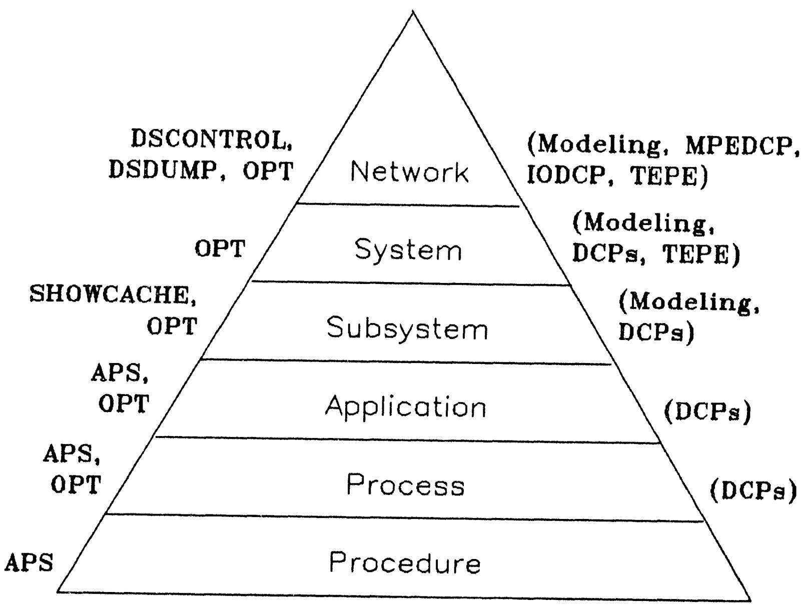
In ***Benchmarking***, the first degree of abstraction from the "real world," we model the range of system and workload configurations that are proposed or perceived to be a correct representation of the current situation and/or the future. Being a model only a step away from the real thing makes Benchmarking tend to be very costly in time and other scarce resources.

For benchmarking, ***TEPE*** (Terminal Emulator and Performance Evaluator) provides a direct hardwire connect from the driver system to the System Under Test. Guided by terminal "scripts" TEPE sends bit-serial messages that look to the System Under Test like inputs from CRT's.

The second degree of abstraction, ***Simulation***, models a trace of the "real world"; for example, a detailed sample of disc I/O traffic, taken from a production system is modeled to predict its behavior in a proposed environment, eg a system where disc domains are cached in main memory.

. . . analytic modeling . . . trend analysis . . . derivation of load factors and other parameters . . .

The third degree of abstraction from the "real world," the ***Analytic Model***, models algorithmically the functions of the "real world"; for example, a queueing-theory based central server network that includes a CPU of variable speed and a variable number of discs with variable speeds and variable user classes and variable think times and variable . . . and variable . . . This model is typically the fastest and cheapest, can be the most flexible, is sometimes the only feasible, but often is the most difficult in some ways since it requires us to ***understand our data and programs and systems in great depth***.



The Mapping of Tools in Structured Tuning

... system components ... CPU ... main memory ... disc caching ...

The ***Central Processing Unit*** is a key variable in the Structured Tuning Methodology, but in the typical commercial job mix it's usually not where you get the primary performance bottlenecks. In the current HP3000 line you have a .25 MIPS processor in the Series 37, a .4 MIPS processor in the 4X series and a 1.0 MIPS processor in the 6X series for vertical flexibility. For horizontal flexibility, the HP DSN (Distributed Systems Network) provides a variety of 3000-to-3000, 1000-to-3000, and 3000-to-other linkage possibilities.

Any performance tuning methodology today in the mid-range to high-end of the HP3000 must examine the implications of ***Disc Caching***. This may be the most significant performance contribution in the HP3000 family since the 1 MIPS processor; it makes something beyond a brute-force raw-horsepower performance improvement. Disc Caching needs 20-40% of the CPU in order to give you really substantial (2X, 3X) performance gains.

If the necessary CPU is not available, Disc Caching can degrade performance.

In the HP3000 family you've got this range of ***main memory*** available:

Series	Min	Max	Write Speed
68	2Mb	8Mb	375 ns
48	1Mb	4Mb	530 ns
42	0.5Mb	3Mb	530 ns
39	0.5Mb	3Mb	530 ns

On the high-end, in the 6X series, you have an 8K byte high-speed (ECL RAM) memory cache that has a 95% hit rate. This gives you an effective 145 nsec memory read speed.

Let's put ***main memory in the Disc Cache perspective***: The essence of Disc Caching is to let you get at your disc data at main memory speeds rather than disc access speeds. If you compare the two, you find that accessing main memory is something like 10,000X to 50,000X faster than your average disc memory. Since disc I/O makes up such a large part of commercial data processing, that kind of data access improvement can really pay off in better response time and throughput rates. If your job mix has a 70% cache hit rate (as is often the case in commercial applications, you can pay the 3 or 4 msec price for Caching, save 30 or so msec on 70% of your I/O, and get a good return on your investment. To give Disc Caching room to work, you need to give it 1 or 2 Mbytes of main memory on the high-end machines to get the 2X and 3X performance gains it's capable of producing.

If Disc Caching is enabled on a system already under memory pressure or has local optimization that conflicts with the global effect of Disc Caching, it may degrade performance.

... disc I/O ... PEP ... disc caching ... other system components ...

Disc I/O has been the perennial performance bottleneck in the typical HP3000 job mix -- or ***so it seems***. With the arrival of the 4X and 6X machines, the old 30 I/O's per second max has gone away. With multiple master drives (each with its own controller), on multiple GIC's (General I/O Controller) (each with a 1 Mbyte bandwidth), on multiple IMB's (InterModule Bus) (each with a 3 Mbyte bandwidth), we can sustain 60 to 120 or more disc I/O's per second.

But we don't.

Why?

Because ***we don't ask for it***.

What we usually do is ***port our jobs 1-for-1*** from the Series III to a 3x, 4x or 6x, configure in 4 or 5 times as many terminals as we had before to go against that single IMAGE database that we built 4 years ago when we were 1/4 the size we are today. And we sit there, queued up on a single-threaded Database Control Block with the CPU only 50% Busy and a disc subsystem capable of 100+ I/O's a second being asked to do 50 I/O's a second.

For the high-end 3000 installation, one of the important disc drive innovations in recent years is the 400 Mbyte HP7935.

When we first heard about the 7935 we were really impressed: ***for a small percent more than you pay for a 7925 you could get more than 300% of the 7925 capacity***. Then we ran some benchmarks and found that for the capacity gain you paid a 15-20% performance penalty.

Then came ***PEP*** (Performance Enhancement Project.) The engineers in Boise tweaked ***better than 15% improvement*** into the 7935 and got it within noise-level of the 7925 performance.

The PEP microcode structure gain coupled with Disc Caching, Buffer Prefill and RPS (Rotational Position Sensing) makes the on-line storage economy of our 400 Mbyte disc drives (7933, 7935) a viable solution for the performance-sensitive customer whose "results must be measured by performance."

But remember: Disc Caching needs file locality, a good (3+:1) Read:Write ratio and a heavy disc I/O load to produce the 2X and 3X performance gain it's capable of.

In addition to the Big Three -- CPU, main memory, disc I/O -- your system performance can be influenced by factors like:

Datacomm line speeds for terminals and distributed systems
Terminal/Workstation speed and intelligence
Mag tape speed/density (especially in back-ups)
Printer capacity
System tables

Structured Tuning looks at these components as they emerge at the various Layers of the hierarchy and assigns them a direction and magnitude appropriate to their station in life. This is usually very application- or installation-dependent.

. . . Structured Tuning the Network Layer . . . the System Layer . . .

Now you've seen the topology of Structured Tuning, the objects that it deals with and some of the tools that it uses. Let's take a look at some examples of ***Structured Tuning in action***.

In some ways, the ***Network Layer*** may be the toughest. You've got a wide range of configurations, line speeds, ranges, types, sharing, simultaneity, concurrency, costs, metrics, vendor compatibility issues.

From the aspect of Structured Tuning, there are some rules of thumb that seem to work. Aim at ***doing fewer things bigger***. From the performance aspect, one user with a large record is better than multiple users with several small records. One thing you can check for is a small overrun on your record size; for example, if you configure 1024, but send 1026 you may generate a lot of continuation records.

If at the System Layer you find a CPU bottleneck that traces to the Application Layer where you find that the heavy crunch of business graphics is consuming your CPU at the expense of general response time, consider relieving the CPU pressure by distributing the load. For example, an RS232 link to an HP9000 could enable the offloading of the heavy-duty processing to a high-performance, high resolution work-station solution. The artwork database in a convenient intermediate form could then be uploaded to the HP3000 for integration with text and output to the laser printer.

Notice the Structured Tuning Methodology at work here: it iterates thru lower Layers for details characterizing problems detected at a higher Layer, and may percolate back up to the highest Layer for the solution.

A good starting point at the ***System Layer*** is to look into the system thru the window that is OPT (On-line Performance Tool). At the global level, OPT shows you CPU, main memory and Disc I/O utilization on a variable interval basis, and CPU and Disc I/O utilization on a cumulative (multi-interval) basis.

Here's an example of the OPT global report (clipped on the right for printing convenience):

```
      RESOURCE USAGE DISPLAY      HP32238X.VV.16  OPT/3000
(C) HEWLETT-PACKARD COMPANY 1979, 1980                          SEBDCOL2 S64 2MB CACHE ON
   CURRENT INTERVAL:    66.5 SECONDS          OVERALL INTERVAL:    11.1 MINUTES
ACTIVITY IN CURRENT INTERVAL
                           10         20         30         40         50         60
      MEMORY USAGE M----MC-----------CS----SD--DK-----------------------------------

         CPU STATE B-----------------------------------------------------BP----------

                           10         20         30         40         50         60
 DISC I/O ACTIVITY SSK-------------------------------------K

---------------------------------------------------------------------------------------
ACTIVITY OVER ALL INTERVALS
                           10         20         30         40         50         60
         CPU STATE B---------------------------------------------------------BP-------

                           10         20         30         40         50         60
 DISC I/O ACTIVITY SK-----------------------------------------K

MEMORY USAGE LEGEND:
  M  Resident MPE
  C  Code segments
  S  Stack segments
  D  Data segments
  K  Cached disc domains

CPU STATE LEGEND:
  B  Busy on processes
  P  Paused for user and/or memory management disc I/O
  I  Idle
  G  Garbage collection
  O  Memory allocation and ICS overhead

DISC I/O ACTIVITY LEGEND:
  U  User disc I/O
  S  Segment management  I/O
  K  Cached I/O
```

Here we may see excess CPU, main memory, disc I/O capacity, Pause for I/O.

If we have a disc I/O intensive load, we can drop to the next Layer in our Structure, the ***Subsystem Layer***, and take a look at the disc subsystem.

... Structured Tuning the Subsystem Layer ... the Application Layer ...

At the ***Subsystem Layer*** we can collect a trace of the disc I/O traffic and simulate what its behavior would be with Disc Caching. We can look at the read:write ratios. We can see if the traffic is sequential or direct. We can look at file access locality in time and space. And we can run the trace thru the Disc Caching Simulator to see what kind of hit ratios we'd get if we enabled Caching and gave it a Mbyte or so of memory to work with.

If it looks promising, we can put in a Meg of Main, enable Disc Caching and take a look (with an MPE-level command) at what it does for performance:

:SHOWCACHE

DISC LDEV	CACHE REQUESTS	READ HIT%	WRITE HIT%	READ%	PROCESS STOPS	K-BYTES	% OF MEMORY	CACHE DOMAINS
1	16436	70	93	73	3776	1434	23	252
2	17809	69	95	66	3673	2319	38	358
Total	34245	70	94	69	7449	3753	62	610

```
Data overhead is 158K bytes.
Sequential fetch quantum is 96 sectors.
Random fetch quantum is 32 sectors.
```

We see we get fewer process stops (the Cache Manager does not interrupt on a cache hit for a read), get bigger and more level physical I/O's, put to use the idle CPU and main memory and, by happy coincidence, double the throughput, cutting the wall time about in half.

For the next case study of Structured Tuning in action, let's look at the ***Application Layer*** at what makes it feasible to print multiple copies of the paper you're reading on an HP2688 Laser Printer. Let's say we have a technical publication that we've typeset on the laser printer, using 3 forms and 12 fonts and we decide to print 50 copies of it on the system. We do a TDP (Text and Document Processor) "Final" to get TDP to format it, deferring the SPOOL file. Then we do a "HEADOFF" (to save a tree), set OUTFENCE up so we don't get alien output commingled with our 50 copies. We then do an "ALTSPOOLFILE #Onnn; COPIES=50" and set the priority over the fence.

It's a 6-pager, so we go away and decide we'll come back in ((6 x 50)/12) minutes. (We've got 50 copies of a 6-page document running on a 12 page-per-minute laser printer). We come back in about a half hour, smile pleasantly as we walk through the scowling crowd waiting impatiently around their/our laser printer. We reach for our 50 copies and find we have only 30, and it's still running.

We pull out our Structured Tuning notes, at the System Layer we run OPT, get the global display, find heavy disc I/O. We go into the Process display, see that the SPOOLER is doing a lot of I/O. We run SPOOK, find 1100 lines of high density environment file at the beginning of our output.

We then observe the laser printer. At the end of each document, as our heavy-duty environment file is downloaded, because of the size of the file, the laser printer stops for the downloading. This cuts our printer performance to under 8 pages a minute.

We smile sheepishly at the irate by-standers, as we set our SPOOL file priority to 1, turn the Headers back on and reset the OUTFENCE to normal.

We then create a STREAM file to:

run SPOOK
text in our SPOOL file
APPEND the environment file once
APPEND 10 copies of the text
  to the environment file
APPEND END to get a new SPOOL file

We run this in background, so no one gets impacted in response time.

When we get the new 10X SPOOL file, we go thru the HEADOFF etc routine we did before, set copies=5, and the printer runs at about 11 pages a minute rather than 8.

Structured Tuning at the Application Layer helps us isolate the problem. Structured Implementation builds this ***ad hoc*** prototype to see if the technology can do the job.

. . . Structured Tuning the Process Layer . . . the Procedure Layer . . .

Now to the ***Process Layer***. Let's say that in our Top-Down Traversal of the Structured Tuning Structure, we establish at the System Layer that we have a performance problem that we trace thru the Subsystem Layer to the Application Layer. And let's say that the application has multiple users running the same program, and we think it's one of the processes of that program that's causing our problem.

We can get into OPT and go to the process display to try to further isolate the problem.

If it turns out that our top-priority performance problem is traceable down thru the hierarchy to a given ***process***, we may want to look at CPU consumption at the ***Procedure Layer*** or even at the granularity of lines of code. APS (Application Program Sampler) can help us here and enable us to pinpoint the sector where the heavy resource consumption is happening. With this focus and granularity, we can have a good calibration on where we can best utilize our limited performance tuning resources.

In the APS CPU Utilization report below, we see that the GEN'CHAR procedure (which does a 2-dimensional raster scan of a character "ROM" within a 2-dimensional scan of a message list) is probably a good place to tune among the 10 procedures in the program -- 8 of which didn't have enough activity to register on the scale.

```
Program Name: SEBPROG.BOLES.CAD -  Segment:   %301 SEG'
Distribution of Direct CPU Time Over Procedures
( 95 Segment Samples =   5.26% Direct of Program Time)
+-------------|---------|---------|---------|---------|---------|+---%---%CUM--
SET'UP'CHAR'ROMDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD    | 45.3   45.3|
|     GEN'CHAR DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD| 54.7   100.|
+-------------|---------|---------|---------|---------|---------|+---%---%CUM--
Minimum bar threshold is   .0%
```

APS CPU Utilization Report

You've seen how the Structured Tuning Methodology is an iterative process that starts at the topmost global Layer. It can help you keep a balanced perspective as you look at performance problems. You've seen some of the tools and how they're applied at the various Layers to zero in on bottlenecks. You've seen how Structured Tuning can help you find and fix the right problem, and at the same time set the stage for Structured Capacity Planning, the next step on the road to better performance.

About the Authors . . . .

Mark Muntean *is the Performance Center Manager at the Hewlett-Packard computer facility in Cupertino, California, with responsibility for performance benchmarks, product characterization, and performance modeling. Since joining HP in 1981, Mark has worked with HP3000 performance analysis and modeling. Prior to HP, Mark was a Hertz Fellow at Stanford University where he received his MS in Physics.*

Sam Boles *is a Member Technical Staff in the Productivity Center at the Hewlett-Packard computer facility in Cupertino, California. With HP since 1976, Sam's computer experience started back in the AUTOCODER days of the 1401/1410, migrated thru the 360/370 era, and now focuses on networking HP productivity technology. Sam received his MS at UCLA in Information Systems.*

sebiug38 1445 27dec84

HP LABS SYSTEMS PERFORMANCE EVALUATION PROJECT

Paul Primmer
HEWLETT-PACKARD, Palo Alto, USA

SUMMARY

This paper will cover the ongoing Systems Performance Evaluation Project (SPEP) started at HP LABS in July, 1983. The purpose of the project is to collect performance profiles of HP's current installed customer base. The information gained from this study will be used to enhance current computer systems, and design future computer systems to meet our customers' ever expanding performance requirements. After stating the overall project objective, this paper will follow the evolution of the three major phases of the project. The **collection** phase deals with the decisions of what data to collect and the collection methodology. The **reduction** phase describes the production of a report from the data collected from a single customer. The **summary** phase outlines the procedures used to combine the information from multiple sites into a single summary data base. This paper is intended to serve has a foundation for the conclusions that will be presented at the **HP3000 IUG Amsterdam Conference '85.**

PROJECT OBJECTIVES

The major objective of the SPEP project is to assist in the design and development of future systems and system components through the use of performance analysis technology. Specifically, this objective is being met by measuring system utilization and performance in actual customer environments and developing the tools required to support this effort. Data collected is used to model future system usage and to predict system performance in order to facilitate design and implementation tradeoff decisions. Further, information collected can be used to generate canonical benchmarks which truly emulate our customers' workload in order to characterize the performance of our products before we go to market.

Modeling, which is an abstraction of a system, requires that the mass of data collected be distilled into the essential elements that affect system behavior. Grouped together, these basic elements form a *workload* that consumes resources at the service centers of a system. Examples of service centers are the CPU, memory, the I/O system and software latches and locks. An objective of SPEP is to identify and separate data collected into various classes of *workloads* such as text processing, graphics, data base usage, compiler usage, office products, etc. Further, with survey information provided by the site manager, these *workloads* can be compared and contrasted across different market cells.

SPEP also collects *patterns of customer usage* that can be used to facilitate future product decisions. Questions such as: What percentage of file space is really being used? What percentage of the total terminals configured are active at any given time? How much data is exchanged between a terminal and the system during a transaction? What is the relationship, if any, of the size of a file and the number of I/Os directed to it? Are any of these patterns universal or are they specific to a particular market cell? Answers to these and many more questions are also the goal of SPEP.

COLLECTION PHASE

The collection phase of the project consisted of defining the **types of systems** to be measured, **the customers** to be measured, the **data** to be collected, the **duration of collection** and the **method of collection**. The **types of systems** to be measured were chosen to be the 3000, 1000, and 9000 series of computers. Due to the large installed base and existing performance instrumentation, we decided to concentrate first on the 3000 family. When the project's collection objectives were met, we would use our experience on similar collection efforts for the 1000 and 9000 families. (At the writing of this paper collection on the 1000 family is under way.) For **the customers** to be measured we chose major accounts, accounts that stretched the limits of a product (such as a large data base user), and as a control group, a set of 100 randomly selected sites. In addition to these targeted groups, articles in INTERACT and HP internal newsletters invited customers to participate in our study. The responses to these articles provided the bulk of the over 225 sites collected.

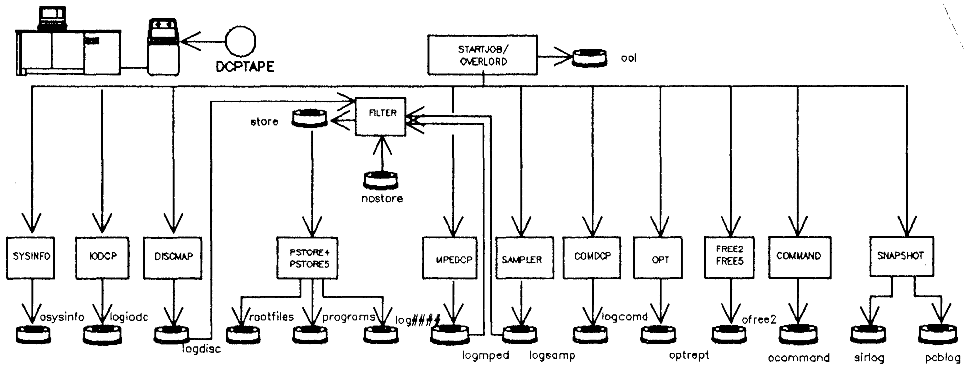
The decision of the **data** to collect and the **duration** of collection were interdependent. We wanted to collect enough data for a representative period of time without perturbing the system or consuming too many resources. We did not want to exclusively own the tape drive for our collection period so we decided to collect to disc. Since we wouldn't know until all the data was analyzed how much free space a typical customer had, we estimated 50,000 sectors for a one hour run would be acceptable. However, running all our collection tools (most of which run in the linear queue) would consume about 17 percent of the CPU which we felt would perturb the system. Therefore, we decided to split the desired one hour collection into two 1 hour phases. Those programs which consumed little CPU and disc space would run for the entire 2 hour collection period. Also, a group of programs would run in the C queue in a clean up phase at the end of the collection.

To capture necessary demographic data a survey form was developed for the site manager to record information such as: principal business, current measures of performance, principal applications, network information, etc. We provided a cover letter which explained the purpose of our collection and asked the customer to perform the collection during a representative time. Some sites were collected twice when no single time was considered representative (e.g. interactive workload during the day and batch work overnight).

For the **method** of collection we wanted a process that would require only an operator and no special setup requirements such as connection of hardware monitors or changes to the operating system software. A simple tape collection system was created. The collection programs were sent to the customer on a 2400' tape which was returned to use with the collected data. Several generations of the collection tape, all backward compatible, were necessary as customers started updating from MPE IV to MPE V.

Figure 1 depicts the flow of the collection phase. The tape labeled **DCPTAPE** (**D**ata **C**ollection **P**ackage **TAPE**) is placed on the systems tape drive. The passwords to MANAGER.SYS are removed for the beginning of the collection. A file equation for the tape ":FILE LOADIT;DEV=TAPE" is followed by a ":STREAM *LOADIT" which causes the **STARTJOB** batch job at the beginning of the tape to be executed. **STARTJOB** creates the collection accounts and reads the second file on the tape which contains the encrypted collection programs. The collection programs are separated and decrypted into job temporary program files. The main controlling program, **OVERLORD**, using process handling, spawns son collection processes depicted by the boxes in Figure 1.

SYSTEM PERFORMANCE EVALUATION PROJECT



COLLECTION PHASE

Figure 1

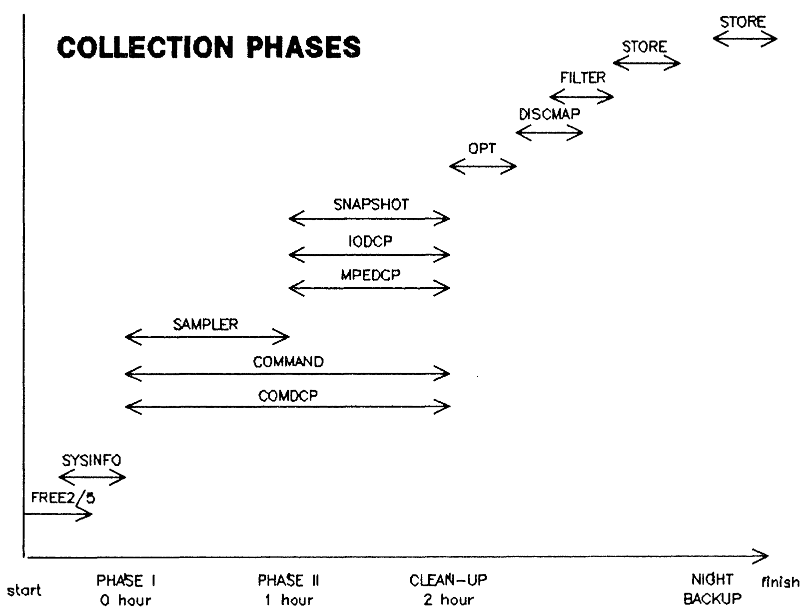


Figure 2

As shown in Figure 2, at the **start** of the collection **FREE2/FREE5** is run to capture information on the amount of free space per volume and the degree of disc fragmentation. This program is followed by **SYSINFO** which captures the system static configuration parameters such as: I/O configuration, volume table, communication devices, device classes, MPE table settings, scheduling, directory, virtual memory, system logging, user logging and spooling. The programs run during the start phase are run at batch priority so execution time will vary depending on system load.

PHASE I begins the main body of the collection programs. Two collection programs are started that run for the next two hours. **COMDCP** does a progammatic SHOWCOM on all configured data communication devices (INPs, SSLCs, and HSIs) once every minute. **COMMAND** does a SHOWCACHE command every minute to capture disc caching information. Since the CPU and disc requirements of these two programs are insignificant they begin with PHASE I and continue through the end of PHASE II. **APS/3000 (SAMPLER)** begins at the start of PHASE I and runs for 1 hour. It monitors the execution of software and produces histograms showing the amount and distribution of CPU time spent by various operating system modules.

PHASE II begins at the 1 hour point. **MPEDCP** turns on the operating system measurement interface and collects global and process level information at 1 minute intervals. **IODCP** turns on the I/O trace portion of the measurement interface and collects a 20 word block of I/O information for every I/O. **SNAPSHOT** wakes up every 10 seconds and copies the entire process control block table (PCB) and the SIR table to a disc file.

CLEAN-UP begins at the 2 hour point and execution returns to batch priority. **OPT** is run to capture current and maximum MPE table information. **DISCMAP** scans the directory and captures the 128 word file label for every disc file on the system. **FILTER** scans the log files just collected for the names of all active non-HP programs on the system. These names are placed in a file to be used later in the STORE process. **FILTER** also scans the log file of file labels for the names of IMAGE root files and adds them to the list of files to be STOREd later. An MPE STORE of the log files from the collection programs and the MPE log files in PUB.SYS is then done to the original collection tape.

At **NIGHT BACKUP**, when no files are busy, the program and root files flagged during the collection are added to the collection tape by streaming a batch job. For security reasons this step is optional.

REDUCTION PHASE

When a collection tape is returned the information from the survey form is entered into a data base and the contents of the tape are restored to a unique group for reduction. Figure 3 shows the relationship of the data collected to the reduction programs. Activity occurring at the customer site is shown above the dashed lines in the section labeled **COLLECTION**. Below the dashed lines in the section labeled **REDUCTION** are the functions that are performed on our system. A single master batch job runs all the reduction and reporting software to produce the single site **REPORT**. A copy of this REPORT is sent back to the customer and the account SE.

Referring to Figure 3, **TERMDRP** uses the system configuration information captured in **osysinfo** to determine which logical devices are terminals. Next, the I/O trace captured in **logiodc** is searched in order to reconstruct transactions for each terminal. Each I/O entry was originally time stamped in milliseconds and identified as a read or a write. Think times for each logical device are calculated as the time difference between the last write and the next read completion. Any think time less than .3 seconds is assumed to be a response from the terminal (i.e. status returned from an escape sequence sent to the terminal). Each time a terminal cycles through a think time it is counted as a transaction.

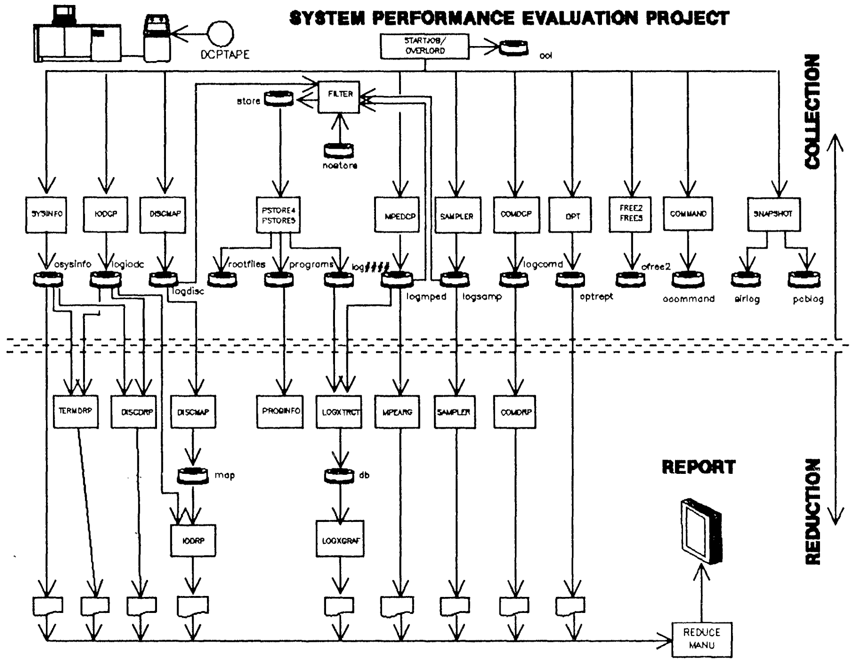


Figure 3

DISCDRP similarly uses **osysinfo** to determine which logical devices are disc drives. The I/O trace in **logiodc** is searched for disc devices. Addresses accessed, seek distances and data transfers are analyzed.

DISCMAP uses the file labels captured in **logdisc** to build a KSAM file with a file's extent addresses as the keys. This file serves as a directory of file names ordered by extent addresses. **IODRP** then replays the I/Os in **logiodc** and matches the disc addresses of the I/O file with the KSAM directory file. The end result is a report of both logical (cached systems) and physical I/O by file name and file type.

LOGXTRCT & LOGXGRAF produce the graphical section of the REPORT. LOGXTRCT creates an IMAGE data base of the raw data found in **logmped**. From this data base LOGXGRAF creates line, pie and bar charts which show global system states and process level information.

MPEARG also analyzes the **logmped** to produce numerical information for the REPORT of global system states and process level statistics.

SAMPLER produces a histogram of system module utilization from the **logsamp** file. This information shows where the system was spending its time by module name and how much time was spent in user code versus system code.

COMDRP reduces the data communication data captured in **logcomd** which shows line utilizations, error rates, retransmissions, etc. for the active INPs, SSLCs and/or HSIs on the system.

During the collection phase **SYSINFO** created a listing, **osysinfo**, that is now added to the REPORT to record the system configuration at the time of the collection. **OPT** created an **optrept** file containing system table configuration information that is added to the final REPORT. Some of the files not mentioned are used later in the summary phase while others are currently not used at all.

Each reduction program directs its output to a disc file where it is further edited using TDP *use files* into a consistent format. Finally, the pieces of the report are assembled into one large TDP document and sent to a HP2680 laser printer. Production of the REPORT takes about 2 hours and consumes 1 hour of CPU time on a series 68. During the reduction many secondary files are created for the further summarization that follows. A typical site will start out needing 18 megabytes of disc space which will expand to 40 megabytes by the end of the reduction.

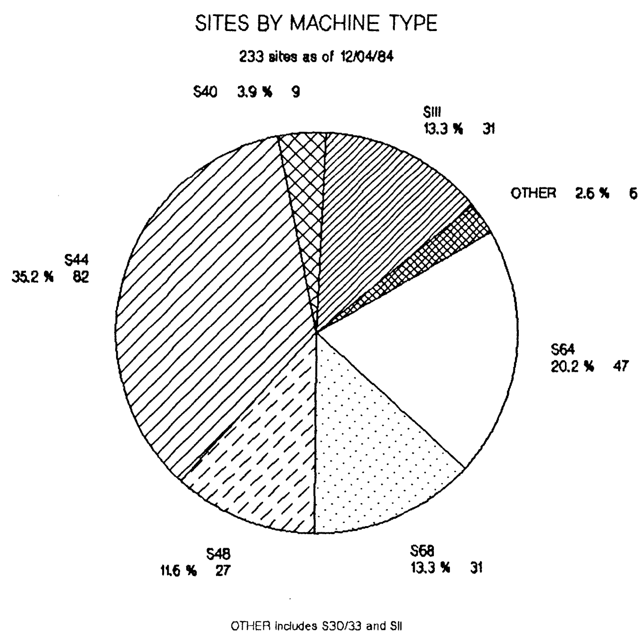


Figure 4

Figure 4 is a breakdown by system type of sites that have been collected and reduced as of 12/04/84. Since the beginning of the collection project there have been several major new releases of the operating system. This presents a constant challenge in maintaining a single collection/reduction system.

SUMMARY PHASE

The overall objective of **SPEP** is to produce *workloads* and *patterns of customer usage* across multiple classes of users and market cells. To perform this comparative effort a summary data base was defined and additional reduction programs were written.

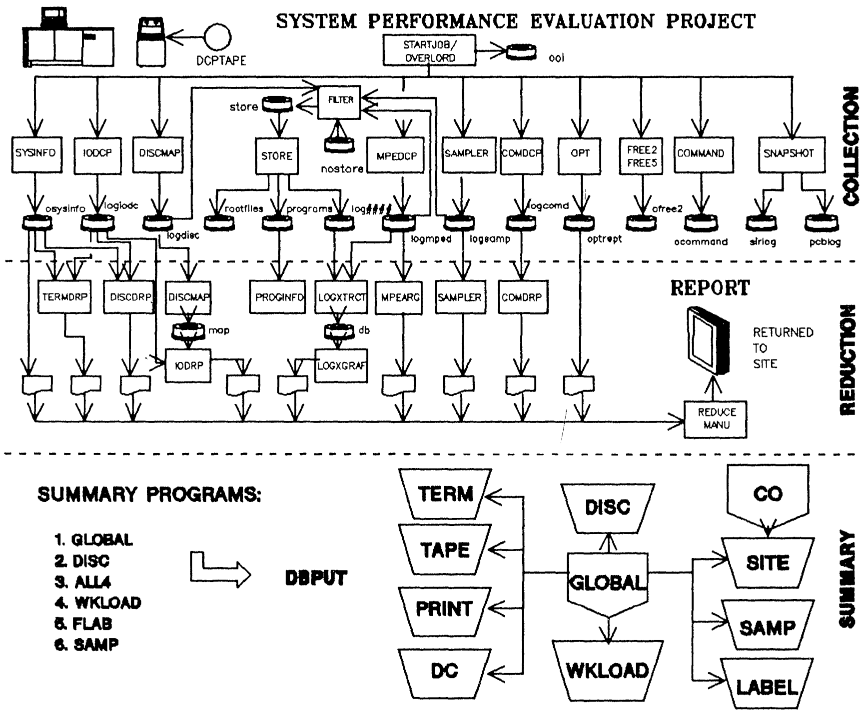


Figure 5

The bottom third of Figure 5 represents the summary effort of the SPEP project. The summary programs draw upon the raw log files and temporary files created during the reduction phase to distill the individual site data into summary records for the summary data base. The IMAGE data base is composed of over 250 unique data items defined with DICTIONARY/3000.

TERM, TAPE, PRINT, DC and **DISC** contain a summary record for each active device for each site summarized. Typically the information in these data sets summarize device utilization. **CO** and **SITE** contain company and site specific information that helps track the collection/reduction effort and identify the market cell the site belongs to. **SAMP** contains SAMPLER information regarding the activity in the various operating system modules for each site. **LABEL** contains a shortened file label for all of a site's files identifying the size, type and accesses patterns. **GLOBAL** contains summary information about the system configuration such as number of terminals configured, amount of memory and type CPU (e.g. 64,44). Finally, **WKLOAD** contains information about the active *workloads* on the system with breakdowns on number of transactions, CPU times, response times, and disc I/Os.

This paper is intended to serve as a foundation for the motivation and implementation of **SPEP.** At the **HP3000 IUG Amsterdam Conference '85**, conclusions pertaining to the *workloads* and *patterns of customer usage* for the sites surveyed will be presented.

BIOGRAPHY

Paul Primmer is the Project Manager of the Systems Performance Evaluation Project at the Hewlett-Packard computer facility in Cupertino, California. With HP for 8 years, Paul started as a Systems Engineer in the District Sales Office of Atlanta Georgia. While in Atlanta, Paul was a 1000 SE for 2 years followed by 4 years as a 3000 SE and SE specialist. In January of 1983 Paul moved to Information Networks Division in Cupertino, California then joined the SPEP project at HP LABS in Palo Alto, California in July, 1983 as a member of the technical staff. The project is continuing and has moved to HP's new Information Technology Group in Cupertino, California.

A FILE ACCESS METHOD FOR SOURCE VERSION MANAGEMENT

Carolyn Spitz
Hewlett-Packard, Cupertino, USA

Summary

Of the several file access methods in use on the HP3000, TSAM was designed for the purpose of simplifying version management for software developers and software maintainers. TSAM maintains one physical file which may represent many versions of a source file and then controls the access to that file by multiple users. This paper discusses the features of TSAM version management and how these features are implemented. The paper will also briefly discuss disc space utilization, recovery and performance of TSAM files.

Need for Version Management

The management of the software becomes more complex as programs are enhanced, corrected and modified during their lifetime. More sophisticated bookkeeping is needed in order to track the various releases of a source program. Recreating past versions of software becomes more difficult and sometimes impossible without knowing the history of its modification. And source files may even experience overlap error caused by two programmers modifying the same code at the same time.

Since the maintainability, auditability and reliability of source programs is necessary to achieve maximum productivity, many version management systems have been developed to help alleviate the problems associated with continually changing programs. RCS[1], SCCS[2], SDC[3] are a few of the version management systems in existence today. TSAM (which stands for ToolSet Access Method) was designed as a file access method which allows version management on the HP3000 and was incorporated into the integrated programming environment of HPToolset. This paper looks at the design and implementation of TSAM, and examines those factors which affect its performance.

Design of TSAM

TSAM was designed so that one physical MPE file may contain many logical source files. Users do not have to set up complicated system accounting structures or archiving techniques to manage multiple copies of their source files. This also results in more efficient use of disc space since TSAM only needs to store the changes between versions. Not only can users access any version of their source from one TSAM file, but they can also trace the history of changes made to the source file.

TSAM allows up to 32 active versions in one file. An active version is any version that has not been purged. TSAM allows multiple users to simultaneously access various versions of the file. TSAM imposes a few restrictions, though, to maintain file integrity. Only one user can be modifying the file, and only the most recent version can be changed. Earlier versions are "frozen" from further modification.

Access to TSAM files is accomplished through a set of intrinsics similar to the file access intrinsics used by MPE or KSAM. The user can open, close, read, modify or request information about the TSAM file by using these intrinsics. The access version for a particular user is determined when the file is opened. The user passes "MYFILE#3 " as the filename parameter to the TSAM intrinsic TBOPEN if version 3 of the file "MYFILE" is desired. TSAM allows "#LATEST " or "#REFERENCE " as part of the filename, too, for designating the most recent version or the default version for readers, respectively. If a version is not specified by the user, TSAM will determine the access version. For a new file the default access version is always version 1. For an old file TSAM will use the reference version as the default version unless a HPToolset workspace name is passed to the TBOPEN intrinsic. The reference version may be declared inside HPToolset using the "SETREF" command. When a workspace name is passed, TSAM will look at the workspace to see what version was intended. It is good practice when calling TSAM from outside HPToolset to always specify the version that is intended.

Another feature of TSAM should be noted here. Unlike some version management schemes that must create the desired access version when the file is opened, TSAM is immediately able to access the desired version. The initiation time for a TSAM file is always constant.

Although TSAM provides both a set of intrinsics and a file structure to permit version management, additional functionality is possible when accessed through the HPToolset environment. HPToolset has a file conversion utility which converts standard MPE files to the TSAM format and visa versa. It also has a utility which lists changes made between versions; allowing users to view the history of a file. Reading and editing TSAM files are also provided as part of the HPToolset environment.

Implementation of TSAM

Examining the internal file structure of TSAM is essential for understanding how TSAM works. TSAM files consist of two or more MPE blocks of 512 words each. Each block, or physical record, represents the unit of information that is moved between memory and the disc file. TSAM does its own deblocking. The logical records in each block may represent one of three different types of information. A given block may contain either key, data or free space information. Each block will have an identifier in the first word indicating its type. The block type cues TSAM to the data structure represented by that block. An identifier of -4 (%177774) indicates that this block is part of the data block data structure, an identifier of -5 (%177773) indicates that it is part of the free space data structure and an identifier of -3 (%177775) or -2 (%177776) indicates that it is part of the key block data structure depending on whether its a leaf or non-leaf block respectively. The following discussion will explore the various data structures maintained by TSAM.

Key Block Data Structure

The keys in a TSAM file are stored in a variation of the B-tree data structure known as a B^*-tree[4]. The advantage of a standard B-tree is that it has a balanced structure requiring only one access for each level of the tree. This allows fast access to any random key in file. The B^*-tree[5] implementation not only has the advantage of fast random access of a standard B-tree but also permits fast sequential access to keys in the file. This paper assumes some knowledge about tree data structures and will not attempt a detailed discussion of B^*-trees. It will only show the implementation of the TSAM B^*-tree. Interested readers may refer to the bibliography for some papers giving more detailed information.

In a simple 2-level B-tree (Fig. A), the lowest levels of the tree are called leaves and all upper levels are called non-leaves or branches. In this case there is only one upper level which also happens to be the root of the tree.

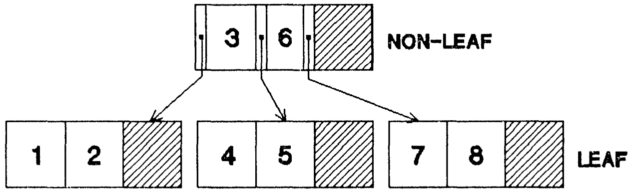


Fig. A. A standard B-tree.

In the examples, we show only a few keys in each block whereas in reality nodes of a B-tree may contain many key records. This reduces the time required to traverse the tree since each node visited represents one disc I/O. Furthermore, as keys are added or deleted, the tree is expanded or contracted in a balanced manner. Unlike standard binary trees which may have some very long and some very short paths, all B-tree leaf nodes will be at the same level. The balancing of a B-tree is accomplished by merging an empty key block with its neighbor or splitting a full key block into two neighboring blocks. Some B-tree implementations, including TSAM's, employ a technique known as underflowing which further maintains the balance of the tree. During an underflow, keys are redistributed evenly between neighbors to postpone splits or merges and to more evenly utilize storage. In TSAM, underflowing is only done on leaf blocks. Maintaining balance in the upper levels is unnecessary overhead since most of the keys exist in the leaf level of the tree.

The B^*-tree structure (Fig. B) is slightly different from the standard B-tree. First, all active keys in the file will be represented in the leaf nodes of the tree. Leaf keys will point to actual data records while non-leaf records serve merely as an index to keys in lower levels of the tree. Notice that if a key appears in an upper level of the tree but not in the leaf level, that key does not have any data record associated with it. The key "6" in our example has been deleted but it remains a placeholder in an upper level of the tree. Updating the upper levels with valid key values is unnecessary overhead. This is because all key searches must terminate in the leaves since this is where the data record pointers are kept. This also implies that the time needed to find any two random keys in the tree will be equal since the number of levels traversed in the tree will always be the same. The format of the key records and the 10-word header stored at the beginning of each block depends on whether the key represents a leaf or non-leaf level of the tree. In some implementations the actual size of the nodes may differ. In the TSAM B^*-tree, the block size is kept constant but a non-leaf block may contain a maximum of 62 index keys while a leaf block will contain no more than 38 keys.

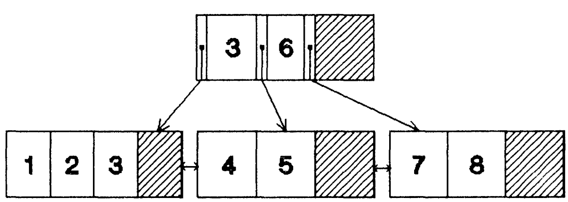


Fig. B. A B^*-tree.

The second feature of a B^*-tree is that the leaf blocks are linked to their neighbors. This allows a user to sequentially read across the leaf nodes without having to traverse the tree. This provides very fast sequential access to the file. In a TSAM B^*-tree the leaf nodes are doubly linked so that forward and backward searches are possible.

A more complicated tree structure (Fig. C) will help demonstrate how key searches actually work. First, if a leaf key block is in memory, this is searched first. Even though a user is requesting a particular key, often the key is in the same part of the file. Checking the memory block first may save several disc accesses. If the key is not found, the key search will then start in the root key block. TSAM uses a binary search to determine the path to take down the tree. Note that any given key in a non-leaf block will have a left son pointer to a block of keys less than or equal to itself and will have a right son pointer to a block of keys greater than itself.

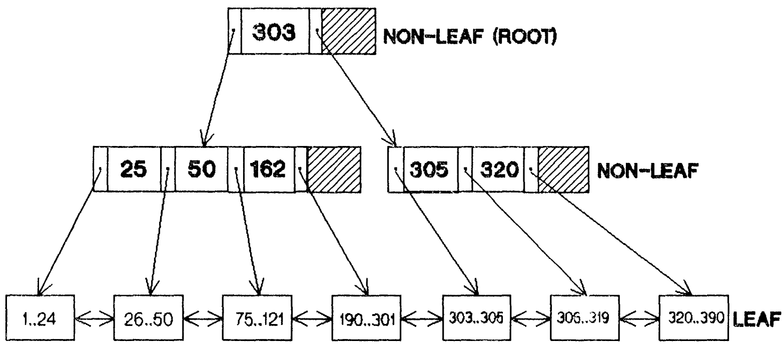


Fig. C. A sample TSAM B^*-tree.

When the next key block in the path is brought into memory, the process is repeated until the leaf level is reached. The leaf is searched to determine whether the key exists and its location if it does. Once the appropriate key is located, TSAM must determine the correct data record to return to the user.

A leaf key record may point to many data records, each data record corresponding to a different version of that key. The format of a leaf key record is shown in Fig. D. Notice that keys are stored in a six-word (12-byte) field. TSAM files may contain as many as 10^{12} keys, each possibly associated with multiple data records!

RECORD DELIMETER	LENGTH OF KEY RECORD	KEY VALUE	DATA POINTER$_1$	. . .	DATA POINTER$_n$
0	1	2	8	12	

Fig. D. Leaf Key Record.

The data pointer portion is variable length. Each four-word data pointer contains a version number, the location of the data record, and a delete flag used to denote that a record in an earlier version has been deleted. Data pointers are added only when changes are made to a data record in a particular version, so not all version numbers may exist. Those that do exist are stored in ascending version number order. To finish a key search, TSAM must locate the data pointer with a version number less than or equal to the access version of the user.

As an example, use this algorithm to find the keys "25", "26", and "50", in that order. After following the leftmost son pointers from the root we arrive at the first leaf key block and find that "25" doesn't exist. Searching for "26" again begins at the root and follows the left son pointer at key "303" again, but at the next level will take the right son pointer of key "25" before arriving at the second leaf block. Since that leaf block will still be in memory when we begin to search for "50", no tree traversal or file access is done, and the key is immediately located.

Sequential searching is very fast in a TSAM file. Since all keys are stored in the doubly linked leaf nodes, no tree traversal is necessary. TSAM maintains a logical record pointer (LRP) to facilitate sequential access. Typically, to start a sequential operation, a user positions the LRP through an intrinsic call to TBKEYFIND. Thereafter, the user can read or write sequentially in either direction. Once again, TSAM must determine the correct data record to retrieve by locating the appropriate version in the data pointer portion of each leaf key record.

Data Block Data Structure

Data blocks in a TSAM file contain a 6-word header followed by multiple data records. The records are written in chronological sequence and are identified by a block number and offset stored in the key record. Sometimes keys may be in sequential and chronological sequence. When this happens, sequential searching will be even faster since the last data block accessed is always kept in memory.

The format of a data record is shown in Fig. E. Although a user builds a file with fixed length records, the data records are stored internally in variable length format. Trailing blanks are compressed but some padding is added at the end so that most modifications made within the same version may be done in place. The padding is 20% of the compressed data, or 10 bytes if no blanks were passed and there is room available in the record. TSAM stores the actual length of the data so that it can be restored when requested by the user. Note that the version number and a delete flag is duplicated in the data record so that the TSAM file may be rebuilt if the key structure is damaged during a system failure.

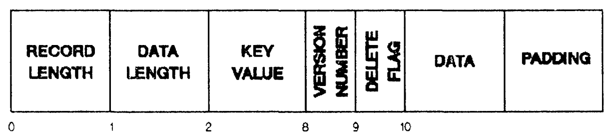


Fig. E. Data Record.

Free Space Data Structure

The remaining MPE blocks in a TSAM file are part of the free space data structure. TSAM attempts to reuse disc space whenever possible by remembering deleted data records and freed key blocks. The free space blocks are composed of 168 three-word entries plus a six-word header. The header is used to link the next block in the free chain. Blocks are never unlinked as free entries are reused so it is possible to have some empty blocks on the chain.

Each entry (Fig. F) for a deleted data record on the chain contains a two-word address giving the block number plus offset of that data record and its length in bytes. TSAM does not attempt to concatenate deleted data records to avoid extra overhead. Two contiguous deleted data records will be viewed separately by TSAM.

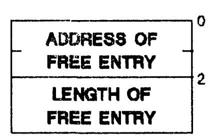


Fig. F. Free Space Record.

Key blocks, on the other hand, may be returned to the free chain as keys are deleted and the B^*-tree structure is compressed. Full block entries can be easily distinguished from partial block entries on the free chain. The length field will be equal to the size of a TSAM block (512 words) and only the block number (no offset) is stored in the address field for full blocks. When a full block is reused it may be used for new free chain, new key or new data blocks. Partial block entries are only reused for data records.

The algorithm that is used to access the free space chain is fairly straightforward. Any time a TSAM file is opened for modify access, the head block of the free chain (which is always MPE block number 0), is brought into memory and kept there throughout the access of the file. Additional blocks will be brought into a work area as needed, but often many operations can be done on the free chain without ever doing a single I/O. To look for a potential free area in the file, each entry length is examined, starting with the entries in the head block, until the first entry that is big enough is found. The "first-fit" entry is deleted from the chain and its location and length returned to the requesting TSAM routine. If none is found, TSAM proceeds as usual, fetching the next available record or block at the end of the file. To add a free entry to the chain, TSAM tries to add it to the first empty slot on the free chain. By maintaining a pointer to this slot, TSAM does not need to traverse full blocks in the chain just to add an entry in an empty slot.

User Labels

Not only does TSAM store information in MPE blocks, but in user labels as well. Three types of information are stored in the 128-word user labels: file directory and control data, summary version data, and data undefined by TSAM but determined by the user of TSAM. User label 0, the TSAM file directory, contains important documenting information about the file such as the location of the B^*-tree root node, the number of versions in the file, and the current and maximum count of user records.

User Label 1, the version table, contains a summary record of the active versions in the file. Each four-word entry (Fig. G) contains the creation date of the version and the number of user records in that version. The rest of the user labels, thirteen in all, are available to users and are not reserved by TSAM. For example, HPToolset uses one of the labels to hold edit file information such as source language and line increment. The rest of the labels are used to hold one-line comments associated with a particular version that were entered using the HPToolset LABEL command.

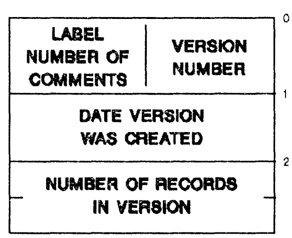


Fig. G. Version Table Record.

Open File Buffer

Each process accessing TSAM requires an open file buffer (Fig. H) which is local to the process. The last type of information necessary for TSAM to function is not stored in the file but resides at the top of its open file buffer. The access mode and version requested by the user, the current LRP and TSAM's MPE file number are maintained here. In addition, a search stack is kept to facilitate access to the B^*-tree.

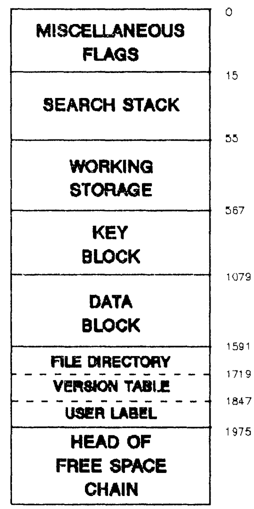


Fig. H. Open File Buffer.

The search stack reflects the search path used during the last traversal of the tree to locate a particular key. Every level of the tree, up to 10 levels, will have a four-word entry (Fig. I) on the search stack showing the block number and offset touched during the last search. This minimizes redundant key block searches when a key must be inserted at an upper level. The search stack also contains a pointer to the end of the used space in each key block so that TSAM can predetermine whether an upper level of the tree will need to be split or merged without having to read the father block. Furthermore, the search stack eliminates the need to maintain a father pointer in each key block.

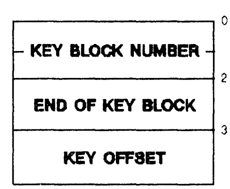


Fig. I. Search Stack Record.

The rest of the open file buffer serves as a holding and work area for the current blocks and labels. The total length of the buffer required for a file opened for modify access is 2560 words. If the file is opened for read access, only 2048 words are required since the head of the free chain is not brought into memory.

Disc Space Utilization

Now that we have examined the file structure in detail, we can look at the implication the file structure has on disc space usage. TSAM files with only one version will require more disc space than its ASCII counterpart because of the extra disc space required for the keys and the free space chain. A savings in disc space is realized when TSAM has multiple versions in the same file because only changes are stored. The common data between versions is not duplicated.

TSAM makes every attempt to save as much disc space as possible. We have already seen that TSAM uses variable length data records in order to compress trailing blanks. We have seen that only records that have activity on a particular version will have new data pointers created in the leaf keys. We have also seen how data records are put on the free chain and reused if the data was added and deleted in the same version. And we have seen how whole blocks can be reused when key blocks in the B^*-tree are no longer needed.

The actual size TSAM builds a file is based on an estimate of the maximum size the user will need. TSAM calculates the size by working from the bottom of the B^*-tree to the top. Given the maximum record size and maximum number of data records requested by the user, TSAM computes the number of data blocks needed based on no compression. Assuming that key blocks are only half full (the worst case), the number of leaf blocks can also be calculated. Based on the number of leaf blocks required, the number of non-leaf blocks required in each level can then be determined in an upward fashion until the root of the tree is reached. Some additional space in the file is allowed for the free space chain. Although it is highly unlikely that the file built would not be large enough to hold all the changes made in the file, the HPToolset user has several options if this happens. The user may expand the file or may copy a later version, or versions to a new file, if earlier version information is no longer needed. In some cases a recovery can also be used to free up some of the disc space.

File Security and Integrity

We have examined the internal file structure of TSAM and discussed how disc space is utilized, but we have not yet seen how TSAM implements a multiple user system while maintaining file security and integrity. A user modifying the latest version also needs to have protection from simultaneous updates. This user is granted exclusive access to the latest version by TSAM. In order to allow others to read earlier versions of the file, TSAM dynamically locks the file at critical times. The file is locked whenever the tree structure changes, whenever leaf boundaries are crossed during sequential access and whenever user labels are written to disc.

Finally note that although TSAM allows multiple user access, some situations require exclusive access to the entire file such as file recovery and file creation. Exclusive access in these cases is enforced by MPE.

Since MPE blocks and user labels are kept in the open file buffer in memory to improve performance and there is some potential risk should the system fail while a TSAM file is open, the dirty blocks in memory are periodically flushed to disc and the MPE end of file is updated. In a cached system, these blocks are written to cache until MPE can satisfy the write request. There is more risk if the cached system should fail because a write may not have been physically complete to disc. As of version A.01.04 of TSAM, MPE will satisfy the write request in the same order as written by TSAM. TSAM turns on the serial write queue flag for the file as an additional precaution on a system that is using a disc cache.

Recovery

Even though precautions are taken to ensure file integrity, it is possible that a file may become corrupted, such as after a system failure or a user abort. TSAM can detect if a file was open with modify access during a system failure and will automatically recover the file when the file is next opened. The recovery routine may also be directly called by the user. There are two types of recovery: full and partial. Both types of recovery are available to the HPToolset user but outside this environment only a full recovery may be done. A partial recover, or "key" recover, is quicker but not as thorough as a full, or "data", recover. Often a partial recover is just used to unlock the file or rebuild the B^*-tree structure.

In a partial recovery TSAM builds a new file having a "TBR" prefix in the name. Starting with the first leaf key in the corrupted file (which is always located at MPE block number 1), the file is sequentially read using TSAM intrinsics. The corresponding data records are written to the new file. The labels are then copied to the new file and updated. If all goes well, the old file is purged and the new file is renamed. If an error occurs during the recover, or the file has more than one version, a full recover will be done automatically.

A full recovery works as follows. Each block in the file is sequentially read using MPE intrinsics. All free chain and key blocks are ignored. Data records are written to a sort file and subsequently sorted by version and key. A new file is built and the data records are written back in key order using TSAM intrinsics starting with the first version and working up towards the latest version. The labels are copied and updated. The new file will be renamed if the recovery was successful. If a full recovery fails, the file must be restored from a backup copy. File system errors and bad TSAM file directory labels are usually the causes of unsuccessful full recoveries.

Apart from correcting file corruption, file recovery has some other benefits. The B^*-tree will be completely balanced at all levels. The free space chain will not have any entries implying subsequent adds, deletes and updates will be faster if the free chain does not need to be searched. Sequential reading may be faster since data records will be in key, not chronological order within the same version. And the allocated disc space may be less since unneeded data is not entered in the new file.

Factors affecting Performance

We have seen how the garbage collection done during a recovery can improve performance. Other factors that can affect performance are listed below.

1. Number of versions in the file. As the number of versions in a file increases, the number of data pointers in the leaf keys also increase, the number of keys that can fit in each leaf block decreases, and the height of the B^*-tree increases. These factors affect the time required to locate the correct data record.

2. Number of data records in the file. As the file expands, so do its supporting internal data structures. This will affect the time needed to locate keys when not searching sequentially.

3. Access method used. Sequential access is generally faster than keyed access.

4. The presence of disc caching. TSAM uses NOBUF I/O so the user will wait while write requests are being honored. If the system is cached, waiting for cache is faster than waiting for disc I/O to complete.

5. Number of entries on the free chain. Massive deletes in the file may cause many entries to be put on the free chain. Subsequent adds to the file will take longer as the free chain is examined for potential open slots in the file.

Bibliography

Note: Although all these references did not appear in the text, they were resources used in the development of TSAM and can be consulted for further information.

1. Rochkind, Marc, "The Source Code Control System," *IEEE Transactions on Software Engineering*, Vol. 1, No. 4 (Dec. 1975), pp. 364-370.

2. Habermann, A. Nico, "Software Development Control System", Technical Report, Carnegie-Mellon University, Department of Computer Science (Jan 1979).

3. Tichy, Walter, "Design, Implementation, and Evaluation of a Revision Control System", Technical Report, Purdue University, Department of Computer Science.

4. Bayer, Rudolf, and McCreight, E., "Organization and Maintenance of Large Ordered Indexes," *Acta Informatica*, Vol. 1, No.3 (1972), pp. 173-189.

5. Comer, Douglas, "The Ubiquitous B-Tree," *ACM Computing Surveys*, Vol. 2, No. 2 (June 1979), pp. 121-137.

6. Knuth, Donald, *The Art of Computer Programming, Vol. 3, Sorting and Searching*, (1973), Addison-Wesley, Reading, Mass., 1973.

7. Bayer, Rudolf, and Unterauer, Karl, "Prefix B-trees," *ACM Transactions on Database Systems*, Vol. 2, No. 1 (March 1977), pp. 11-26.

8. Hewlett Packard, *KSAM/3000 Reference Manual*, Appendix B, (May 1979), Part No. 30000-90079.

Biography

Carolyn Spitz is a Software Development Engineer for Computer Systems Division at the Hewlett-Packard computer facility in Cupertino, California. Ms. Spitz joined HP in 1980 after receiving B.A. degrees in Computer Science and Mathematics from Indiana University. She is currently on the HPToolset project team and her responsibilities include support for TSAM and version management.

VS - VARIOUS

DIGITAL OPTICAL RECORDING . . . WHAT IT IS AND HOW IT WILL IMPACT THE MASS MEMORY HIERARCHY

Greg Engelbreit
Hewlett-Packard, Boise, USA

Summary

The continuing development of new products in the consumer marketplace based on optical recording techniques has brought us to the verge of seeing this new and fascinating technology penetrate into the computer mass storage industry. This paper will trace the migration of optical recording into the world of computers, highlighting the unique technical benefits of the technology relative to magnetic recording while at the same time pointing out the inevitable disadvantages.

As this process of technology differentiation unfolds, it should become evident that optical devices used for the storage of digital information will survive and thrive because they will create a new, versatile layer in the traditional mass memory hierarchy. This layer will contain some of the flavor of both magnetic disc drives and magnetic tape, yet retain an element of uniqueness attributable solely to the technology.

Computer systems vendors are at a pivotal point . . . how is this new and yet clearly viable technology integrated into the current mass storage architecture, and how will it reshape future systems? What applications/customer needs will best be met with this technology? This paper will hopefully serve as the basis for constructive dialogue between ourselves and you, the Hewlett-Packard users, independent software vendors, and third party vendors, who have real-world needs and ideas that can benefit from this exciting new technology.

Introduction

If one were to take a snapshot of the average computer system in the early 1960's, several attributes of that system would stand out. Batch sequential mode was still the order of the day from a processing standpoint, with the workload made up predominantly of accounting applications. Large data files were stored on magnetic tape, but an upstart was just beginning to flex its muscles as a potential competitor to tape . . . the magnetic disc drive. The disc drive proved to be quite appealing as an extension of computer main memory as well as a primary data storage device, and continuing system software developments took advantage of its unique performance characteristics. Soon, magnetic tape moved away from the on-line processing arena and became the choice for secondary storage as a mass memory hierarchy evolved.

A decade later, substantial improvements in software brought the average computer system into the era of transaction processing, with data base applications expanding at a dizzying rate. Hardware improvements, in the form of terminals, disc and tape drives, and communication facilities served as the stepping stones for this key transition. This hardware/software synergy continues to provide the momentum for major improvements in computer performance and usability today.

It is important to quantify the progress made in mass storage technology during these growing years. Areal densities for magnetic tape have increased by a factor of almost 500, while disc drives have improved by a factor greater than 5,000. This means that the capacity of disc drives have virtually doubled every 2-1/2 years since the early 1960's. Data transfer rates for disc drives have progressed from 20 KBytes/second in the 1950's to 3,000 KBytes/second today. Average access times have decreased from the 90 msec range down to today's state-of-the-art figure of 24 msec. As software applications push us headlong into the Information Age, will magnetic mass storage maintain the pace of improvement seen in the past? Just as importantly, will further improvements continue to be affordable? Are there new technologies waiting in the wings to complement or replace magnetic disc and tape drives?

The purpose of this paper is to describe just such a new technology; one that has the potential to impact the computer mass storage industry just as forcefully as the magnetic disc drive did nearly three decades ago. The products that are spawned by this technology are called optical disc drives. This paper will address the key attributes of optical technology and the resultant benefits and shortcomings, expanding on the essential product characteristics required to successfully incorporate the technology into the hierarchy. It will also describe the structure of a mass memory hierarchy that incorporates optical drives as an integral piece of the puzzle. Finally, this paper will propose key applications that benefit from these product characteristics.

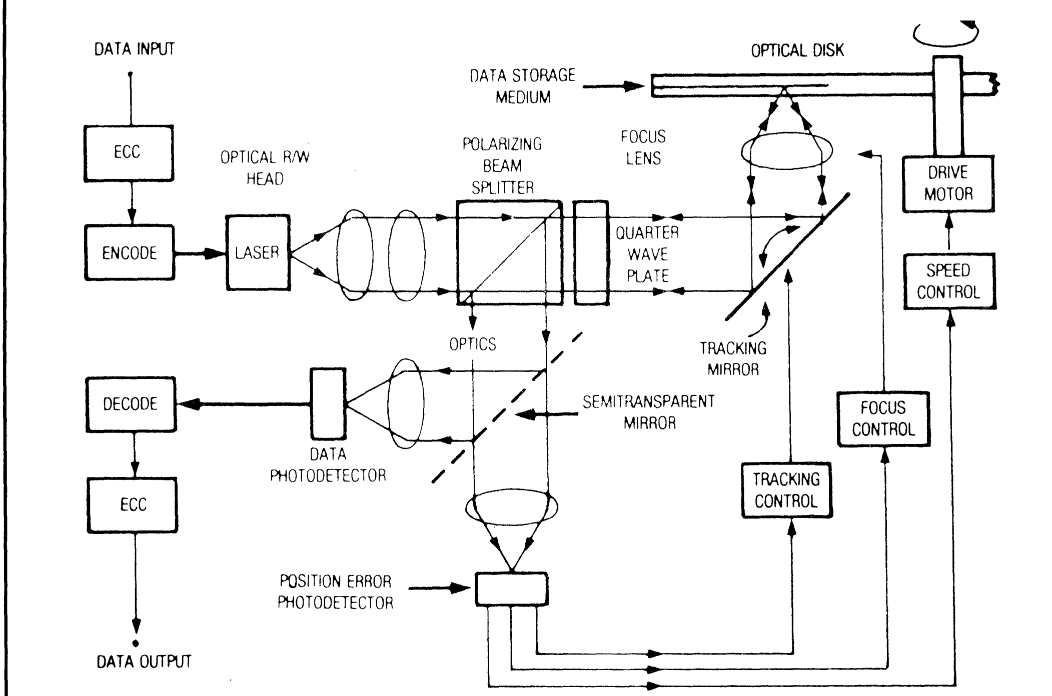
Technology Overview

The invention of the laser has had major impact on today's society. Lasers, combined with a substantial breakthrough in optics, have succeeded in improving the quality of goods and services offered in almost every aspect of life. Among the most recent beneficiaries are the consumer audio and video markets. Several manufacturers have developed products using lasers and optical heads that write information onto special media. This media is then used for mastering purposes to mass replicate the information onto relatively inexpensive media for distribution. In the case of digital audio discs, commonly referred to as compact audio discs (CD), information is stored in digital form and reconverted to analog for playback. How large is the market for these devices? The forecast for CD sales worldwide in 1984 is just over $600M, with video disc players accounting for almost $300M in sales during the same period. Taking into account the interactive video disc games and other commercial applications, laser-based optical technology memory products may reach sales of $1.5B in 1984.

The spectacular initial sales success these laser-based products enjoy in the consumer arena has quickly brought the industry into the next phase of the product life cycle--pricing competition. A healthy result of this volume production ramp-up is the development of sources for individual components and subassemblies. Components of interest in this particular case are the laser source and the optical head. The establishment of key sources for low cost, reliable components such as these parallels efforts toward development of optical devices for the computer mass storage marketplace. There appears to be considerable opportunity for leverage between consumer and computer products at this low performance end of the spectrum. Events of the past year appear to support this, as Sony and Philips have jointly agreed on initial format standards for compact disc read-only-memory (CDROM) products, and forecast product availability in 1985.

The development of a low cost, reliable laser source is a key step in creating a truly viable product in the cost-sensitive consumer market. One source of laser light (which can be tightly focused to a small spot with very high light-power density) is the gas laser. Argon gas lasers are short wavelength (thus permitting smaller spot sizes), but are expensive, require external modulation, and do not have a very long life. Helium-neon lasers are in the intermediate wavelength range and are long-lasting, but also require external modulation, are somewhat large, and are thus not attractive as a laser source for consumer products. Semiconductor lasers represent a better solution due to attributes which include long life, low cost, small size, direct modulation, and low power consumption. The most common configuration is the gallium aluminum arsenide (GaAlAs) laser diode, having a longer wavelength (in the infrared region) of approximately 800nm. Future development of shorter wavelength laser diodes is likely, with resultant increases in spot or areal density.

The challenge of shaping, bending, and focusing the laser beam is left to the assemblage of optical components that make up an optic head. As the light leaves the laser diode, it passes through a collection lens that collimates the beam. From there it passes through one or more prisms whose purpose is to reshape the beam from elliptical to circular. At that point, the circular, collimated light is directed to a mirror which "bends" the beam 90 degrees, directing it through a servo-controlled focusing lens (some optical head designs do not require a mirror). This lens in turn focuses the beam so that the spot size is approximately 1 micron at the point of contact with the active layer on the media (see Figure 1).



Source: Computer design, Alan E. Bell, IBM, January 1983

Figure 1. Elements of an Optical Head

During the **read** operation (the consumer products mentioned earlier are all read-only devices), the laser diode operates in a continuous wave (CW) mode. In addition, it operates at a power level of 1-2 mW. This lower read power (about 10% of write power) ensures that no damage or alteration occurs to the written data during the read operation. The read beam follows the same optical path as noted above on its way to the media. After striking the media active layer, it bounces back through the focusing lens (which now serves as a collimating lens). Instead of continuing on in the reverse direction, at this point it then passes through a beamsplitter and is diverted to a photodetector. This photodetector is tasked with detecting changes in beam amplitude caused by the marked or unmarked portions of the media active layer.

During the **write** mode of operation, the laser diode is pulse-modulated (50-100 ns pulses), and operates at approximately 10-15 mW. This power level is sufficient to modify the active layer of the media in a detectable fashion. Continuing optics research is aimed at improving efficiency, reproduceability, and miniaturization while at the same time reducing cost.

Another beneficiary of development activities taking place with consumer products is the optical media. As with the drive mechanism components noted above, market forces dictate that the read-only media be low cost and capable of mass replication. Before proceeding further, it is appropriate to review in some depth the technological aspects of optical media research and development achieved by media vendors to date.

Rather than alter the magnetic orientation of particles in an active layer as required by magnetic recording techniques, the fundamental approach to optical recording is to effect a change on the media surface that will in turn alter a reflected (or transmitted) laser beam in a manner that is consistently detectable. There are three approaches typically used to modify the surface of the media, all of which are detected by amplitude or phase changes in a low-intensity reflected laser beam:

* **Pit forming or ablative** - in this approach, a pit is formed in the active layer by a short laser burst (typically 10-15 mW) that imparts sufficient energy to melt a pit in the material. The pit has a diameter of about 1 micron. The active material is characterized by having a low vaporization temperature. Tellurium is one type of material that can be used for this purpose, but oxidizes quite readily in the presence of moisture. To combat this, and yet retain its high sensitivity and low melting point, tellurium is usually alloyed with carbon, arsenic, selenium, bismuth, or oxygen (TeOx).

* **Bubble forming** - in this technique, the heat from a laser partially vaporizes the layer of material directly beneath the active material. The active material is a ductile metal film, and deforms into the shape of a bubble with the gaseous pressure from beneath. This film is usually a noble metal such as platinum or gold, thus exhibiting excellent resistance to oxidation.

* **Phase change** - here, the active layer can exist with stability in two different states with differing reflectivity. An example is a material that is stable in an amorphous or crystalline state. With the application of a very rapid laser pulse ($\leq$100ns), the crystalline material is heated just above its melting point. At the completion of the pulse, the rate of cooling is quite fast, effectively quenching the material into its amorphous state.

Changes in the material state alter the reflectivity and are thus detectable with a low-power read laser.

A second aspect of optical media development is the disc substrate. The substrate is the foundation of the media, giving it mechanical strength, stability and geometric precision. Magnetic disc media use aluminum as the primary substrate material, and aluminum is also usable for optical media. It is very stable, and is available in high volume in a variety of sizes. Glass can also be used as an optical substrate. As with aluminum, glass is dimensionally and chemically very stable. With chemical tempering it can be surprisingly rugged, although it also relies on external packaging for mechanical protection. Glass is clearly appropriate for media designs that require through-the-substrate recording. The final class of substrate material commonly used is polymers. These polymers, either PMMA (poly methyl metha acrylate or polycarbonate), can be produced at a very low cost, and with care can exhibit adequate chemical stability. In addition, polymer-based substrates can be easily mass replicated with stamped or molded-in grooves for preformatting and tracking purposes.

The final piece of the optical media development puzzle is the encapsulation of the active layer. This is critical for dust and handling protection as well as prevention of corrosion. One solution is offered via the "air sandwich" structure, in which the active layer is contained inside two transparent substrates placed face-to-face with an air gap in the middle. The active layer is thus isolated in a very clean chamber of air (see Figure 2). Another solution is contact encapsulation, in which an optically stable material such as SiO_2 or PMMA is applied directly over the active layer with sufficient thickness to provide dust and scratch protection.

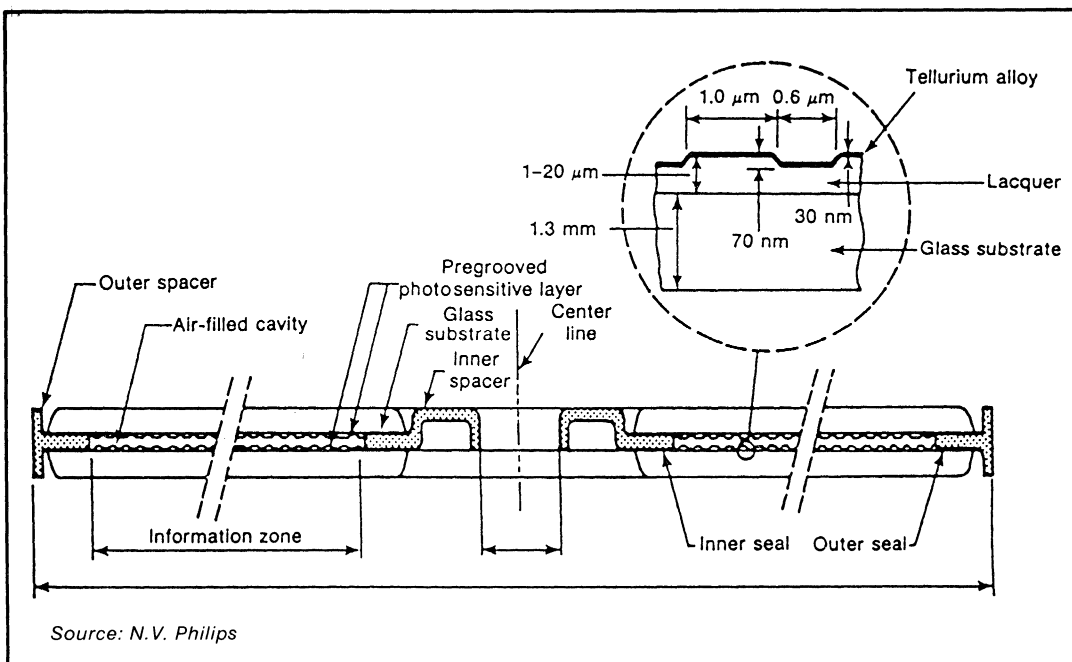
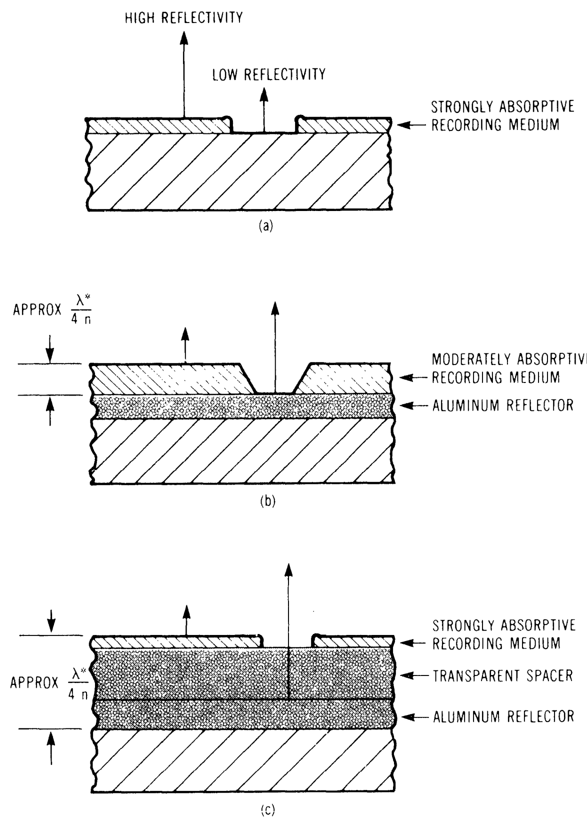


Figure 2. Philips Air Sandwich Media Structure (Double-Sided)

Many techniques have been considered for combining the substrate, active layer, and encapsulation material together to form a functional piece of media. Interspersed among them can be optical spacers, reflector layers, and absorbing layers designed to arrive at the best set of optical properties that match a given drive mechanism. This multi-layering approach can be considered a "tuning" process, in which the laser power requirements can be minimized. This can have a positive affect on the operational life of the laser source. Tuning can also make the detection process in the read mode somewhat less of a challenge for the photodetector. Cooperative drive and media development is essential due to the many interdependencies (see Figure 3).



$^{*}\frac{\lambda}{4n}$ = APPROX 125 nm WHEN λ = 800 nm ASSUMING A REFRACTIVE INDEX (n) = 1.5

Source: Computer design, Alan E. Bell, IBM, January 1983

Figure 3. Optical Media Layering Configurations

Characteristics of Optical Products

It is appropriate at this time to break away from a technology review and focus on the potential benefits of applying optical recording technology to the computer mass storage sector as a possible replacement for or enhancement to existing magnetic storage devices. First, however, it is important to add one or two ingredients to the review. What would happen if a more powerful laser source and modified electronics were placed in a device such as the CD? That device now becomes a write-once, read-many (also commonly known as direct-read-after-write, or DRAW) product with excellent chance of success as a computer peripheral. Almost all of the recent announcements and product introductions for first-generation optical drives in the computer arena are in fact write-once devices. Potential applications for such products will be discussed later.

A key benefit of optical recording is substantially increased areal density. Taking into account bit density, track density, and error correction techniques, areal density improvements can be at least a factor of 10 greater than current magnetic recording technology. Commercially available optics and semiconductor lasers permit bit (linear) densities approximately the same as magnetic disc drives (14,000-16,000 bpi). Due to the circular configuration and small diameter of laser-written spots, however, track densities are greatly improved in optical recording. In fact, track density is roughly equal to bit density (15,000 tpi). Track density for current magnetic drives is approximately 1,000 tpi. Clearly, improvements in areal density for both technologies will continue, but it is not within the scope of this paper to outline the methodology for future enhancements. Forecasts for recording densities anticipated by 1990 are approximately $2x10^8$ bpsi for magnetics and $10-13x10^8$ bpsi for optics.

For perspective, it is interesting to note that the already-standardized CD media is 4.72 inches in diameter and has a capacity of 550 MBytes. This high capacity is partially attributable to the constant linear velocity (CLV) method of recording used in CD. Using the CLV method, rotational speed of the disc is decreased as the head moves outward from the center of rotation. Because the RPM decreases toward the outer diameter, the spots or marks are uniformly spaced on every track. Although excellent for serial recording, CLV is not practical for a random access environment due to the delay of varying rotational speed over and above seek and latency delays. If the more common (to magnetic discs) constant angular velocity (CAV) method of recording were used, that same CD media could still store an impressive amount of data.

Initial development activities for write-once products tend to be focused on larger diameter media. As noted later, intended applications stress high capacity and low cost-per-megabyte. Several Japanese vendors, among them Sony and Matsushita, have concentrated on 8-inch media. One American computer peripheral vendor, STC, has developed a drive using 14-inch media that represents a formatted capacity of 4 GBytes. The majority of first-generation products, however, will have media with 12-inch diameters (see Table 1). Smaller diameter drives in the range of 5.25 inch, 4.72 inch, 3.5 inch, and perhaps even 2 inch, will most certainly be addressed by new start-up ventures as well as the above noted major manufacturers.

As the seek times indicate in Table 1, first-generation optical drives are roughly an order of magnitude slower (hundreds of ms versus tens of ms) than magnetic drives. This same relationship also exists when comparing latency times between the two technologies. Data transfer rates for optical drives typically compare with today's high-end floppy and low-end Winchester drives. Future generations of optical drives will narrow the performance gap. This understanding of relative performance is important when considering the logical application areas for optical drives in the computer mass storage hierarchy.

Earlier comments in this paper highlighted the market potential for laser-based consumer video and audio products for 1984. The cost-reduction programs that are an integral part of the ramp-up of volume manufacturing for these types of products bode well for the computer industry. Leveraging common technology, these pacesetting efforts have the effect of reducing the price for first-generation write-once products currently being announced by computer peripheral vendors. This phenomenon, combined with the inherent advantage of high density and high storage capacity, permits optical drives to be extremely competitive with magnetic disc drives and tape drives on a cost-per-megabyte basis ($/MByte).

Table 1. Optical Disc Drive Comparison

Company	Formatted Disc Capacity (GBytes)	Disc Diameter (Inches)	Average Seek Time (MS)
Hitachi	1.3	12	250
Matsushita	0.7	8	300
NEC	1.3	12	450
Optimem	1.0	12	150
OSI	1.0	12	150
Sanyo	1.0	12	300
Sony	0.5	8	
STC	4.0	14	85
Thomson-CSF	1.0	12	100
Toshiba	1.2	12	500

Three attributes of optical technology are important to note. The first attribute is **random access** to data. Although the preferred method of storage involves writing data sequentially onto a rotating disc, the retrieval of information is identical to a magnetic disc drive--that is, random access.

The second attribute relates to the method of writing data onto the media. Since the laser beam spot size of approximately 1 micron is created by the focusing lens working in combination with the multi-layered media and modulated laser power, the distance that the optic head must "fly" above the media is not critical. The head-media gap is typically on the order of 0.5mm (magnetic heads fly at 0.25-0.5 microns). This means that optical drives are significantly **less susceptible to contamination.** Fingerprints, hair, and smoke particles do not create the danger of catastrophic head crashes. For protection of media during handling and transportation, however, most manufacturers sell media encased in a protective cartridge or cover.

This leads to the third attribute, which is **media removeability.** Without this capability, read-only and write-once products would be much less viable in the marketplace. Taking the removeability aspect a step further, it is easy to envision a multi-platter autoloader mechanism, somewhat like a jukebox, that could store and retrieve any number of optical media platters. The amount of data stored on-line with this jukebox approach is almost unlimited, bringing $/MB figures down to the unheard-of levels of less than $1/MB). Several companies are developing (or already selling) jukebox hardware. These include Integrated Automation, Matsushita, Toshiba, Kodak, RCA, Philips, Hitachi, STC, and Filenet.

Mass Memory Hierarchy

Earlier, several fundamental questions were raised about the pace of progress into the Information Age, and the ability of magnetic mass storage technology improvements to maintain the necessary forward momentum. From a storage capacity standpoint, there is considerable potential yet to be realized for discs and tapes. For instance, an earlier comparison of areal densities that could be achieved by 1990 for magnetics and optical technology placed magnetics within a factor of five or less of the potential for optical. On this basis alone, magnetics would not appear to be a growth constraint. However, the demand for on-line information is only bounded by the imagination. The push toward such applications as document and image storage, talking encyclopedias, computer-aided instruction, etc., implies that storage needs must increase by orders of magnitude. Today, the addition of "disc farms" to a local area network or stand-alone system is not only impractical but cost-prohibitive. Simply stated, pressure is being applied to the basic mass memory hierarchy structure as it exists today. A review of that structure is appropriate in order to better understand the changes necessary to accommodate the need for more on-line storage capacity.

The evolution of the mass memory hierarchy is somewhat brief. The opening paragraphs of this paper traced the progress of computer systems architecture to the point where on-line storage consisted of semi-conductor memory (RAM) combined with magnetic disc drives, and off-line storage became the primary domain for magnetic tape. This hierarchy remains essentially intact today, but external forces such as demand for greater system performance and increasing software sophistication have caused considerable refinement. Within RAM (CPU-resident), there is now a tier of functions that include CPU registers, cache memory, main memory, and I/O cache. At the next level, magnetic disc drives are used for virtual memory, directory storage, and primary program and data storage. Discs and tapes combine to form secondary storage for staged programs and data. At this point, the transition is made to off-line (archival) storage. The definition of off-line has now expanded beyond tape to include paper, microfilm, etc. Aside from redefining the activities that take place at each level within the hierarchy, little has been changed or added to accommodate the exploding on-line mass storage needs.

To better understand this mass memory hierarchy, it is best to envision a pyramid with the top part occupied by RAM. The middle portion, which consists of magnetic disc drives, joins together with the top level to represent on-line storage. The bottom level is considered off-line storage and is made up of tape, paper, etc. For proper perspective, consider that 95% of the world's data is made up of source documents, which of course is at the bottom level of the pyramid. At the top of the pyramid, descriptive terminology includes metrics such as very high $/MB, volatility, very high performance (fast access), and low capacity. Moving toward the bottom, these become low $/MB, permanence, infrequent access, and very high capacity (see Figure 4).

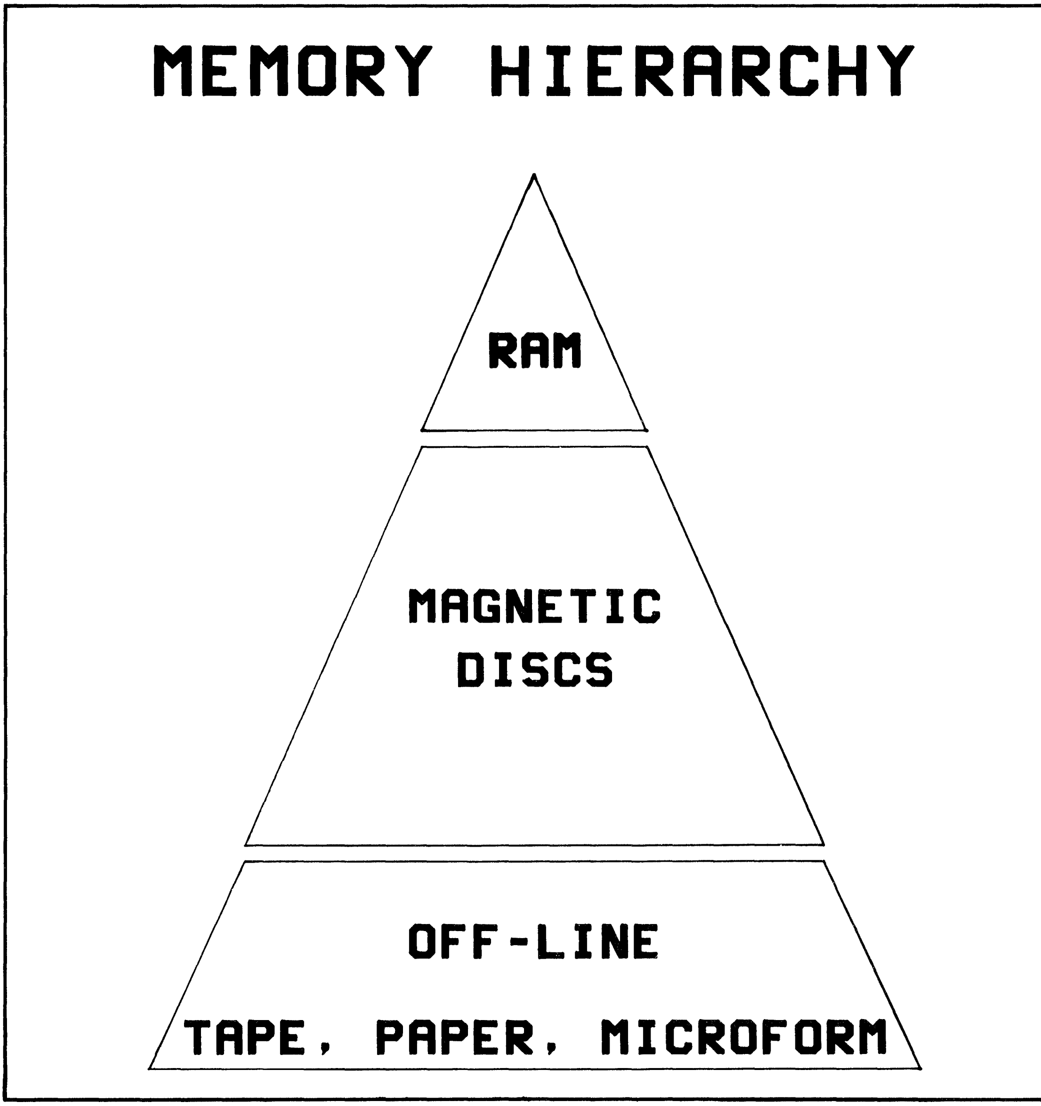


Figure 4. Traditional Mass Memory Hierarchy

Given this hierarchical structure, it is clear that write-once products can make a measurable contribution. With characteristics such as very high capacity, permanence, long shelf life, low $/MB, random access, and media removeability, optical technology fits extremely well into the **on-line secondary storage** role. Adding optical drives with associated mechanical jukeboxes to the off-line storage lineup literally changes the definition of archiving. The archival function could move up into the on-line arena, reducing off-line storage to the minimum essential activities of backup (for catastrophic failure) and data interchange (see Figure 5).

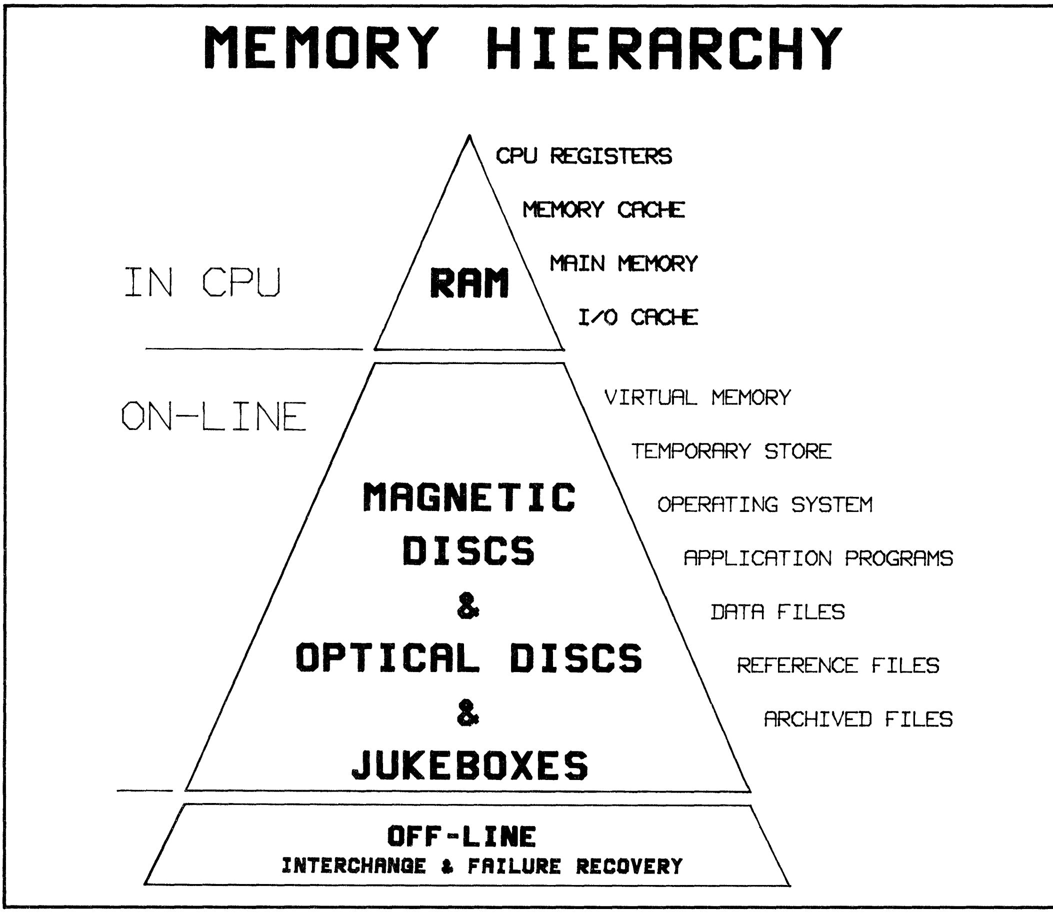


Figure 5. Enhanced Mass Memory Hierarchy

Applications

Clearly, a fresh approach to this mass memory hierarchy is appropriate to permit optical technology to contribute substantially to the success of tomorrow's computer system architectures. This approach not only includes a study of the interplay between magnetic discs and tapes with optical devices, but also the modification of computer operating systems to make efficient use of all devices. It is the task of systems integrators to bridge this gap between old and new, resulting in truly useful computer processing applications.

For completeness, it is important to distill the preceding optical technology overview and mass storage hierarchy discussions down to a meaningful application summary. The following list is not meant to be all-inclusive, but will hopefully serve as a catalyst for further discussion in the readers' areas of interest.

Read-Only (CDROM)

* interactive computer-aided instruction
* electronic publishing
* software distribution
* data distribution
* intelligent, talking encyclopedia
* point-of-sale displays

Recall that CDROM is characterized by inexpensive drive mechanisms, inexpensive media, and fast, low-cost mass replication of media. Small size and ruggedness of media permit easy transportability. Data is non-modifiable.

Write-Once, Read-Many

* archival storage
  - medical histories, X-rays, CAT scans
  - pharmaceutical drug abstracts
  - financial audit trails
  - census information
  - legal case histories, court records
  - stock and security analysis
  - university registration and transcripts
  - actuarial tables
* document storage and retrieval
  - electronic file cabinet
  - microform replacement
  - source documentation
  - manufacturing specifications drawings
  - CAD/CAM
* data acquisition
  - satellite telemetry
  - seismological data gathering
  - intelligence gathering
  - transaction logging/journaling
* on-line library
* backup
* data interchange
* data migration (file server)

Write-once drive products, although not as inexpensive as read-only, will be very competitive with magnetic products. Media pricing will become quite attractive when manufacturing volumes mature. Media will be easily transportable. Data is once again non-modifiable and has a long shelf life.

Erasable

Yet a third thrust within optical technology is the progress toward development of erasable (write-many, read-many) optical drives and media. It is generally agreed that these products will lag write-once to the market by 2-3 years. They will exhibit many of the characteristics of magnetic disc drives, yet retain much of what is unique to optical technology. As a consequence, they will compete directly with magnetic disc drives in the marketplace. A review of erasable optical drive development was not in the scope of this paper, but will be the subject of future review.

CONCLUSION

Optical technology holds considerable promise as an integral part of the computer mass storage offering of the future . . . an offering that must be geared to the burgeoning storage requirements of the Information Age. This paper offers a glimpse at the technology basics of optical recording devices and describes characteristics unique to that technology.

First-generation optical products are being introduced almost daily. Read-only devices are carving a new niche outside of the consumer marketplace, and write-once product integrators are facing up to the challenge of complementing, and supplanting magnetic disc and tape drives in their traditional stronghold. Timely and thorough systems integration, combined with clear applications focus, is the key to success.

Biography

Greg Engelbreit
is a Product Manager at Hewlett-Packard's Disc Memory Division in Boise, Idaho. For the last two years, Greg has been involved with product marketing activities for future mass storage products using optical technology. He has been with Hewlett-Packard for five years.

References

Ammon, G. J., "Archival Optical Disk Data Storage," Optical Engineering, 20, 1981, pp. 394-398.

Bell, A. E., "Critical Issues in Magnetic and Optical Data Storage Technology," Proceedings SPIE, 382, 1983.

Bell, A. E., "Optical Data Storage Technology Status and Prospects," Computer Design, 22, 1, January 1983, pp. 133-146.

Bell, A. E., "Optical Disk Storage Technology: Past and Future," Proceedings SPIE, June 1984.

Hoaglund, A. S. "Measuring The Potential For New Storage Technologies," Digest of Fifth IEEE Symposium on Mass Storage Systems, October 1982, pp. 7-8.

Kenville, R.F. "Hardware Issues for Optical Mass Storage Systems," Digest of Fifth IEEE Symposium on Mass Storage Systems, October 1982, pp. 52-57.

Mallinson, J. C. "The Next Decade In Magnetic Recording," Digest of Sixth IEEE Symposium on Mass Storage Systems, June 1984, pp. 89-92.

Optical Memory News, 11, Sep-Oct 1983, p. 11.

Optical Memory News, 15, May-Jun 1984, pp. 14, 17, 19-22.

THROW AWAY THE KEYBOARD

Dennis Heidner, Boeing Aerospace Co., Kent, WA, USA
Amy Heidner, John Fluke Mfg. Co., Everett, WA, USA

Abstract

Good-bye QWERTY. Hello Barcode, Mouse, and OMR!

This paper reviews alternate data entry techniques that can be used with the HP3000 computer. The authors discuss barcoding, touch screens, graphics tablets, optical mark sense readers, and voice entry. This paper is intended to be practical in nature, reviewing how each technique works and some special considerations which must be made before totally eliminating the keyboard as an entry device. The paper is based on a review of available products by manufacturers such as Hewlett-Packard, INTERMEC, SCAN-TRON, and Votan, and on the practical experience of the authors.

I. Introduction

As the computer evolution has continued we are beginning to look at other areas to improve throughput and performance. In the past most of the effort has been on main processor (hardware) or software improvements, with little if any time spent on examining the user interface to the system. The intent of this paper is to discuss the user interface problem, new user interface devices and software that will allow us to adapt the computer to the user, not the user to the computer.

Historical Perspective

Today the predominant method of data entry is a CRT device using the widely accepted QWERTY keyboard. The general layout of the keyboard was developed in the 1860's by Christopher Latham Sholes [1]. Contrary to common belief the arrangement of the keys was not done to minimize the effort of the typist, but instead was chosen to slow the typist down so that the hammers would not jam. No effort was made to use an arrangement that would reduce the training time for the novice typist. Seventy years later, (after the mechanics of typewriters improved) a professor at Carnegie, August Dvorak, developed a keyboard based on the frequency that keys were used. The result actually not only improved the speed of experienced typists, but also reduced the learning curve for new typists. Unfortunately this keyboard arrangement did not gain widespread acceptance because the QWERTY keyboard had already managed to become the mainstay of the office. Use of the newer faster keyboard would require retraining of large numbers of experienced typists.

We have thus encountered one of the first tradeoffs in the user-machine (computer-related) interface. In this case a less than optimal keyboard is exchanged for user acceptance. It is important to remember that any system we conceive must be accepted by the users and adapted for the computer.

User-Interface Guidelines

There are a number of informal "laws" which guide the development of user interfaces; these apply to keyboard alternates as well as to more traditional data input methods.

Rule Number One: Know your users.

In the paper "Building a User-Derived Interface" [2], the authors noted that "Novice (users') behavior is inherently sensible, and the computer system can be made to adapt to it." Therefore the first goal we must consider when eliminating the keyboard is to adapt to the user. We should also look carefully at their requirements; are we trying to improve productivity, increase the user acceptance of a new system, or reduce the input error rate?

Rule Number Two: Know what the goals are.

If the goal is to improve productivity, then we must be realistic. Remember that all new systems will have hidden costs. In the case of a new computer system the cost that is almost always ignored is the operation costs (backups, paper, etc.). Hidden costs are also applicable when you are looking for a replacement for the keyboard.

If the goal is to improve user acceptance, carefully review the users' needs, considering their backgrounds and educational experience. It is important to remember even an inexperienced computer user's ideas may be very sensible if you listen carefully to what is said. If the goal is to reduce errors, then you need to classify the types of user errors encountered. Are the errors minor mistyping, slips, due to the lack of training, or intentional?

As you might suspect, very seldom do you optimize for just one of the above goals. Instead you must strike a careful balance of productivity, reliablity, ease of use, acceptance by the users, and finally cost.

Rule Number Three: Be consistent.

If a program requires the user to enter data in several different ways, there is an increased chance of introducing errors. Furthermore, since the users may not always know what is next they will slow down and wait for cues as to which devices or data entry method to use. So if you are going to use a barcode reader, for example, try to use that barcode reader in as many places that you can, with very obvious visual clues that you expect the user to read, if you want some other form of input.

Rule Number Four: Reduce clutter.

There is an old adage which says "out of sight, out of mind". This becomes very important when designing user interfaces that the novice will be using. The fewer switches that must be flipped, the cleaner the screens, the better accepted the system will be. Whenever possible suppress (from the screen) information that you may be downloading to the keyboard alternate.

Rule Number Five: People will make errors.

Make sure your method of data entry is forgiving. We generally assume this means only during data entry, however it also includes the operation of the alternate keyboard devices. Make sure your replacement is insensitive to human errors regardless of whether they are accidental or deliberate.

> Rule Number Six: For a system to be accepted and liked by the users it must provide feedback.

There are two feedback pitfalls to avoid. The first is the nonsense error message. How often have you seen the message "UNKNOWN COMMAND NAME"? Although this means something to the computer, it means very little to the novice. The other common mistake is to provide only negative feedback. If the users enter the data properly, let them know; this may be as simple as printing "TRANSACTION COMPLETED" in a window at the bottom of the screen.

> Rule Number Seven: The keyboard alternate must be reliable and accurate.

Even for the novice users a keyboard is simple to use and very reliable. If they press "0" they know that "0" will be accepted by the computer. Some keyboard alternates are pushing the current technologies; as a result they are more often prone to generating errors. When this is the case, the users must often repeat the input action or double check their work.

II. Keyboard Alternates

There are five classes of keyboard alternates which we will cover; they include: barcoding, touch-sensitive screens, tablets and mice, optical mark readers, and voice input. As we discuss the various methods, keep in mind the rules which we have already covered.

Barcoding

The basic goal of barcoding is to represent data in a form which is easily machine-readable, in this case, a sequence of dark and white areas. There are more than 25 different methods to encode the information onto the barcode labels. Currently three methods are the most popular. These are "Two of Five", CODE39, and UPC.

Two of Five

The "Two of Five" code was originally developed for use in photofinishing laboratories. The information is represented by the width of the barcodes. Wide bars represent a 1, narrow bars a 0. The code derives its name from the fact that all digits are 5 bars wide of which two bars must be ones. Table 1 shows how the digits are encoded for "Two of Five" and Figure 1 shows a sample barcode.

Digit	Weighting Value 1	2	4	7	Parity
0	0	0	1	1	0
1	1	0	0	0	1
2	0	1	0	0	1
3	1	1	0	0	0
4	0	0	1	0	1
5	1	0	1	0	0
6	0	1	1	0	0
7	0	0	0	1	1
8	1	0	0	1	0
9	0	1	0	1	0

Table 1. "Two of Five" Encoding

An improved version of the "Two of Five" barcoding scheme is called "Interleaved Two of Five".

In this method both the spaces and bars are encoded. This results in improved error detection and increased density. For this reason industry is switching from "Two of Five" to "Interleaved Two of Five".

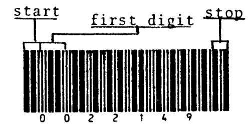


Figure 1. Sample "Two of Five" Barcode

CODE39

CODE39 is a barcode encoding scheme patented by INTERMEC in 1974. CODE39 has been accepted as the standard barcode labeling method for most forms of manufacturing, libraries, hospitals, and by most U.S. government agencies. The name is derived from the fact that there were 39 characters in the original alphabet. In CODE39 two of the five bars are wide, and one of the four spaces is wide. In the final implementation of the code there are codes for the digits (0-9), all 26 capital letters, a space and six other special function codes. Figure 2 is an example of a typical CODE39 label. The TELEPEN (British) code is similar to CODE39 but has representations for 128 ASCII characters.

INTERMEC SAMPLE

ABCDEF-1234567890

Figure 2. Sample CODE39 Label

UPC

UPC stands for Universal Product Code. UPC codes are commonly used to identify grocery, drug, and health-related products in the U.S. There are five basic versions of the UPC codes; all are basically the same, differing mainly in how the bars are organized into data fields. In UPC each character is made up of two dark bars and two spaces. The UPC alphabet consists of the digits (0-9) and three special characters: start, stop, and center-of-label. Figure 3 is an example of a UPC label.

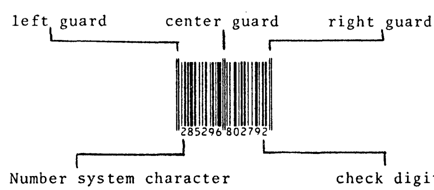


Figure 3. Sample UPC Label

Barcode Printing

Barcodes can be printed using a wide variety of techniques, including dot-matrix printers, special barcode printers, commercial photocomposition, and thermal printers. The quality of the printed label is more important than the print method used. There are several key items that effect the quality of your labels; these include the quality of the paper stock, the condition of the ribbon that was used, the type of ink, and the quality of the adhesive.

The paper stock should be white, without a glossy finish. Be sure to monitor the barcodes for voids in the labels (see figure 4); voids result from a poor ribbon or contamination of the paper stock. The ink (dark bars) should reflect less than about 20% of the light in the near infrared region of light. The contrast between the dark bars and white spaces should exceed 50-60%. [3] Monitor the printing tolerances of the bars. If the printer is having difficulty maintaining the barcode widths, the edges will appear jagged. Voids can sometimes result in misinterpreted labels, whereas bars that are out of tolerance will generally result in a low read success rate.

Be selective when you chose an adhesive for your barcodes! Some adhesives actually give off fumes or absorb moisture from the air. Over time barcodes with these adhesives may discolor and fall off. If you expect the barcodes to be scanned many times and last for more than a few months, they should be covered with some form of clear plastic overlay. Overlays improve the usablity of the barcode by reducing the amount of dirt that collects on the label, and preventing the ink from becoming contaminated or smearing.

Which Barcode Symbology?

Most of the newer barcode readers are capable of reading UPC, Code39, 2 of 5 Interleaved and several of the other encoding schemes without operator intervention. Even so, when choosing between CODE39, "Two of Five", UPC, or one of the other symbologies, several factors must be considered.

Are your labels being attached to products which are delivered to customers? If so, do they have a standard? Sometimes it is easier to adapt to their requirements.

Do you need to be able to print alpha as well as numeric characters? If so, you must use a symbology such as CODE39 (or TELEPEN).

Is there a company barcode standard already? When possible use the expertise already established in your company.

Reading the Label

There are three general barcode scanning methods; these are: light wands, laser readers, and fixed (mounted) scanners. Light wands require the user to move a wand containing a light source and detector over a fixed barcode. The wand picks up the reflected light and sends the electrical impulses back to the reader box where they are decoded.

Laser readers do not require the user to move the reader over the barcode; instead the reader mechanically scans the barcode many times per second. Again the information is passed to a decoder box. One advantage of the laser wand is that since it can make several scans very quickly, it is possible to have the decoder check and verify the label several times before passing the information to the computer. One disadvantage of the laser readers is that they emit a very pure form of light that can be hazardous if improperly used, or abused.

Fixed readers are designed for use in grocery stores and on conveyor belt assembly lines so that the computer can automatically sense and read information without human interaction. Although these units will work on an HP3000 computer, they are intended to work with computer systems that are capable of processing data in "real-time".

We will therefore concentrate on the first two forms of readers.

Choosing Barcode Equipment

When evaluating barcode equipment, be sure to test the equipment with barcodes representative of those you intend to use. If the barcodes will be printed with a dot-matrix printer, test the reader on dot-matrix printed labels.

If you will be using high resolution barcodes, make sure the reader can read the high-density codes. Does the wand require contact with the label? If it does, then the barcode labels may wear. Does the wand have an open tip? If so, the wand may pick up contamination which requires periodic cleaning. Is the wand rugged? Does the cable have adequate strain relief to prevent breaks in the wires leading to the decoder box?

If you are evaluating laser readers, one of the first considerations is the size and weight to the reader. In addition, most laser readers user a moving mirror to scan the label. These and the laser tubes are more fragile than the standard LED or incandescent light of wands. Are there service and parts available?

Will the reader be used with a CRT? Most readers, such as those made by INTERMEC, Skan-A-Matic, etc., are designed to be placed in-line with the regular CRT. Figure 4 shows a setup of this type. If your programs read the data from the terminal in a character mode manner, then using a barcode reader may not require any modification to your existing software. However if you are using VPLUS (block-mode), most of the readers that are placed between the CRT and computer will not work; instead you must use the special readers that are available from HP (data capture models HP3075A, HP3076A or the HP92911A).

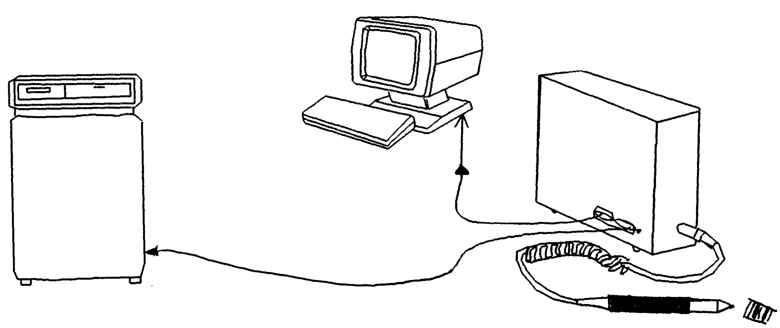
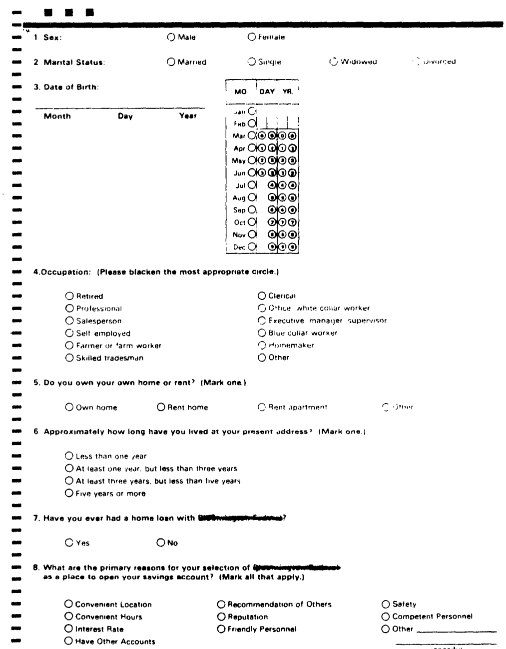


Figure 4. Typical In-Line Barcode Reader Setup

<u>Optical Mark Readers</u>

Perhaps you are familiar with the HP7250 optical mark sense reader that was available for the HP3000 computer. Most people view mark sense readers as an obsolete technology. However optical mark readers can be a very efficient form of data entry since the information is read directly from a form marked by a human, eliminating the need for a separate data-entry person. There are a number of applications which can directly benefit from the use of OMR devices; these are often applications in which the customer fills out a form containing data which is later entered into a computer. OMR can be used to read the customer form directly, thus simplifying the data-entry step and eliminating the need for someone to retype the data. Examples of these types of applications include customer surveys (see Figure 5), and magazine "bingo cards". Other applications which can also benefit include inventory control, warehousing, receiving, many manufacturing steps and quality control.

Mark sense readers are designed to detect the reflected (or absorbed) light from pencil marks as they pass optical sensors. There are two basic types of readers, self clocking (HP7250) and synchronous (SCAN-TRON 5000 series). With self clocking readers the format of the cards generally are designed to match the old IBM keypunch-style cards. Synchronous readers have special timing marks recorded in one margin of the form. The timing marks are generally placed wherever a field is printed on the form. The forms used in synchronous readers can be of varying length and dimensions. It is possible to design a form which is very similiar to forms that you are already using.


1 Sex: Male Female

2 Marital Status: Married Single Widowed Divorced

3. Date of Birth:

Month Day Year

MO DAY YR.

Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec

4.Occupation: (Please blacken the most appropriate circle.)

Retired
Professional
Salesperson
Self employed
Farmer or farm worker
Skilled tradesman
Clerical
Office white collar worker
Executive manager supervisor
Blue collar worker
Homemaker
Other

5. Do you own your own home or rent? (Mark one.)

Own home Rent home Rent apartment Other

6 Approximately how long have you lived at your present address? (Mark one.)

Less than one year
At least one year, but less than three years
At least three years, but less than five years
Five years or more

7. Have you ever had a home loan with [redacted]?

Yes No

8. What are the primary reasons for your selection of [redacted] as a place to open your savings account? (Mark all that apply.)

Convenient Location
Convenient Hours
Interest Rate
Have Other Accounts
Recommendation of Others
Reputation
Friendly Personnel
Safety
Competent Personnel
Other ____________
(specify)

Figure 5. Sample Customer Survey Form Used with OMR

Interfacing Concerns

Most OMR's were designed to read and score test forms; some contain microprocessors which can be used to provide some field editing. In most cases the readers transmit data back to the host computer in a binary format. For example the SCAN-TRON 5000 series readers will send the sequence "'@P@H@D@B@" to the host computer whenever the numbers "12345" are read from the mark sense form. You should be prepared to write a short procedure to control the reader.

When evaluating OMR devices check to see if the reader can buffer the information before transmitting. This feature may make the device

A Information preprinted on form

Mark-sense slugs pre-marked

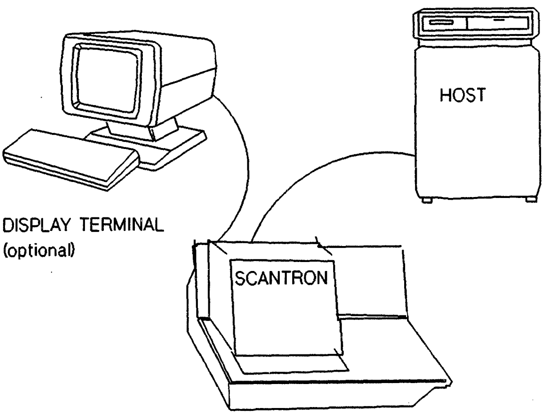
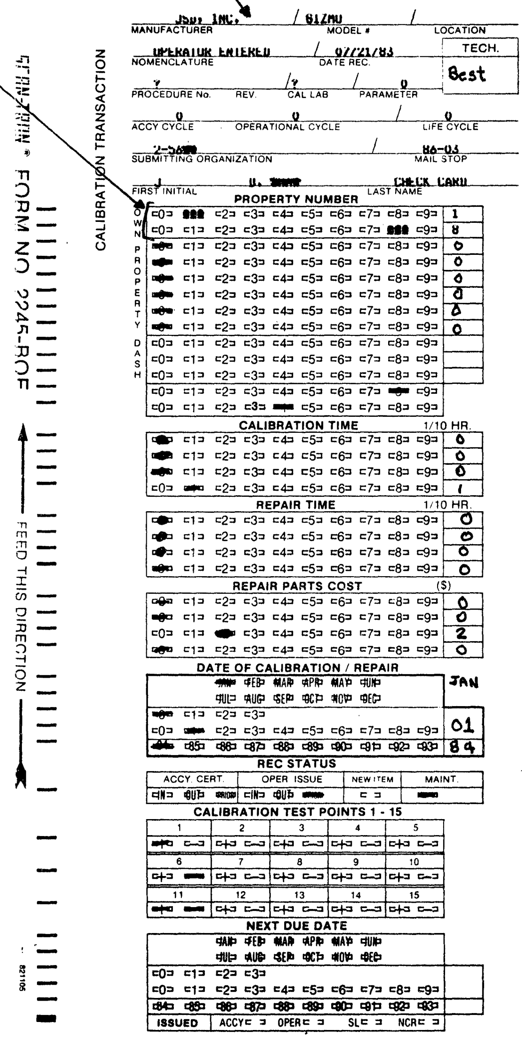


Figure 6. Typical OMR Setup



easier to interface to the HP3000 without missing characters or having data overruns. Check to see if you can program a "trigger" character into the OMR. If so, it may be possible to tell the reader to read the next card whenever an FREAD is requested for that device (see the section on interfacing tips for the HP3000). If you will be using the reader with a CRT, make sure you get a reader that "locks out" the CRT from transmitting data when the reader is sending data. Figure 6 is an example of a typical OMR setup.

Marking the Forms

Most OMR's require that the slugs be marked with a #2 lead pencil. Since the reader senses the reflected (or absorbed) light, any deviation from pencil lead (such as ink) will often reduce the read

Figure 7. Pre-slugged OMR Form

rate or accuracy. It is possible to pre-slug information on the forms using a dot matrix printer. Since most printers do not have a character which fits a single slug, it may be necessary to overprint the desired mark several times, e.g., with a sequence such as "#, backspace, @, backspace, X, backspace, !". See Figure 7 for an example of pre-slugging.

Touch-Screen Data Entry

Touch screens may be regarded as the ultimate in softkeys -- the programmer can assign not only the functions, but also the sizes, number, and placement of the "keys". Two methods of implementing the touch-screen capability are illustrated by the HP150 and the Fluke 1700-series instrument controllers.

The HP150 uses a grid of infrared light beams across the front of the CRT. The user touches the screen, breaking a pair of the infrared beams. The location touched is determined from which of the beams were interrupted. This touch-screen scheme does not affect screen readability, but it is susceptible to dirt covering the infrared emitters and detectors.

The HP150 provides two means to use the touch screen: a line of "softkeys" at the bottom, and a standard touch screen on top where the program must draw boxes to indicate the areas to touch. HP has been slow to provide information on how to design with and use the upper portion of the touch screen. The softkeys are easier for programmers to use because they are accessed in the same manner as keyboard softkeys. In addition, the HP150 allows the program to download the string to be sent back to the host computer when a softkey is touched. The HP150 has a resolution of 27 lines by 40 characters, i.e. every other character.

The Fluke instrument controllers such as the Infotouch 1780 use a touch screen with a grid of tiny wires embedded in it. When it is touched, two wires make contact and create an electrical signal. This type of touch sensing is not susceptible to dirt, but can affect the visibility of the screen if the wires are not fine enough, or if the grid wires are too close together.

The Infotouch 1780 provides sixty touch locations in a screen area of approximately 7 inches by 3.5 inches. These touch locations are roughly the size of one fingertip. The Fluke touch screen is representative of most of the touch screen devices; when the screen is touched a simple escape sequence is sent to the host computer.

Some considerations for use of touch screens in general are: environment, grid size, and ergonomics.

Environment: Infrared scanning is susceptible to dirt. All touch screens will pick up fingerprints, necessitating extra cleaning of the screen. However, they are easier to clean and less susceptible to spilled coffee than keyboards.

Grid size: If the grid is too large, users have a hard time hitting the grid intersections, and it is more critical to space the icons on the screen widely and center them around a grid intersection. If the grid is too small, it is expensive to produce and may have wasted resolution -- the user's finger is only so small. In the case of a wire grid, a grid which is too small can be visually annoying.

Ergonomics: Unlike keyboards, mice, tablets, etc., touch screens require large movements of the hand, and often of the entire arm. For occasional entry (as in industrial process control) this is not a problem. However, for frequent repetitive use by experienced keyboard users, it is often both tiring and annoying, especially when the user must switch between the touch screen and another input device.

Graphics Tablets, Pads, and Mice

Another class of keyboard alternates is exemplified by tablets, pads, and mice. These are table-top input devices requiring a flat work surface.

Graphics tablets or pads are basically digitizing devices which report the location of a puck or pen relative to a grid system embedded in the tablet itself. The movable portion of these devices has some mechanism to select the desired location, differentiating it from the locations traversed in arriving at that point. A pen may sense a stiff pressure or "click"; pucks usually have buttons.

Mice report the direction of motion and (sometimes) velocity across a flat surface. They do not depend on an absolute location, and therefore do not require the complex detection circuitry of a tablet. For this reason, they are less expensive. There are a number of mice, such as the one from Microsoft, which are designed to work with the IBM PC. Perhaps someday we will see terminal emulator packages, such as PC2622 from Walker, Richer & Quinn, adapted to use these mice to provide capabilities similar to the touch screen on the HP150.

As with any keyboard alternate, the utility of tablets, pads and mice depends on the application. They are popular in computer-aided design (CAD) and computer-aided engineering (CAE) applications. For situations of this sort the easy two-dimensional input from these devices lends itself well to the graphical nature of much of the data.

The user acceptance of table-top input devices often depends on the consistency of the user interface. (Remember rule number three?) If the user is constantly required to return to a keyboard or other device between executing tasks with a graphics pad, the result is usually frustration. To combat this problem most pucks and mice have three to four buttons, allowing the programmer to utilize one for location selection and the extras for command input. The result (hopefully) is a usable compromise.

Ergonomics also plays a role in acceptance of these devices. Is the puck, pen or mouse easy to hold? Muscle cramps can result when the object is too large or too small. How far does the user have to move her hand to achieve the desired motion of the screen? No one is happy if the puck must move 24 inches to move the cursor across a 10-inch monitor.

A few performance issues should be considered when selecting a graphics tablet, pad or mouse. How fine is the resolution? In general tablets or pads are capable of smaller geometries than mice. How repeatable is the location? Some pucks "skate" so much that once the user has positioned the puck in the desired location, it is virtually impossible to keep it there long enough to push the button. Thus an incorrect location some short distance away is selected instead of the desired spot; the higher the resolution of the tablet the worse this problem can become. (A graphics tablet such as the HP17623A, often used for digitizing with HPDRAW, has a resolution of approximately 0.006 inches.) Thus programs written to utilize these devices must be tolerant of "near hits".

An interesting adaptation of the graphics tablet is the PENPAD terminal sold by PENCEPT, Inc. It is more nearly a direct keyboard replacement, interfacing to the computer via an RS-232 port. The user lays a form on the pad and uses the attached pen to hand print data in the boxes on the forms. The forms look rather like keypunch forms, with one character entered per box. Unlike some optical character recognition schemes, the PENPAD terminal does not require "perfectly" printed characters; Figure 8 shows some of the handwriting samples which would be correctly interpreted by the PENPAD terminal.

The characters are digitized as they are printed, and the ASCII equivalents are displayed on the screen. The completed form can then be discarded or saved as back-up or an audit trail. This type of data entry is especially well suited for casual users who are normally intimidated by the keyboard.

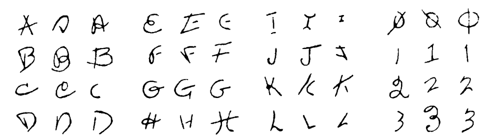


Figure 8. Handwriting Samples Accepted by the PENPAD Terminal

Voice Input

Voice input is no longer a science fiction dream, but a practical reality. There are a number of voice recognition systems available on the market ranging from $2000 to more than $50,000. Voice input systems represent the most technically complex method of data entry discussed in this paper. Typical systems consist of a voice input module, and an output module.

The voice input module contains a microphone, a spectrum (frequency) analyzer, a feature extractor, and comparison logic which classifies and determines which word was spoken. The output module can range from a very simple bell when a word is recognized to a voice output module which acknowledges and prompts for the next input. See Figure 9.

The function of the spectrum analyzer is to break the speech into discrete frequency bands. In addition the spectrum analyzer portion may also attempt to count the pieces of the word and detect the volume of the various frequency components. The technique to perform the spectrum analysis which is becoming the most popular is called "Linear Predictive Coding" (LPC). LPC provides an efficient means to compress speech into a compact format; it reduces the amount of memory required for speech storage. This becomes critical if a voice recognition system is going to have a vocabulary of more than a few dozen words.

The feature extractor takes the output of the spectrum analyzer and further categorizes the speech based on the amplitude. One important function of the feature extractor is to locate the beginnings and endings of words. The feature extractor also attempts to recognize discrete components of speech such as vowels and consonants.

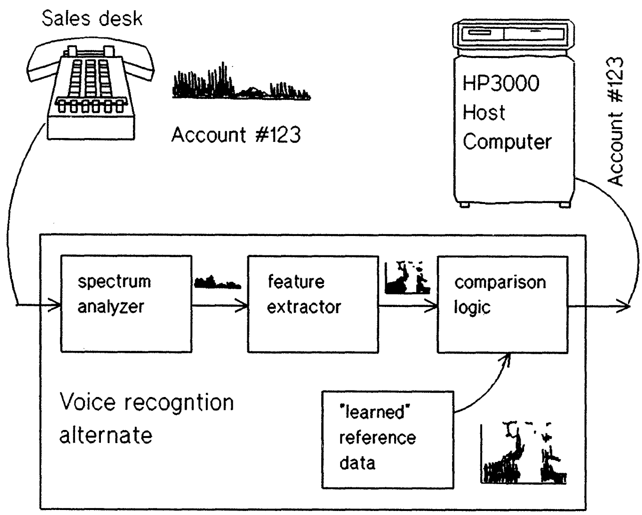


Figure 9. Typical Voice Input Module

The classifier takes the time-ordered output from the feature extractor and compares it to the spoken words which have been trained (recorded) into its vocabulary. Under normal conditions it is very unusual for the classifier to find perfect matches; instead the classifier will score the possible choices. The pre-trained word with the highest score is the word that is chosen.

Voice Recognition Problems

As you can see the voice recognition process is not simple. It becomes even more complicated because the users often try to help the recognition process along by speaking very slowly and clearly. However in normal use they tend to speed up their speech and enunciate less clearly. Most speech recognition systems are designed to recognize single words with a small pause after each word. If voice input is used in an environment which has a high background noise level, the voice recognition system may not be able to find the beginnings and ends of words. Finally, word recognition below 95% [4] often leads to user dissatisfaction and frustration.

Voice Recognition Applications

Voice input is applicable to material handling, quality control inspection lines, many inventory control processes, and is well suited to remote information processing over the telephone. The voice recognition system manufactured by Votan can be interfaced with the telephone system, so that it provides an effective automated means to handle unmanned remote tasks such as credit card verification services. In the paper "One Way to Talk to Computers", Thomas J. Martin noted that in one receiving and shipping application they were able to reduce the manpower and time usage by 90%. [5]

Design Considerations

A typical recognition rate for trained users is 120 - 125 digits per minute. [5] Most speech recognition systems are speaker-dependent, (you train the recognizer). It is possible to buy speaker-independent systems; the penalty is generally reduced accuracy and recognition rates.

If you are using the voice input system in an area with a high level of background noise, then you may need to use a noise cancelling microphone. Operator-originated noise such as coughs, clearing the throat, sneezing, or conversations with other workers are also picked up, and erroneously interpreted as data input.

III. Interfacing to the HP3000

The best complete source on interfacing devices to the HP3000 is the work that Ross Scroggs and John J. Tibbetts did in their papers "Everything You Wanted to Know About Interfacing to the HP3000". [6,7] Virtually everything they covered in their papers still applies to the current version of MPE IV or MPE V. I highly recommend that anyone who wishes to use a keyboard alternate read their papers.

There are four general areas that cause problems when interfacing a keyboard alternate to the HP3000. They are cabling, parity, blockmode, and handshaking.

Cabling

Unfortunately for us users, there appears not to be a standard for whether the receptacle end or the plug end of the RS-232 connector is supposed to be on the peripheral. Cabling becomes even more complicated because the peripheral can be designed for either of two different connection specifications: Data Terminal Equipment (DTE) or Data Computer Equipment (DCE). It may be necessary to make a cable which reverses the type of connector and also switches which pins correspond to the signals. You will need to read the installation manual for the keyboard alternate carefully and perhaps consult HP and the other manufacturer prior to installation.

Parity

The HP3000 treats parity in a very peculiar manner. You must determine whether you have ADCC's, ATC's or ATP's, since each deals with parity slightly differently. For instance our HP3000 uses ADCC's; when we interface keyboard alternates to it we must **make sure that they generate parity, but ignore all parity errors sent to them.** This is because the ADCC's expect the parity bit to be properly set (once the first carriage return is sensed), but do not always generate parity when outputing to the peripheral! Although this difference sounds like a minor inconvenience, it caused a great deal of wasted time when I first tried to interface a SCAN-TRON mark sense reader to our system. Two years later Ross Scroggs published his excellent work (be sure to read it).

Blockmode

With the exception of HP's barcode readers, very few of the keyboard alternates are directly compatible with the blockmode implementation of VPLUS. There are a number of ways to address this problem; one method is to use a product such as PreView from Tymlabs, which intercepts the VPLUS blockmode intrinsic calls and converts them to character transfers. The end result is that you can use your current VPLUS screens with non-HP barcode readers, writing tablets, voice recognition systems and in some cases even optical mark readers. In addition to the product from Tymlabs there are a number of character oriented terminal handlers that are also available for the HP3000 (check the exhibits).

Another approach to the problem is to paint the VPLUS screen and, when information is needed from the keyboard alternate, drop the terminal out of blockmode, read the input from the alternate, and then return the keyboard to blockmode. Mike Casteel, in his paper "Advanced Techniques Using VPLUS", at the 1983 HPIUG [8], talked about some of the ways that you can adapt VPLUS to handle such special cases.

Handshaking

One especially nice feature of the Votan voice recognition system is the ability to program the device-to-computer handshaking rules, thus making communication to the host machine relatively easy. However, not all data devices are this accommodating.

Some of the keyboard alternates may transfer the data in a format which is not directly usable by your application programs. The SCAN-TRON 5098 mark sense reader is an example of these. The mark sense reader detects twelve columns at one time, however you can only send an 8-bit-wide character over RS-232. The SCAN-TRON reader breaks the information read from one column into two binary characters which are sent. Many of the graphics tablets and mice do the same thing. Be prepared to write some special subroutines to handle these devices.

Many manufacturers of barcode readers also sell portable data entry terminals. These devices allow the user to collect information, then come back to a telephone and send the collected information to a host computer. Be careful when you evaluate these units. Remember that the HP3000 does not support type-ahead. This means that before you send information to the computer you must wait for a trigger character to be sent by the computer to the data device. The default trigger device is a DC1. Most of the "off the shelf" portable devices do not support HP's way of sending and receiving data. However in most cases the manufacturers are willing to tailor their readers to your applications for a fee.

Another approach is to write your own programs on the HP3000 to perform the necessary handshaking. This requires that you FOPEN $STDINX and $STDLIST, set the parity and the trigger character (if you want something besides a DC1), and set up an alternate termination character. Virtually all of the set up commands for $STDINX and $STDLIST are done by using the FCONTROL instrinsic (the best reference available is Ross Scroggs' article) [6]. When you are ready to read information from the data entry device, call FCONTROL to set a terminal time-out limit (in case the data entry device does not respond) and call FREAD. Everytime that FREAD is called it will send a trigger character to the data device, which should respond by sending information to the HP3000. If the device does not respond in the appropriate amount of time, then a time-out will occur. After a finite number of time-outs the transfer program should assume that the data device is no longer functional. This technique works quite well if the data device is capable of listening for a trigger character. The Radio Shack Model 100 computer with a barcode wand is an example of a device which is low cost and capable of being programmed for easy data collection and transfer to an HP3000 computer.

If the data entry terminal is not capable of listening for trigger characters, then it may be necessary either to buy a interface box (such as a type-ahead engine from TELAMON) or to request an FREAD large enough to hold all of the information from the data device in one shot. The latter approach implies that you will be using the paper tape mode of FREAD. There are some problems with this technique: a limited amount of information can be sent (about 32,000 characters), ADCC's do not support paper tape mode, and if the information is being sent over a modem, it is unlikely that you will have a sucessful data transfer without a lot of incorrect characters added into the data stream because of telephone line noise.

IV. Summary

Is there an alternate for you? Possibly. (One common fallacy is that if the alternate cannot be used for all input, it is not worth using at all.) Any alternate should be selected carefully. Once you have evaluated the keyboard alternates and made your choice, take one last look, ask a few last questions.

Does it save time? Carefully estimate the time saved by switching to a machine-readable form of data entry. Unlike dollar savings, time savings are not necessarily cumulative; if you save one second per entry by using a barcode reader instead of the keyboard, it does not mean that the user will spend the time saved in a more productive manner.

Does it improve your data accuracy? This is more complicated to estimate. The literature provided by the barcode manufacturers suggests that a barcode read error only will occur once in every four million reads. If you had about 200 data errors per week with keyboard input you might assume that your errors would drop to only one every several years. However, if you must connect the barcode reader or terminal to the host computer over a phone line (as is common) then a more reasonable error rate estimate is one error every ten thousand characters. This means one or two errors a week from noisy phone lines. Using a machine-readable form does not guarantee that human errors are eliminated; for instance, the result is the same whether a user writes down the number of the wrong item onto a keypunch form, or scans the wrong item with a barcode reader. Despite the caveats, the real benefit from switching to a machine-readable form of data entry is the improvement in data accuracy. In the example, this means only having to resolve a few mistakes a week, instead of two hundred or more. An important side-effect of the reduced error rate was the users' increased confidence in the computer.

Is it practical? Some of the neatest ideas are impractical. One of the experiments I tried was the use of voice output. The idea was to use barcodes for data entry, CRT for the users to look for data, and a speech box to "read" what was displayed on the CRT, thereby freeing the user to perform some other task while listening. A clever idea (or so I thought); however, too many of the users found the machine's voice distracting. More effective use of a voice output would have required changing all the application programs so that only the most important information was spoken.

Is it too complex? Do not overcomplicate the system! Remember the goals of using machine-readable entry are to make the data more accurate and collect it faster, at a lower cost. However if you install a sophisticated system which requires special skills and extensive training, you may lose more than you gain. If the system is too complex the users will avoid it! We experienced this when we installed a barcode printer. Controlling it from the computer required switching several switches, typing a number of special commands from the computer operator's console, then running a special program. It was easier (for our low volume), though probably more error-prone, to type the numbers directly on the printer's keyboard.

If the answers to enough of these "last look" questions are "yes", then proceed. With proper planning and selection, keyboard alternatives can provide many advantages.

References

[1] Montgomery, Edward B.; "Bringing Manual Input into the 20th Century: New Keyboard Concepts"; Computer; Volume 15, Number 3; March 1982; p.12.

[2] Good, Michael D., Whiteside, John A., Wixon, Dennis R., and Jones, Sandra J.; "Building a User-Derived Interface"; Communications of the ACM; Volume 27, Number 10; October 1984; p.1032.

[3] Allais, David C.; "Which Bar Code Symbology?"; Bar Code Symbology and Applications; Northcon83, May 10-12, 1983; Portland, Oregon, USA

[4] Welch, John R.; "Automatic Speech Recognition -- Putting It to Work In Industry"; Computer; Volume 13, Number 4; May 1980; p.65.

[5] Martin, Thomas B.; "One Way to Talk to Computers"; Spectrum; Volume 14, Number 5; May 1977; p.35.

[6] Scroggs, Ross; "Everything You Wanted to Know About Interfacing to the HP3000, Part I"; Proceedings of the 1982 HPIUG Conference; San Antonio, Texas, USA

[7] Tibbetts, John J.; Everything You Wanted to Know About Interfacing to the HP3000, Part II"; Proceedings of the 1982 HPIUG Conference; San Antonio, Texas, USA

[8] Casteel, Mike; "Advanced Techniques Using VPLUS"; Proceedings of the 1983 HPIUG Conference; Montreal, Quebec, Canada

Bibliography of Related Material

Crane, H. D. and Savoie, R. E.; "Special Feature: An On-line Data Entry System For Hand-Printed Characters"; Computer; Volume 10, Number 3; March 1977; p.43.

Hacker. Edward; "Bar Code in Manufacturing"; Interact; Volume 3, Issue 7; November 1983; p.44.

Kaplan, Gadi; "Words into Action: I"; Spectrum; Volume 17, Number 6; June 1980; p.22.

Norman, Donald A.; "Design Rules Based On Analyses of Human Error"; Communications of the ACM; Volume 26, Number 4; April 1984; p.254.

Shneiderman, Ben; "Human Factors Experiments in Designing Interactive Systems"; Computer; Volume 12, Number 12; December 1979; p.9.

List of Manufacturers

This list of manufacturers is provided for your convenience and lists the makers of products mentioned in this paper. It is not an exhaustive list of manufacturers of keyboard alternates. The authors recognize that non-US manufacturers are not represented here, but have elected to include in this paper only products with which they are familiar.

Barcode Readers

Hewlett-Packard Company
P.O. Box 10301
Palo Alto, CA 94303-0890 USA

Interface Mechanisms, Inc. (INTERMEC)
4405 Russell Road, P.O. Box N
Lynnwood, WA 98036 USA
(206) 743-7036

Skan-A-Matic
P.O. Box S
Route 5 West
Elbridge, NY 13060 USA
(315) 689-3961

Optical Mark Readers

Hewlett-Packard Company

SCAN-TRON Corporation
P.O. Box 45706
8820 South Sepulveda Blvd.
Los Angeles, CA 90045 USA
(213) 776-4811

Touch Screen Devices

John Fluke Manufacturing Co.
P.O. Box C-9090
Everett, WA 98206 USA
(206) 347-6100

Hewlett-Packard Company

Graphics Tablets, Pads, and Mice

Hewlett-Packard Company

PENCEPT, Inc.
1710 South Amphlett Blvd., Suite 128
San Mateo, CA 94402 USA
(415) 341-5688

Voice Input Devices

Votan
4487 Technology Drive
Fremont, CA 94538 USA
(415) 490-7600

Interface Devices and Software

TELAMON
(formerly The Type Ahead Engine Company)
534 Rosal Avenue
Oakland, CA 94610 USA
(415) 835-5603

Tymlabs Corporation
211 East 7th Street
Austin, TX 78701 USA
(512) 478-0611

Walker, Richer & Quinn, Inc.
Lake Washington Place, Suite 201
1914 North 34th Street
Seattle, WA 98103 USA
(206) 634-0503

About the Authors:

Dennis Heidner received the BSEE degree from Montana State University, Bozeman, Montana. He joined the Boeing Aerospace Company in 1978 on a special project to study techniques for management of general-purpose electronic test equipment. Based in part on that review of management methods used throughout industry, the Boeing Aerospace Company Test Equipment Management group received approval to buy an HP3000 computer system. Mr. Heidner was responsible for the system requirements, planning, design, and program implementation. In May 1980, the Test Equipment Inventory Management System (TEIMS) became functional. Mr. Heidner has written "Transaction Logging and Its Uses", presented at the 1982 HPIUG. He co-authored two papers, "Transaction Logging Tips" and "IMAGE/3000 Performance Planning and Testing", which were presented at the 1983 HPIUG in Montreal. He presented the paper "Disaster Planning and Recovery" at the Anaheim 1984 HPIUG conference. Mr. Heidner is a co-author of "The IMAGE/3000 Handbook", published by WordWare, Seattle, Washington. He has also written technical articles published in several magazines.

Amy Heidner is a licensed professional engineer with a degree in electrical engineering. Ms. Heidner is employed as a design engineer at John Fluke Manufacturing Company, makers of electronic test equipment including the 1700-series instrument controllers with touch-sensitive screens.Ms. Heidner has programmed on various computers including the HP2100 series, forerunners of the present HP1000 computer family, and a number of Digital Equipment Corporation (DEC) computers. She was involved in the editing of "The IMAGE/3000 Handbook", published first quarter 1984 by WordWare.

MANAGING INFORMATION SERVICES IN THE EIGHTIES

John F. Lehane, Ph.D.
San Jose State University, San Jose, U.S.A.

Summary

The information evolution has progressed through three distinct time periods. The first period began in the forties and ended with the sixties, and it emphasized the processing of structured data flows. The second period of the seventies emphasized the start of data base and data communication tecnologies within structured data formats. The third period of the eighties emphasizes the integration of all information services (data, image, text and voice) within the concept of effective organizations and unstructured data flows. The information evolution has been driven by the ever changing technological products of the vendors and not the informational requirements of the users. The transition of informational products has created considerable stress in the management offices of the organization. The information evolution has not modified the basic principles of management, but it has caused a great deal of confusion and problems in the development of organization structure and placement.

The purpose of this paper is to analyze information as a resource of management and to determine the potential impacts of information services on organization structure and placement. The main contentions are that management needs to view information as a resource and to manage it as any other resource of the organization. The responsibility for efficient and effective information services lies with management, and management has to develop the structure and placement for its objectives in a cost effective manner. An organization's structure and placement are critical issues in the fulfillment of the managerial responsibility with respect to information services.

Introduction

The information evolution has been with us for approximately forty years. These forty years have seen the information cycle move from a very ill defined concept of data manipulation to a rather sophisticated need for full information services. Hardware and software products of the vendors have pushed the information evolution to faster and more accurate and reliable information systems. The users of information services have been forced to move from a vague appreciation of data processing to management information systems to decision support systems and finally to information resource management. Managers have been forced to provide more information services in order to satisfy the ever increasing demands of the user community. The focus has been on more information services and new technology. The idea was that the new technology was the means of resolving information problems and could provide fantastic opportunities for those organizations willing to keep current with the latest products.

Organizations in the eighties realize that their problems are not being solved. In fact, their expenditures for information services keep increasing and still the demands for information services are not being satisfied. If anything, the more computer literate users and managers know that new technology offers increased opportunities for information services at a fraction of the cost and inconvenience of the present systems. However, their attempts to use this new tehnology have been restricted to short term solutions to their long term problems of information services.

The information service group has been the central service area in many organizations for years and has responded to new information requests as an inconvenience. Their attitudes have been conditioned by their concepts of efficiency and not effectiveness. The emphasis was to obtain the maximum amount of processing for the least time and cost, to keep the information requests routine and to develop applications that could be run over and over again at the least cost.

Early Days of Computing

The early days of computing were in the forties. Attempts were made to mechanize the data manipulation cycle using the mechanical and electrical devices of the times. The military was willing to pay for the attempts to speed up the calculations of tabular data. The applications were well defined and fully structured. The computers of the times provided a means of producing results in a relatively short time when they worked. Unfortunately, due to their mechanical and electrical nature, they had a tendency to break down more than was expected.

The early users of information processing were mathematicians and physicists. The early users were truly pioneers and emphasized a certain mystery to the process of converting data into a usable product called information. The times called for experimentation and trying to learn what this new technology had to offer. The idea was to provide a stimulating climate since the tools were new and it was not known just what could be done with the technology.

Management was delegated to a mathematician or physicist who was interested in the broader issues of the application of the technology rather than the immediate appeal of something new. The early pioneers did not need a lot of direction since their tasks were well defined and it was up to them to decide how the new technology could be harnessed to satisfy their needs. The business community was busy trying to keep their organizations moving and to satisfy the immediate demands for their products and services. Their information systems were manual and their procedures were defined. The rapid frowth of the economy during the end of the forties forced mechanization of information systems.

Computing in the Fifties

The fifties brought change and the information evolution was identified by accounting departments at a few organizations as a means of finally automating their service areas. Usually, dedicated persons would step forward and argue for mechanization based on some previous experience that they had had using the information technology of the forties. It seemed apparent to them that computers of the day could be used to do the routine functions of a business provided that the procedures were defined. The accounting area with its defined area of procedures offered the first opportunity. In many cases, a data processing group was established as part of the accounting department. Their operations were to use the computer to process data for accounting applications. Their first attempts were concentrated on payroll and inventory control. Even these applications required a great deal of effort as the organization was not ready for the mechanical and electrical problems of the computers. The equipment keep breaking down, and it required enormous efforts to complete their assigned work in a specified time frame. The culprit was the computer, and all failures could be blamed on this newest machine. Data processing became the catchall group for all data manipulations at least in the accounting and budgeting areas.

The end of the fifties emphasized the beginning of high order programming languages and the transfer of data processing services from the early pioneers and dedicated users to a less dedicated but educated group of engineers and businesspeople. These people saw the information evolution as a means of reducing the repetition of their jobs. Their emphasis was on change, and by this time the technology was beginning to support them with a variety of products and services. Management felt confused by the power of computers and still tried to ignore the ramifications of the information evolution on their organizations. Users felt confused and a few individuals and organizations led the evolution to the new applications of this interesting but maintenance plagued technology.

The accounting applications, like the scientific applications of the day, had the distinguishing charateristic that the problems were well understood from a manual point of view and had been used and tried for a number of years. The procedures were in place and it became a task of adding the information technology to mechanize the processing of the procedures. The computer was viewed as a replacement for the calculator and the adding machine.

Computing in the Sixties

The business use of information processing did not gain wide acceptance until the sixties. By this time, the early users had established the feasibility of business applications on computers and a number of vendors identified that the business applications of computers were greater than the combined technical and scientific applications. The market was being established for the business use of the computerized information processing. Again, the applications were well defined and followed well understood rules of accounting and inventory management. The market experienced a fantastic growth in the number of vendors offering a variety of products which seemed to satisfy the processing of conceivable business applications. Organizations developed information processing groups to specialize in this new area of service.

The sixties was devoted to a head long attempt to bring as many of the information needs of the organization into the information processing group as possible. The shortage of trained personnel and the lack of educational programs in the information processing area forced organizations to concentrate on a centralized service. These centralized groups were established to address the organization's needs for information services, but, in fact, they were established to satisfy the more immediate needs of the accounting group and thus were organized within the accounting department. The accounting department realized the terrific capabilities of this new technology and saw it as a means of continuing to control the information processes of the organization. This early decision affected the eventual development of information processing in each of the organizations. The information needs of the accounting department were emphasized at the expense of the total organization's need for information. The sixties was a time of specialization by function and centralization by service.

A number of new ideas appeared as possible alternatives to routine batch proceesing applications of the day. They were data base and data communications. These new ideas were to open the doors to unlimited opportunities in information processing. The promise was great, but the realization was something else. Many organizations were committed to large inhouse systems and they were busy trying to utilize what they had before becoming involved with new information products. The information group was busy trying to use the new technology products and maintain a level of productivity.

Computing in the Seventies

The seventies did not show much promise until the advent of the inexpensive microcomputer and packaged software. Prior to these ideas, products were custom made for a client and as a result it required a long and tedious process to accomplish the mechanization of information services. The larger organizations could afford the cost, but the smaller organizations were ignored. A number of minicomputer vendors identified that their products offered an inexpensive solution for some organizations. The minicomputers of the day were modified military and scientific computers that were adapted to satisfy the business market for inexpensive processing. The software was identified as a potential bootleneck to the continual evolution of information processing. A number of new ideas and concepts appeared as means to address the issues of information services. The idea was again that technology could resolve the difficulties of information services and management did not have to participate in an active role. The technology would come to the rescue.

A few organizations continued to develop data base technology, and data communications and timesharing exploded. Organizations allowed their user community in some cases the option of going outside the organization for specialized services. If an organization could find an application that was not readily available on the inhouse system then it could use the outside service. The test was on availability and in some cases cost became important. The users were given the options of deciding what they wanted in some specialized areas. The data processing group was still centralized and data communications seemed to emphasize the benefits of a large mainframe. The user community was experiencing time delays, but at least it offered an alternative to batch processing and the long turnaround cycle of data processing.

The microcomputers and VisiCalc changed the organization's use of information services completely. A user could purchase a relatively powerful processor very inexpensively. The user could control its turnaround and most importantly resolve its own problems without having to consult the information specialist. Inexpensive processing power plus packaged software presented a new challenge that few if anyone understood in the seventies. It was interesting technology but not serious computing.

A few organizations decided to use minicomputers to replace mainframes. The problems were limited personnel and how to best fit the two together. The applications were still structured and the information service was very routine and extremely nonresponsive. Management wanted efficiency and new technology was seen as the means of providing the information services at a cost that the organization could afford. Alternative technologies were becoming to be recognized as part of the same information cycle. Word processing, text processing, image processing and voice processing were identified as part of the total information process. Each of the technologies was used by separate organizations in a number of different applications without too much thought to system integration. Again, management was caught with advancing technology at a time when it was trying to organize for data processing, but the technology was moving to the next step without too much direction from management.

Computing in the Eighties

The eighties started slow. The concepts of microcomputers and packaged software gained acceptance with every new announcement that emphasized ease of use and a specific application. The software designers were using the enormous capabilities of the microcomputers to design very sophisticated user oriented information products. The ideas of centralization and large mainframes were questioned because of the lack of responsiveness and the inexpensive microcomputers. Users discovered that many of their applications would run very nicely on these limited microcomputers. The important factor was that it would get done in a reasonable time period without having to subsribe to the lead times of the data processing group.

Users in the eighties have been asking why is it so difficult to obtain what they consider an easy request for information. As users receive more information regarding information services they wonder why an organization was willing to accept the dictates of the information processing group and not the requests of the users. Management is faced with a better educated group of users that are beginning to understand information processing technology and are not willing to accept excuses. Their demands are now, and they want answers in a reasonable time frame. Delays and equipment failures are seen as deficiencies in management. Management has responded to the changing information environment by decentralizing some of the information processing capability and introducing microcomputers for limited applications. The concept of distributed systems and integrated software packages has become the technological ideas for the eighties. Once again, technology will save the organization.

Organizations and Information Structures

The problems are real. The users of information services have been suffering for years. Every new technology was supposed to resolve the difficulties of information processing. Unfortunately, the solution has been to continue to mechanize and produce more information without first identifying the end uses. More technology supposedly will cure the problems of managing the organization's information flows. The typical response to a request for information services is eighteen months and $100,000 for even a simple application. This lack of resonsiveness has led the user departments to microcomputers and packaged software products. The era of the turnkey systems and comparatively user friendly interfaces has identified the potential for information services. Users are rushing into the purchase of an interesting application that provides a service in a timely manner without any interference from the information processing group. This means that the users are becoming more computer literate and see the information service as an organization resource rather than a highly structured and highly restrictive possession of any one group.

The initial introduction of microcomputers and packaged software seemed to answer the problems of responsiveness and service. At least, the end users now had control regardless of the outcome. If they made a mistake, it was their mistake and they could take corrective action in a relatively short time. The power of computing for the more traditional data processing activities plus word processing applications led the users to understand that the surface for end user software was just being developed. Their response is to move their information requests to a higher level of responsiveness and demand access to the centralized files of the organization. Now, the problems begin to reappear again. The end user determines that the inexpensive microcomputer does not communicate with the central system. Then, it becomes the responsibility of the staff of the information service group to explain the problems. The experiences of microcomputers and packaged software have focuse a new realism on what can and cannot de done on computerized systems. Management has been forced to understand that the information service area is not isolated from the organization. Information is central to the organization and it must be organized and controlled, controlled in the sense of what it is trying to accomplish and not some arbitrary and unstated objective.

From the start of the information evolution, organizations identified that the information service area was concerned with systems, development, maintenance and operations. The system side was concerned with maximizing the use of the overall system and developing broad applications for general use. The development side was concerned with new applications, and the maintenance side was concerned with maintaining the present library of application software. The operations side was concerned with the physical operations of the system as well as the execution of the applications. The sixties brought timesharing and data base. A new group was identified as data communications specialists and data base specialists. The seventies brought a need for quality assurance and production control. A new specialty was determined as the quality control or production control specialist. Now the eighties are emphasizing distributed systems and a continual and increasing demand for data communications and telecommunication services.

The organization is faced with the problems of how does it best organize its resources to satisfy the information flows considering the changing technologies and the demands of the users. The basic structure of the information group has been traditionally systems, programming and operations. The new specialties of data communications, data base and quality control are attempts to recognize the diverse information needs of users and new technology. Specialization is seen as a means of justifying different salaries and education levels. The real question of how to best satisfy the customers given organization objectives and constraints is not being addressed. I feel that the key to a particular organization structure for information services must consider the long term organization's objectives, management styles and policies. The question is not to consider a general model that seems to apply but instead to find a model that fits what an organization is trying to accomplish. These accomplishments need to be written and stated as part of the information plan for the organization. An organization as a minimum needs a central information staff to process requests for information services that can only be satisfied from a central source. Examples of this are the financial records of the organization and personnel records. A support staff needs to be in place to provide continual user training and technical expertise when requested. as an organization grows and develops or acquires additional software it will need to continually keep people up to date on changes and, of course, initial training for new hires.

The central information processing staff will decrease in numbers as the number of microcomputers and packaged software products increase. This in itself is not bad provided that provisions are made for the support services that will be needed to make all these items productive and fit within the objectives of the organization. The organization will find itself being asked to serve as a facilitator of information services and not necessarily a doer of services. This should present an interesting change for the traditional information processing group. The transition to a true service orientation will require that an organization recognizes the skills of integrators of information services and not specialists. The specialists will be needed to maintain the systems and identify problems before they develop. Support groups of application users will be established to answer specific requests for software packages and to aid in their integration into the organization's information services. If an organization decide to move into distributed systems with distributed support services, it will have to pay an additional cost for the duplication of these services. Duplication by itself is not bad if it causes the organization to be more effective. As a minimum, a central group will need to police the distributed services to make sure that they fit within the organization's plan for information services.

What does management need to do in the eighties to affect changes in their organizations? I feel that the first step is to have a clearly defined information service plan for the organization during the next five years. Plans are not carved in granite, but they need to be written and discussed among the levels of management. Plans can be modified as time changes, an organization should not be locked into a given plan. The plan should emphasize what specifically the organization expects from the information service area. The actual structure of the information service group should be left to the individual in charge.

The education community at the university level is beginning to recognize the need for better trained generalists and technical specialists. This means that as the organizations begin to integrate and catch the next wave of technology the universities will be graduating students knowledgable in the technologies of information services. Hopefully, for once, the technology and the users and generators of the information technology will come together to view their tasks as mutually binding. It should be interesting to see how the twain shall meet and the degree of success of the matchup.

Can organizations wait for the matchup? No, I think that organizations have to begin today to prepare themselves for the change. The realities are here. Each organization needs to consider the technology that best supports its operations. Having decided that then it becomes a task of structuring an organization to satisfy its stated organization's objectives. This becomes a task of balancing resources. What tasks will the information service group be resposible for and to whom? Given the statement of tasks, it becomes a management exercise for the individual in charge to find the best means of satisfying the statement of tasks within a specified resouce allocation.

Organizations and Placement of the Information Service Group

Placement refers to where in the organization the information service group is positioned. By now you are saying why not abolish it altogether. The reason that this is not an acceptable alternative is that there will always be a need for some type of a central information group for those tasks that can only be done centrally. The actual placement will change and certainly tasks will need to be reorganized given the changes in information processing. A brief summary of what has taken place in organizations regarding placementis very helpful.

Initially, the information service group was part of the accounting department. Some of you might say that this is still true in your organizations today. The placement of information processing established the extent of authority that management assigned to the group. Consistent with the intentions, the information processing group knew that it was placed in the accounting area to concentrate on accounting applications and only when everything else was done should it consider the nonaccounting applications. These organizations may have in fact established very automated accounting systems. Accounting management was happy and used the opportunities to impress the other managers with the advantages of information services. Some organizations today are proud of their accounting operations and can provide various reports to emphasize the efficiency of their operations. However, organizations do not survive based on accounting information. Marketing, manufacturing, personnel, and engineering were identified as second class information users. Thus, the placement of information services caused more problems than it resolved. Information processing needed a sponsor initially and accounting was probably a good choice as any. The fact that accounting did not allow the information service group to grow as the organization changed resulted in its downfall.

The second choice was to establish the information service group as a separate department. As a separate department, it was responsible for its operations and it was forced to justify its budget. the group was funded for tasks on an ongoing basis and it was required to compete with outside services for the inhouse development applications. Depending on the setup, the separate department could have wide appeal and satisfy the inhouse information services. Unfortunately, it seemed that in many cases the inhouse department became a captive support organization for those groups or departments that had the largest budgets. New development work was limited to those applications with the biggest payoffs. Integrated applications were not stressed unless upper managment had the foresight to understand the operations and the need to integrate information services.

The third choice was to place the information service group as a staff group answerable to the Chief Executive Officer (CEO). The concept was that information is so important to an organization that it needs to be accessible at the highest levels in the organization. The problem with this is that now you have identified information services as a staff function and placed it on the CEO level, you have also provided for its downfall. As the economy turns and profits decrease, it becomes necessary to reduce overhead and the most expendable group is staff. The information service becomes a victim of the vagaries of the economy as well as the CEO.

Now that we have eliminated the traditional placements for the information service group, where do we go from here? I think we start all over again and begin to realize that information is a resource of the organization and we have to decide its worth based on its ability to influence decision-making. Who decides this? The answer has to be top and middle management. The managers that influence decision-making will decide what they are willing to spend for information services. I feel that realism has to be expressed in terms that executives can understand. Dollars are the standard and will be used to jusify information services. Once the decision has been made to spend a certain number of dollars, then it becomes a budgeting exercise on how to most effectively use those dollars for information services.

the eighties are emphasizing the integration of information services and with it a continual need to gain maximum benefits for the dollars spent. How can this be accomplished? Planning provides the answer and the use of the technology. Organizations will need to be centralized with respect to key information services and also decentralized for local reporting needs. Centralization works best for control and cost and decentralization works best for responsiveness. The means of gaining the benefits of both is to concentrate on that part of the information service that is best performed by each. We need centralized services for middle and top management reporting, standards and regulatory reporting. We need decentralized services for the lower levels of management and the day to day operations of the organization. The total information service will need to be administered to serve the total organization's objectives and reflect the information requirements of each of its parts.

The centralized part of the information service will be set up as a separate department serving the whole organization and also be resonsible for the integration of decentralized services into the centralized service. The services will be expected to keep their operations current with the technology and define their tasks within favorable benefit-cost relationships. The idea is to recognize that information is a resource that needs to be handled as any other resource of the organization. Namely, planning and budgeting is required to accomplish stated objectives. The objectives need to be stated and understood. Performance will be measured against the attainment of the objectives. Does this mean that the planning procesd will dominate the information service group? No, it means that after forty years of growing it is about time that management realizes that information is a resource and needs to be managed.

Organization Structure and Placement in the Eighies

The eighties have produced a number of drastic changes in the information processing area. These changes have for the most part concentrated on the use of new technologies. Organization problems have been neglected at the expense that technology is changing so quickly we need not plan for tomorrow. Unfortunately, tommorrow is here today. Microcomputers seem to offer a number of benefits and for once reduce the pressures on the central information processing group. Users find very quickly that microcomputers are great for spreadsheets, word processing and limited relational data bases but extremely restrictive for the more interesting large applications. Worst yet when they want to access the large central mainframe, they find that they need to play all types of games to operate as a terminal on what they have been told was a uswer friendly system. The old problems of hardware and software compatibilities reappear.

Organizations are facing the integration of information services without a viable model for information services. We have management theories and organization theories and we even have something called system theory. Unfortunately, the precise answers as to how to best manage an information service organization are not readily available. I feel that an answer is in the use of common sense and deciding that the needs for information services are different in the organization and most importantly that we can not plan for all applications. We need flexible systems and systems that are build to be used by large groups of users. Fortunately, technology is coming to the rescue here. We need the diversity of distributed systems and the standards of centralized systems so that the information can be passed from user to user without having to reenter data over and over again. Security and auditability have to build into the information service from the start.

The best plan for organization structure would emphasize the tasks faced by the information service group for different time periods. We need to do today the tasks that are assigned but also to plan for the future information services. The technology will continue to change. Organizations need to consider which of the new technological products offer an opportunity to improve their information services to the users. Organizations will need to evaluate the actual performance of their systems and forget the marketing hyperbole. What counts is what can be done and not what is promised.

Information service groups will be asked to respond increasingly to unstructured information requests as the users understand that the new information technology makes it possible to obtain information in a variety of formats in a reasonable time period. These requests will cause the information service groups to become more end user oriented if they are to survive. The organizations that make the change will begin to realize that information processing is not just programming but a bringing together of the best technology to satisfy the information requests of the organization within cost, time and quality. The new horizons hold great promise provided that management realizes the need for flexible planning and the integration of information technologies.

Biography

John F.Lehane, Ph.D.
is a Professor of Marketing and Quantitative Methods at San Jose State University, San Jose, California with responsibilities for the Information Resource Management concentration and the Business School Computer Labs. He has approximately twenty-five years of information processing experiences in the commercial, consulting and academic areas.

THE NATURE AND NEEDS OF HUMAN BEINGS

E. R. Simmons, Ph.D.
Protos Software Company, Austin, U.S.A.

Summary

The author looks at the data processing business and observes what appears to be a lack of concern for the general physical, psychological and emotional welfare of the individual worker.

Instances are cited that tend to support this thesis and then suggestions are made as to how to counteract what is seen as a detrimental force in the lives of professional computer scientists.

A few months ago, a friend of mine in the dp world talked with me about how he came to be MIS Director for his company. It was an interesting, enlightening and provocative discussion.

It seems that his company had encountered a great deal of difficulty in their search for someone to fill that particular position. They had tried data processing experts but discovered they were lacking in management expertise. They tried managerial types, but they were lacking in data processing experience. Some of them could never quite grasp the company's general business philosophy as it related to customers and employees. It was a veritable thorn-in-the-flesh.

Finally, the company decided to look among their own trusted employees to determine if there was any one among them who had a combination of managerial potential, data processing knowledge and who would not only have a good understanding of the company's philosophy, but be attuned to it. The search was expected to take several months and one person was selected to evaluate various recommendations that were made by those in the company who were apprised of the need and encouraged to make suggestions.

The time eventually came when the person conducting the search tapped my friend on the shoulder and told him that he was the man they needed for that particular position. He told me that when he was selected, he regarded it as quite an honor and thanked his colleague for his expression if confidence in offering to recommend him to the company. The response of the selector was "Don't thank me. One year from now you will be smoking too much, drinking too much and in all probability be divorced". This was somewhat shocking to the candidate and he didn't know quite what to make of it until about a year later, he said, when he found himself smoking too much, drinking too much and on the verge of divorce.

Fortunately, since he had been forewarned, it dawned on him that his work had taken possession of him and that he would have to make some drastic changes in order to save his marriage and family. That, he was able to do; however, he realized that he might not have had much chance to make the turnabout had he not had a strong marriage to begin with. He had been married to a helpful and loving mate who had borne two children and held marriage to be a sacred commitment. As a result, they were able to communicate about their needs and the needs of the family and bring about the desired changes.

I wondered, then, what it was about the computer world that produced this unhealthy attitude manifested by the company representative in the above story. So, I began to talk to others for whose accomplishments I had a great deal of respect. I found a strong influence among them for this type of thinking: that is, if I am going to be successful in this business, I have to forget everything else in life and compete with all of the strength I can muster. If that means sacrificing friends and family, so be it.

I found that many of my acquaintances in the industry were plagued with problems for which they had no solution. One young lady from the midwest said she had reached the point where whe was not suitable company for other humans. She was under constant pressure from her company to keep computer services rolling smoothly. She was spending long hours at the plant and then, since that didn't seem to be enough, she had been equipped with a beeper so she could be contacted any time of the day or night in the event of a system crash or other crisis. It got to the point that, during her free time, she could only go to her country hideaway and communicate with her animals. She was so weary and her energy level was so low that she did not have the stamina to make the effort required to be polite company for other human beings.

Another dp woman from Texas found that her husband felt threatened by the amount of time and energy her job required. Of course, her family suffered because of her physical absence, but even when she was with them, they felt shut out of her life because she was so preoccupied with thoughts of her responsibilities at work that she was unable to devote herself to handling the day to day needs of her husband and children. It got to the point where there was more or less an armed truce among members of the family. No one dared ask anything of another member for fear of provoking an argument, so they wtched TV and seethed inside. Again, they were able to recognize what was happening in time to take some corrective measures.

Another friend told me of a situation that had developed in a company for which he had worked for a few years. It seems that there were 10 married men who were assigned a task that took two years to complete. At the end of the two years, eight of the men were divorced. That is a rather high rate to be pure coincidence even in this day and time.

Yet another person who worked for a Canadian company in Canada told me that there had not been a single day during a two-year period when he was not called in for company business. That included his vacation time.

At about that time, I began to read articles in several different publications, mostly professional, regarding turnover and burnout in the dp shop. As I got more into my study, it became apparent to me that something was missing in the approach to the computer age. That something was concern for the humanity of man. The nature and needs of human beings were being ignored. Perhaps ignored is not the right word. It appeared that people were not aware of their own nature and needs. They were just uninformed about what their bodies and minds needed to live life to its fullest. This did not seem to be a matter of mercenary, calloused business people trampling on workers and their rights. There was obviously a lack of understanding on the part of the workers themselves when it came to their own personal life.

Much of this lack of understanding came about because of the computer revolution itself. It appeared that those responsible for production in the world of computers were influenced by the performance of computers to think of human beings in the same way they thought of their computers. Men and women came to be looked at in the same light as machines. A computer can be made to work faster this year than last; the same should be true of humans. Why can't human beings become more efficient and more productive in the same way that computers have become faster and more productive?

This idea was shared by worker and manager alike. Those who were responsible for the computer's production began to worry about themselves. They were geared to quick operations and greater production.

They just had to find the keys that would enable them to function in a more complete manner. Coincidentally, about this time, there was a sharp rise in enrollment in seminars that taught about handling stress in the workplace and seminars that emphasized organization and motivation. There was a sense of urgency about the simplest business functions. I began to notice this even among people who really could not be considered computer professionals.

As an example, one day recently I went to the offices of my local county tax appraisal district for some information. Now, when I was a youngster working for a real estate broker, I had spent quite a bit of time looking up this kind of information so I expected to spend half an hour or so just gathering a few facts. Well, lo and behold, the office I entered had been computerized. The man behind the computer punched a few keys and waited for the answers to our questions to appear on the VDT. It took about 20 seconds and as time wore on (remember, we are talking about 20 seconds here), the man behind the counter became somewhat fidgety. He even made some remark about how slow these machines were. He was learning to exhibit impatience with anything that took a little time.

This reminded me of a RUG meeting I attended in which a person was demonstrating a new productivity tool to a group of 40 or 50 people. Of course, there were the usual questions about how the magic was accomplished, about the quality of the documentation and what kind of support could be expected. Then, came the inevitable question. How much time does it take? The tone of voice of the person asking the question indicated that he would be willing to bet that it was a time-consuming process. The person giving the demonstration agreed, half apologetically, that it was a little slow. It seems the response time was about four seconds. And, of course, all one could do during this entire four seconds was twiddle one's thumbs and bemoan one's lack of productivity. I have heard people talk about a tool using too much machine time. I learned that meant it took a particular tool a couple of hours to massage data and produce what previously would have taken two man-weeks to accomplish. So, the idea of hurry, hurry, hurry takes over even to the point of causing people to be upset with the machines, the computers, even when they are efficient and highly productive.

As I tried to digest all of the information I was getting from professionals in the field, my attention was directed to reports of ergonomic studies being done in the workplace.

In an effort to handle the many complaints about physical illness and discomfort emanating from the computerized workplace, business, industry, social and fraternal organizations, unions and government were fighting to protect the interests of workers.

Workplace furniture manufacturers were producing adjustable, form-fitting chairs and bi-level desks to reduce the number of reported incidences of backache, muscular soreness and fatigue.

VDT manufacturers were studying all aspects of the screen, its surface, angle, tilt, character resolution, etc.

While I am sure there are some altruistic motives behind the efforts that have been put forth, it appears to me that the main effort is to make the human more productive, more efficient and more manageable. It also appears to me that most people forget the nature and needs of human beings.

The basic nature of human beings has long been established and does not admit of change; that is, the human being has sensory powers, intellectual powers, capacity for emotional experience, physiological and physical characteristics. These are part of the human whether he is interacting with machines or tilling the soil by hand or using his brain to solve intricate problems, scientific or social. Although there are exceptions, man is a gregarious animal and is inclined toward social activities.

In the opinion of the author, any attempt to look at man as an element in an equation for getting work done is doomed to failure, unless one looks at the human in his entirety. For example, trying to build a workstation that has all the comforts of home and will yield just the right distance between eye and reading material will not insure that the person will be delighted to sit at a desk for eight hours a day with no time to walk about and stretch and get a break from the work at hand. Soft lights and mellow music may not insure a peaceful mind because within the heart of the individual may beat a drum that insists that he separate himself from the 9-5 work scene and strike out in a different direction.

The thesis here is that if the company recognizes the humanity of the individual and attempts to satisfy the basic needs that their production will be enhanced considerably whether or not the physical situation in which they find themselves is desirable.

Ergonomically speaking, a company may provide the ideal arrangement and still be left wondering why production does not equal prediction.

In other words, the person's mental attitude and emotional condition are going to determine efficiency and productivity. Naturally, providing satisfactory workstations will do a great deal to make the worker feel his interest is being considered by the company and will have a positive effect on his attitude. On the other hand, if the company stops there, they will surely not accomplish their desired goal.

Look at a practical matter that is with us in this day and time. The experts differ to some degree, but most would accept the fact that half of the marriages today in the United States will end in divorce. These divorces are not taking place in a vacuum leaving all persons unaffected. The mental anguish that pervades the parties' entire beings exacts a toll on the person's ability to perform well on the job. There are many other stress-producing experiences that we could mention, but this one will suffice. If a company expects the person to leave his personal life at home, that may be expecting too much. Perhaps it would be a wise investment on the part of the company to make counseling available for employees. If the company takes the attitude that the person had better leave his personal life at home or risk being replaced, it will probably not be bettering its own situation. It may just exchange one set of problems for another. It will have to hire another person who will have his own set of problems and neither the company nor the individual concerned will profit. Individuals still need an emotionally supportive environment in which to work.

It is foolish to spend all of a company's money trying to figure out which dial is easier to read by a human being and then expect him never to make a mistake. Whether or not it makes sense, mistakes will be determined more by the condition of the mind than by the nature of the workplace. A highly motivated person with a sense of self-worth and a conscience that requires him to give an honest day's work for an honest day's pay will perform to the best of his ability providing he has a supportive environment regardless of the working conditions, unless they are simply inhumane. This is not to say that he will ignore a calloused management who are exploiting workers; rather, it is to say that a company would do well to provide those things that will make a person more at peace with himself so he will be better able to concentrate on the job.

In the high tech atmosphere of the computer world, there are some real dangers to the welfare of the individual. The most glaring threat is that of becoming a pawn in the company's operation to the extent that the individual drifts into a one-dimensional existence. He can become swallowed up into the company's operation in subtle ways until he leaves behind all semblance of real contact with his family, friends and social institutions that can help him over the rough spots in life. There are rare individuals who can spend all of their days and most of their nights attending to the work at hand and never miss the contact with spouse, children or a golfing partner. Most, however, will come to a time in their lives when work will not be the be-all end-all of existence. When that time comes, if they have cut themselves off from their support systems, they will fall apart completely and be unable to function efficiently or effectively in the work place. And for a company to encourage this sort of thing is unconscionable and detrimental to its own welfare.

I know of one man who said that people such as he were not subject to burnout, that they encountered vapor lock occasionally, which could be handled by an afternoon on the golf course, after which they would be ready to meet the demands of a gruelling schedule for another six months. I would have been inclined to believe him if he hadn't been nervously puffing on a cigarette after all of the time we were talking. He really did not give the appearance of one who was handling his activities with aplomb.

Human beings need other humans. They thrive on interpersonal relationships and social networks. Women are seeing the need for this type of thing as they search for and occupy positions in business and industry that previously have been held exclusively by males. Humans can survive neither alone nor by instinct.

A human infant cannot survive on instinct alone. It has to be cared for and nurtured in order to survive and has to be taught the ways of humans. Humans, apparently, never outgrow their need for care and instruction. There is always something to be learned on the human front. We have to prepare ourselves for the work force. We have to develop a skill, learn a trade, become professionals; and it takes long years of training to accomplish this. We have to be educated and nurtured. We have to learn how to become good mates for marriage purposes. We have to learn how to parent for the rearing of children. We have to learn how to live as middle-aged people. We have to learn how to cope with old age. We never reach the point where we are totally self-sufficient. We have needs that can only be met by others who are committed to the human family.

The kinds of people experiences I talked about early in this article indicate that the computer age has brought with it new and severe threats to the welfare of individuals.

What is it about computers that is so detrimental to human development? I am speaking here especially of those who consider themselves computer professionals. I am not suggesting that we rid the world of computers. I am simply trying to get things in perspective so we can handle the problems that have arisen.

It seems to me that two characteristics of human beings come into play. One is addiction and the other is obsession.

Human beings are susceptible to addiction. Usually, we think of addiction as being related to drugs; however, there are other addictions, such as psychological. One dictionary definition of addiction is "slavery to a pernicious habit" with pernicious being defined as that which "causes great harm, is destructive, injurious". All of us who have been associated with computers even in a tangential manner, know someone who has neglected his family, friends, even his work in order to pursue his addiction to the computer. To say this is injurious is putting it mildly. Many have pursued their addiction to the point of losing their friends and family completely.

Human beings are capable of obsession, also. One dictionary definition of obsession is "an idea that haunts one and cannot be shaken off. It is usually associated with dread or anxiety".

To many, their computers have become obsessions. They cannot shake off its appeal. Individuals tend to become fascinated with certain objects or developments in society. That is the reason for fads. Human beings are indeed like sheep in some ways. They are certainly inclined to follow what they consider to be a leader.

What is it about computers that cause them to become obsessions or create an addiction? Why are people so fascinated? As a psychologist, I think the principle thing about them is that an individual cannot obtain what is referred to as closure when working with them.

It has been shown that it is not easy for a person to erase from his mind that call for completion until that job is completed. If, for example, a person is working on a jigsaw puzzle, his mind will stay with the project even when he takes a break from the activity. When he goes off and leaves it for awhile, his mind takes the partially completed puzzle with him. He remembers much about the puzzle, how much has been completed, how much remains to be done and where the missing pieces are located. After he has completed the puzzle, the allure is exhausted and the puzzle is easily forgotten. As long as it is incomplete, it stays in his mind; at least, for a much longer time.

The same is true in other human activities. If we neglect to make an apology when we have been delinquent in social obligation, the concern stays with us, influences our thinking, continues to surface in our minds and disturbs us for a lengthy period. If we go ahead and do what we are supposed to do, even if it is painful, our mind is relieved of the worry connected with the unfinished act. This is one of the reasons why most of us have a feeling of insecurity, or impending doom, what is called a free-floating anxiety about life, because we carry around with us so much unfinished business.

In the case of the computer and its appeal, we never obtain closure. If we write a program, it can always be improved. There is something we can add to it. It is not unlike getting a software product ready for market. Once it is "done", it has to be tested. No matter how long you test it or how many tests you subject it to, there will always remain bugs and as soon as you send it to market, you can expect to receive calls from people who have found some bugs that you missed. I am sure some potentially good products never make it to the market place because the producer cannot get closure. He never reaches the place where he thinks the product is finished.

Again, all of us who have been associated with those who are considered computer science professionals have been ignored by our colleagues who are walking about with a dazed look, as though in a trance, thinking about some knotty problem that has them confused at the moment. The real problem is that when that problem is solved, another will take its place. And so, they lose contact with other humans, contact they sorely need.

As I see it, the chief concern of computer professionals should be the emerging of a one-dimensional life. When it gets to the point where you go from one day to another awaiting the latest publications about computers and their use and all you try to do is keep up with the latest advances in operating systems, or learning the differences between the various computers and database management systems and screen handlers and forget to develop your human characteristics in order to satisfy human needs, you are headed for trouble. Fortunately there are some things you and the company you work for can do to lessen the negative impact of your professions on your daily life, as it concerns others.

You, as an individual, are going to have to recognize your social and family needs.

Business is going to have to realize that it can only go so far in modifying tasks to suit the needs of individuals and be willing to put forth efforts on behalf of employees to provide them with supportive environments. Human beings need to be able to accommodate themselves to the task at hand.

The computer industry itself will have to deal with the following issues in order to insure itself of quality professionals who can contribute to the ongoing success of a company.

1. Identify those people who have the intelligence, background educationally, or who have the capacity to benefit from education and have the strength of character to make a positive difference in the company's operation.
2. Provide them with ongoing education in terms of keeping up with advancements in their field. It is impossible for anyone to keep up with all of the reading in various professional publications to keep themselves informed in their chosen profession. They must have it in digest form and then they can specialize in those areas of their own choosing.
3. They must require their employees to take time off from the work place so they don't become buried to the extent that they have no time for any other facet of life.
4. Provide flexibility in worktime so they can produce the most work in the shortest amount of time.
5. Encourage their employees to develop interests other than work. I know this sounds odd to some, but in order to have a balanced life and not burn out, individuals need variety of experience.
6. Bring mates and even children into the corporate experience so they will learn to appreciate what their parents mean to the company for which they work. It will help develop pride in what they do and what it means to loved ones.
7. Make it possible for them to spend time at professional meetings where they can share experiences with others who have the same problems from day to day.
8. Make it a practice to notice contributions when made and demand to know of management personnel who is responsible for advancements.

It is not at all unusual to find out who is at fault in a given situation so someone can be blamed. It is quite a different matter when someone contributes to a well-oiled machine and keeps it running. Never let special achievement go unnoticed and unrewarded.

9. Be realistic about the time that it takes to achieve a particular goal. Do not put your people under pressure from the very beginning of a project, knowing that they cannot possibly complete the work in the allotted time.

Workers are going to have to make themselves do the following:

1. Guard against a one-dimensional existence.
2. Develop a life style that includes time for relaxation and rejuvenation of spirit.
3. Learn to communicate with mates and children on their terms rather than in computer terms.
4. Settle on one area of computer science and try to learn as much as possible about it so as to become an expert in that field. Do not try to keep up with every advance in the computer business.
5. Develop interests that will give you something to think about other than your job during your offtime, or time away from the office.
6. Practice communicating with others about things other than computers.
7. Develop some hobbies that include other human beings.

Biography

Dr. Ernest R. Simmons is a licensed psychologist in Texas for private practice. His background includes positions as a college professor, minister, and school psychologist. He is presently Director of Human Resources and Education for The PROTOS Software Company in Austin, Texas.

Interex
2570 El Camino Real West
4th floor
Mountain View, Ca. 94040
USA

BONDED PRINTED
MATTER
BULK RATE
U.S. POSTAGE
PAID
Mountain View, CA.
Permit No. 382

Printed in The Netherlands