COMPETITIVE TIME SHARING OFFERINGS

MARKETING
ANALYSIS
SERVICES

HONEYWELL PROPRIETARY

DISTRIBUTION

RR DOUGLAS	B42
AA BERGLUND	K28
JM BLAIR	C85
FG BLEDSOE	A9
CR BLOSE	A126
RR BROOKMAN	C97
DW BURBECK	B48
H CART	C17
CT CLINGEN	MA22/CISL
MS DAVIS	A126
RL DAY	B48
JH DENNY	C60
JE DEARS	A79
PM DICK	A10
JW ENGLISH	B77
HB ERLICK	A79
WL ESTFAN	C63
WR GAINES	A79
GC GALLIHER	B48
GA GILLETTE	C73
LE GOERNER	C16
MA GOLDMAN	B10
JR GUILFORD	A126
R HESSER	T60-10
VC HOLLAR	C17
RS HUNT	A79
FC JACKSON	A79
AR JOHNSEY	A126
WS KAR	C17
LL KELLY	A24
S KLEE	CA19/LA
GB KREKELER	A126
AL LONGANECKER	A126
LF LUNETTA	A126
DF MANZER	A79
RJ MCDERMOTT	C79
KE NORLAND	B83
DJ O'CONNOR	B126
GE OLSON	A79
VE PIACENTI	A10
FR PRIEST	B8
AJ PRUNEAU	B38
JC RICHTER	C64
J RING	C17
DC RUPLEY	B48
AJ SEITER	A126
PN STOUGTON	C65
ER VANCE	C58
MD VANTZELFDE	A122
DJ WEST	B41
JR WILKINSON	B10
DG ZIEGLER	C17

O MIRELES	T-99-1
AE FERGUSON	T99-3
R FROMMER	T99-4
J JONES	T99-4
WG KAISER	T99-4
J LAMB	T99-4
RM MAY	T99-4
I MCKINZIE	T99-4
EH MULLEN	T99-4
C CHANCE	T99-5
AR MEHTA	T99-5
JD ABEL	T60-7
M BERRY	T60-8
CJ BREWER	T60-4
CJ BURKHART	T60-8
RW BUSCH	T60-2
DL EDWARDS	T60-2
J FIRNHABER	T60-8
HW GESHWIND	T60-5
R GILBREATH	T60-2
GA GUNN	T60-2
R HESSER	T60-10
R HETRICK	T60-5
RG INGLE	T60-2
HE JOHNSON	T60-5
D LAMBERT	T60-7
D LEO	T60-2
FS MATTHEWS	T60-3
WF MCCAIN	T60-1
JL McCULLOCH	T60-2
J McNULTY	T60-6
JL MEE	T60-7
JL MEYERSON	T60-2
PJ NASH	T60-2
WR ROBERTS	T60-9
E RUSSELL	T60-5
JM SELMAN	T60-2
TE SPACKMAN	T60-8
CB THOMPSON	T60-1
JL UNDIANO	T60-2

R ALVARADO	K28
A BEATTIE	K28
J BROWN	K28
K CARLYLE	K28
G DIXON	K28
A DOWNING	K28
J FALKSEN	K28
J GILDERSLEEVE	K28
JE GRAY	K28
R HOLMSTEDT	K28
A KEFNER	K28
R KRESS	K28
R LACKEY	K28
F MARTINSON	K28
J OHLIN	K28
T VAN VLECK	K28
E WALLMAN	K28
D WARD	K28

Honeywell Interoffice Correspondence

Date: 3 May 1979 copy: J.L. Undiano

To: DISTRIBUTION

From: J. L. Meyerson

Location: Market Analysis Services
Phoenix, T60, 249-7980

Subject: COMPETITIVE TIME SHARING OFFERINGS

Attached is the Competitive Time Sharing Offerings description. It is hoped that this manual will be useful to you as a reference source. The document will be updated as more competitive information becomes available.

Your comments and questions should be directed to Competitive Analysis, mail station T60-2.

James L. Meyerson

James L. Meyerson

/jk

CF 25 (4-78)

COMPETITIVE

TIME SHARING

OFFERINGS

JAMES L. MEYERSON
MARKET ANALYSIS SERVICES
MAY, 1979

HONEYWELL SENSITIVE

THIS DOCUMENT AND ITS CONTENTS ARE HONEYWELL SENSITIVE. IT IS DESIGNED TO INFORM COMPETITIVE ANALYSIS, PRODUCT PLANNING, TSS DEVELOPMENT, MULTICS DEVELOPMENT, CP6 DEVELOPMENT, AND RELATED AREAS WITHIN HONEYWELL OF COMPETITIVE VENDOR TIME SHARING SYSTEMS. UNDER NO CIRCUMSTANCES SHOULD THIS DOCUMENT OR ITS CONTENTS BE SHOWN OR DISCUSSED WITH A CUSTOMER, PROSPECT, OR PERSONS EXTERNAL TO HONEYWELL.

TABLE OF CONTENTS

Burroughs

BURROUGHS CANDE

DESCRIPTION: CANDE (Command and Edit) is a message control system that enables users at remote terminals to enter programs or data files into disk storage, compile and execute programs, edit and alter programs or files, search files, send messages to other terminals, and perform a variety of other functions.

GENERAL APPLICATION: CANDE is used on all Burroughs time sharing.

DEVELOPMENT HISTORY:

CURRENT RELEASE/RELEASE CYCLE: Latest release first appeared in 1976.

STRUCTURE:

- Uses memory overlap (virtual memory)
- Uses "segmented" memory concept. Page size $\approx$ program segment length
- A stack is associated with each job in the system. The stack is a contiguous area of memory assigned to job. Two high-speed top-of-stack processor locations are linked to job's stack memory area. Processor execution uses RPN for data manipulation.

OPERATING SYSTEM: The MCP is an integrated operating system that oversees and controls all operations of the B 6700 and B 7700 systems. It consists of a group of routines organized in three-level hierarchical fashion. The first level is a kernel routine that fields all interrupt signals and transfers control to the appropriate MCP routines. The second-level routines handle the MCP's major task: dynamic resource allocation of main memory, disk storage, I/O devices, processors, and time among the concurrently operating programs. The third-level routines handle utility functions such as job scheduling, control card interpretation, file control, library maintenance, etc.

Virtual memory is through automatic program segmentation. Programs are segmented at compile time and called in as needed by MCP during execution. A complex "variable memory allocation" allows system to commit memory according to job mix: Reverse Polish Stack organization. WORKFLOW management is used for job queuing once jobs are entered from CANDE. (100 levels of priority.)

LANGUAGES SUPPORTED: ALGOL, XALGOL, DCALGOL, FORTRAN, XFORTRAN, COBOL, BASIC, PL/1, ESPOL, (the intermediate level language MCP is written in).

OTHER FUNCTIONALITY: Reentrant code, unmodified by execution allows a single copy to be shared by several jobs in main memory.

NUMBER OF CANDE INSTALLATIONS:

FILE TYPES/FILE NAMING: Consists of a list of 1 to 12 identifiers separated by slashes. (Each identifier may be 1 to 17 characters in length.) EBCDIC is the primary internal code, although ASCII is available. Words are 48 bits long for data with an additional 3 control bits and 1 parity bit (total of 52 bits)

CANDE COMMANDS:

BIND: invoke the standard system binder
COMPILE: invoke compiler(disignated in filename)
EXECUTE: execute object program
RUN: same as EXECUTE except RUN will provide for compilation if necessary
GET: recalls an existing file as the workfile
MAKE: creates a new workfile
REMOVE: removes the workfile or any other file from user's library
SAVE: save current workfile in user's library
TITLE: change name of file or workfile
BYE: terminates current terminal session
HELLO: logout without disconnecting terminal
RESEQ: resequence line numbers in workfile
TAPE: read input from paper tape
SEQ: invokes automated line sequencing

CANDE EDITING COMMANDS:

DELETE: discard lines from workfile
FIX: insert new text or replace parts of line
INSERT: copy lines from workfile or other file into workfile at given sequence location; resequence line numbers afterward
MERGE/RMERGE: merge file or portions of file into workfile
MOVE: move lines from one point to another; change sequence numbers
FIND: locate first occurrence of specific string
REPLACE: insert new text for every occurrence of old text in workfile.

- DATAMATION, DECEMBER 1978 -

RESULTS

CANDE

Users Reporting:	6
Overall Satisfaction:	3.2
Throughput/Efficiency:	3.0
Ease of Installation:	3.2
Ease of Use:	3.2
Documentation:	2.8
Vendor Tech. Support:	2.8
Training:	2.5

Advantages: flexible, inexpensive
Modifications usually required: none
Price: $1,800 plus $180/year license fee

References:

Master Control Program Burroughs # 1032927	May 1975 CL # B7700 2927
MCP Reentrant Programs Burroughs # 1031770	May 1975 CL # B7700 0 1770B
MCP Virtual Memory Burroughs # 1032901	May 1975 CL # B7700 0 2901B
MCP Dynamic Resource Allocation Burroughs # 1031796	May 1975 CL # B7700 0 1796B
MCP Workflow Management Burroughs # 1084985	May 1975 CL # B7700 0 4985B
Auerback, Datapro	

P&PM-MAS
03-14-79
Rev. 0


CONTROL DATA
CORPORATION

CDC KRONOS

Description:

KRONOS emphasizes interactive processing from remote terminals along with intermixed local and remote batch processing.

General Application:

High volume scientific, educational, and governmental applications where there is a heavy dependence on time sharing.

Current Release/Release Cycle:

KRONOS 2.1 released during early 1977 was the final version. It was not offered on the Cyber 170 systems and is no longer supported on Cyber 70 and 6000 Series computers.

Operating System:

The basic module includes facilities for local and remote batch processing, the COMPASS assembly language, and the Cyber Record Manager, a general-purpose input/output package that supports the file structures.

Structure:

Various components of the KRONOS software reside in main memory, Extended Core Storage, Peripheral Processor (PPU) storage, and the system disk unit. KRONOS (like NOS) uses the Peripheral Processors to handle system and I/O functions. One PPU contains the Monitor routine and permanently controls the system's overall operations. A second PPU is permanently assigned to control the console keyboard and displays. In the event of a malfunction, the Monitor and operator display functions can be reassigned to alternate PPU's. The remaining PPU's are used to perform I/O and system tasks on a dynamic pool basis. If the Time-Sharing Subsystem or the Remote Batch Subsystem is active, a PPU is dedicated to each subsystem.

File System:

The physical record unit (PRU) is the smallest division of data on a device. It consists of 640 six-bit characters. A logical record contains one or more PRUs and is independent of physical environment.

Languages Supported:

FORTRAN, COBOL, COMPASS (assembly languages)

Number Of Installations:

File Types/File Naming:

File names can be 1 to 7 characters in length; ASCII code set used.

Terminals:

ASCII code compatible (such as Teledyne models 33, 35, 37, or 38, CDC 718) or correspondence code (such as Datel 30, IBM 2741) terminals.

Time Sharing Job Commands:

APL	select APL interactive interpreter
BYE/GOODBYE	log off system
CLEAR	release all working files
EDIT	select KRONOS Text Editor
HELP	time-sharing command(s) descriptions
LIBRARY	access programs in special system user number LIBRARY
LIST	print contents of file
LOGIN/HELLO	log onto KRONOS system
NEW	create a new working file
OLD	access a file that was previously saved in permanent file system
RESEQ	renumber line numbers of a file
RUN	compile and/or begin execution of program
STOP	terminate currently executing program

CDC NOS

Description: NOS (Network Operating System) is the operating system for CDC's Cyber 170 series computers. Its multimode job processing allows concurrent processing of local and remote batch and remote interactive (time-sharing) jobs.

All CDC hardware and software elements, including the operating system are separately priced.

GENERAL APPLICATION: high volume scientific, educational and engineering applications.

DEVELOPMENT HISTORY: NOS is a repackaging of the operating systems (SCOPE and KRONOS) available on the 6000 and Cyber 170 series. In July, 1975, CDC announced an additional version of NOS. The newer version NOS/BE (Network Operating System/Batch Environment) represents a compatible upgrade to the Cyber 170 processing for SCOPE users, while the original NOS is supplied as an upgrade for KRONOS users. Release 2 of NOS features a Network Supervisor that allows Cyber 170 computers to act as "home resource" in a computer network addressing up to 256 node points.

CURRENT RELEASE/RELEASE CYCLE:

OPERATING SYSTEM: The basic module of NOS 1 includes facilities for local and remote batch processing, the COMPASS assembly language, and the Cyber Record Manager, a general-purpose input/output package that supports the sequential, direct, indexed sequential, actual key, and word-addressable (relative) file organizations. A recent enhancement to the Record Manager, the Multiple Index Processor, permits indexed sequential, direct and actual key files to be accessed by an unlimited number of alternate keys. Access to the facilities of the batch subsystem for time-sharing users is provided by the Time-Sharing Subsystem. The Tranex transaction subsystem provides a general-purpose transaction processing capability, including centralized control for transaction terminals, a specialized data management system, support for multiple task libraries, journaling, and debugging and testing aids.

NOS/BE provides a compatible upgrade for users of the SCOPE operating system. The basic NOS/BE module includes facilities for batch processing, the Cyber Record Manager, The COMPASS-3 assembly language, and the Form general-purpose file utility. Remote batch processing and time-sharing facilities can be added to the operating system through the Intercom subsystem.

STRUCTURE: Like its predecessors, SCOPE and KRONOS, NOS uses the Peripheral Processors to handle system and I/O functions. Various components of the NOS software reside in main memory,

Extended Core Storage, Peripheral Processor (PPU) storage, and the system disk unit. One PPU contains the Monitor routine and permanently controls the system's overall operations. A second PPU is permanently assigned to control the console keyboard and displays. In the event of a malfunction, the Monitor and operator display functions can be reassigned to alternate PPU's. The remaining PPU's are used to perform I/O and system tasks on a dynamic pool basis. If the Time-Sharing Subsystem or the Remote Batch Subsystem is active, a PPU is dedicated to each subsystem.

Extended Core Storage is supported under NOS primarily as a high-speed swapping device and as a storage medium for user data files. NOS/BE also supports Extended Core Storage for buffering and large data arrays.

Both NOS 1 and NOS/BE can include a limited shared-processor facility with the Multi-Mainfram Module, which ultimately will permit up to four computers to access permanent files (but not shared input/output queues) stored on mass storage devices.

LANGUAGES SUPPORTED: ALGOL-60, BASIC, COBOL, FORTRAN, COMPASS (assembly language), interactive BASIC and FORTRAN, and APL (APL is available under NOS, but not NOS/BE).

NUMBER OF INSTALLATIONS:

DATA STRUCTURE: The basic CYBER 170 series data unit is a 60-bit word plus up to 8 parity bits. The instruction set contains only 64 instructions that are available to (non-system) programmers.

6-bit BCD is standard "display code".

TIME SHARING JOB COMMANDS

APL	select the APL interpreter.
BYE or GOODBYE	log user off system and disconnect terminal.
DAYFILE	look at results of previously executed time-sharing job.
EDIT	select the Text Editor.
ENQUIRE or STATUS	requests current job status.
HELP	gives online assistance in the use of system commands.
I	interrupt current output to terminal.
OLD	allows the user to access a previously saved file.
PASSWORD	allows validated users to change their password.
RESEQ	resequences line numbers of the primary file.
RUN	compiles and/or begins execution of a primary file.
S	terminate a job that is currently sending output to terminal.

SUBMIT	create a batch job deck image and submit it as a deferred batch job.
TEXT	enter text mode (editor that does not require line numbers).

INTERCOM: a set of communications control routines that provide users at remote terminals with both interactive and batch-mode access to a Cyber computer system. The Intercom routines establish a suitable interface between the remote users and the NOS/BE operating system. A Text Editor permits interactive creation and editing of programs and data files. Terminal users can also enter programs into the batch queue and receive the resulting output.

INTERACTIVE GRAPHICS SYSTEM (IGS): operates under Intercom and permits interactive problem-solving by users at graphics consoles such as the CDC 274, 241, and 777.

TERMINALS: Two types of terminals may be used:

- ASCII code compatible terminals such as Teletype mdoels 33, 35, 37, and 38.
- Correspondence code terminals (such as Datel 30 and IBM 2741). For correspondence code terminals, modification may be required if transmission at greater than 10 Chars./second is required.

REFERENCES:
- Auerbach Reports, volume 601, section 100.2952.702.
- Datapro, volume 1, pages 70C-263-01a to 70C-263-12r.
- CDC NOS Cyber 170 NOS 1.0 Time Sharing Reference Manual, CL #C0170 0 4355.

Dartmouth DTSS

DARTMOUTH DTSS

Description:

DTSS is widely known for its time-sharing which is its inherent mode of operation. Since its inception in 1964, DTSS software has operated on General Electric (and now Honeywell) computer equipment. DTSS software is owned by DTSS Incorporated (a subsidary of the Metropolitan Life Insurance Company) and is available on a license fee basis.

General Application:

Academic institutions (small percentage of total sales); FORTUNE 500 companies for in-house TSS development; and computer service bureaus.

Current Release/Release Cycle:

Release 4 was released on March 1, 1979.

The previous release (release 2) came out August 1, 1978. DTSS tries to stay on a six month release cycle.

Structure:

See "Job Tree" diagram.

Operating System:

DTSS can handle 30 to 220 simultaneous users. Batch is run as a background facility (to time sharing); a job which runs on-line may be run in batch without program modification.

DTSS Program Descriptions:

A collection of programs forms the basic software for the DTSS operating system. Some of these include:

- EXEC controls physical I/O, scheduling and security aspects of the system.
- LOGIN controls user access to DTSS and accounts for usage.

DTSS Program Descriptions..continued:

- LDUMP and LLOAD are the logical dump and load programs that transfer the entire DTSS file system between disk and tape, primarily for backup purposes.

- SIMON is the monitor users converse with at their terminals.

- BAKMON is the background monitor that controls the processing of background jobs and jobs that use peripherals such as magnetic tape drives and line printers.

- CATALOG provides information such as names and lengths of saved files.

- EDIT provides context - independent editing functions such as rearranging the order of lines in a file.

- TEXTEDIT is an easy-to-use, line-oriented context editor.

- STREDIT is a sophisticated, character-oriented editing language processor capable of searching for strings that match regular string expressions.

- BILLING manages system usage accounting and produces reports.

- SORT is an efficient sorting package that can be accessed conversationally or called from user programs.

Languages Supported:

BASIC, FORTRAN-76, COBOL-74, DTSS APL, ALGOL 60, LISP 1.5, MIX, XPL, DYNAMO, GMAP. All compilers operate in the normal user's workspace. The code occupies about 8K words. Compiles for FORTRAN, COBOL, and BASIC are single-pass, compiling directly into core.

File System:

Files are automatically allocated as needed before a program is run (it's not necessary to preallocate storage). One level of file protection can guard against accidental file destruction, another level can withstand casual attempts to access a file, and still another can resist determined efforts to access a file.

Data Base Capabilities:

Historically, DTSS has been weak in this area. DTSS now provides three data base systems:

- FIND is a simple interactive information retrieval tool.

- DDBMS is a CODASYL DBTG - consistent data base. It utilizes a data definition language and a data manipulation language. DDBMS separates the data base file structure from application programs. DDBMS programs must be written in FORTRAN or BASIC.

- DaTaSyS is a relationsional data base that was developed after FIND and DDBMS. It runs as a separate job within the DTSS operating system and communicates with application programs via the file system. An interactive facility, guide, allows the creation and manipulation of DaTaSyS data bases by non-programmers.

Other Capabilities:

The DTSS sort/merge facility is a powerful and convenient programming tool which may be run as a separate job or called as a subprogram from a running program. Input files are sorted into a single output file according to user-specified merge procedures and sort fields. Files may be sorted as multi-line blocks in either ascending or descending order. Several sorting fields may be specified. The user may specify one of five options for resolving collisions. Tape and/or disk files may be sorted in the same job.

Editors:

DTSS provides four different on-line editors to meet different requirements. EDIT is used for merging, listing, and reordering line-numbered files. TEXT and STRING both offer sophisticated character-by-character editing. STRING operates in either conversational or file execution mode. QED (Quick Editor) is a fast, easy-to-use context editor that features an efficient and highly abbreviated syntax.

Major features of QED are:

- Forward and backward searching.
- Line orientation.
- Addressing by context.
- Insertion of new text.
- Deletion of old text.
- Substitution of characters on a particular line or set of lines.
- Moving of lines to temporary files.
- Combining, splitting, or rearranging files.
- Macro commands sequences for repeated execution.

Terminal Support:

DTSS supports all common 10, 15, and 30 character per second terminals, including Teletype, DecWriter, Terminet-300, IBM 2741, and similar models. Faster devices, such as the 1200 bit per second Terminet-1200 and Tektronix graphics terminals, are also supported as are synchronous lines. DTSS automatically sets the proper speed and conversion mode for the modem and terminal being used.

DTSS can utilize a variety of types of communications networks. Special software features help support inexpensive multi-drop, frequency-division-multiplex networks. The software also provides an inexpensive connection to Telenet. The DTSS Datalink modules provide direct connection to IBM 360/370 and similar systems for file and job transfer.

References:

DTSS - The Dartmouth Time Sharing System; 1976 DTSS Sales Notes

DTSS "job tree"

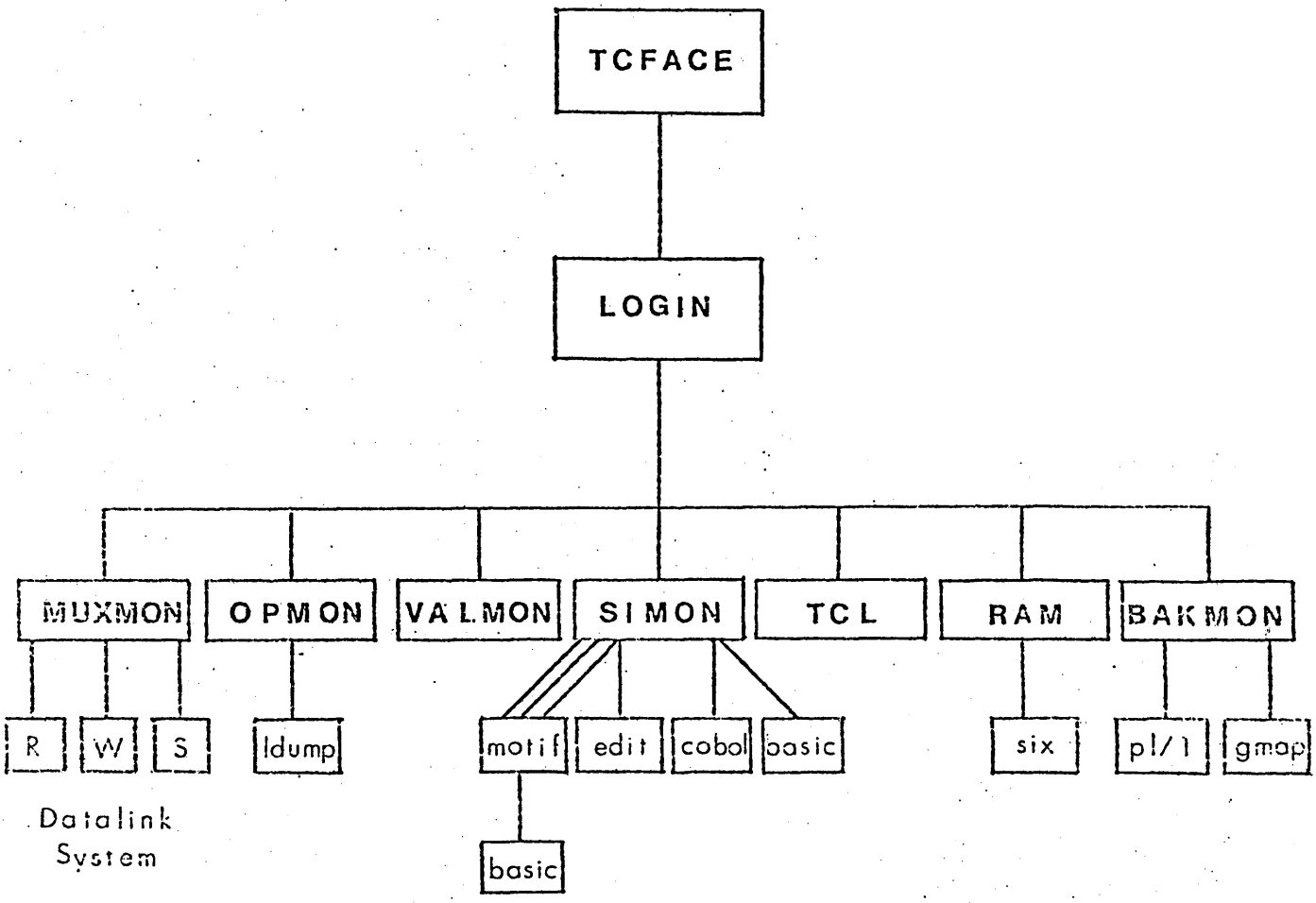


Figure 3.4

DARTMOUTH TIME-SHARING SYSTEM

Key DTSS Modules

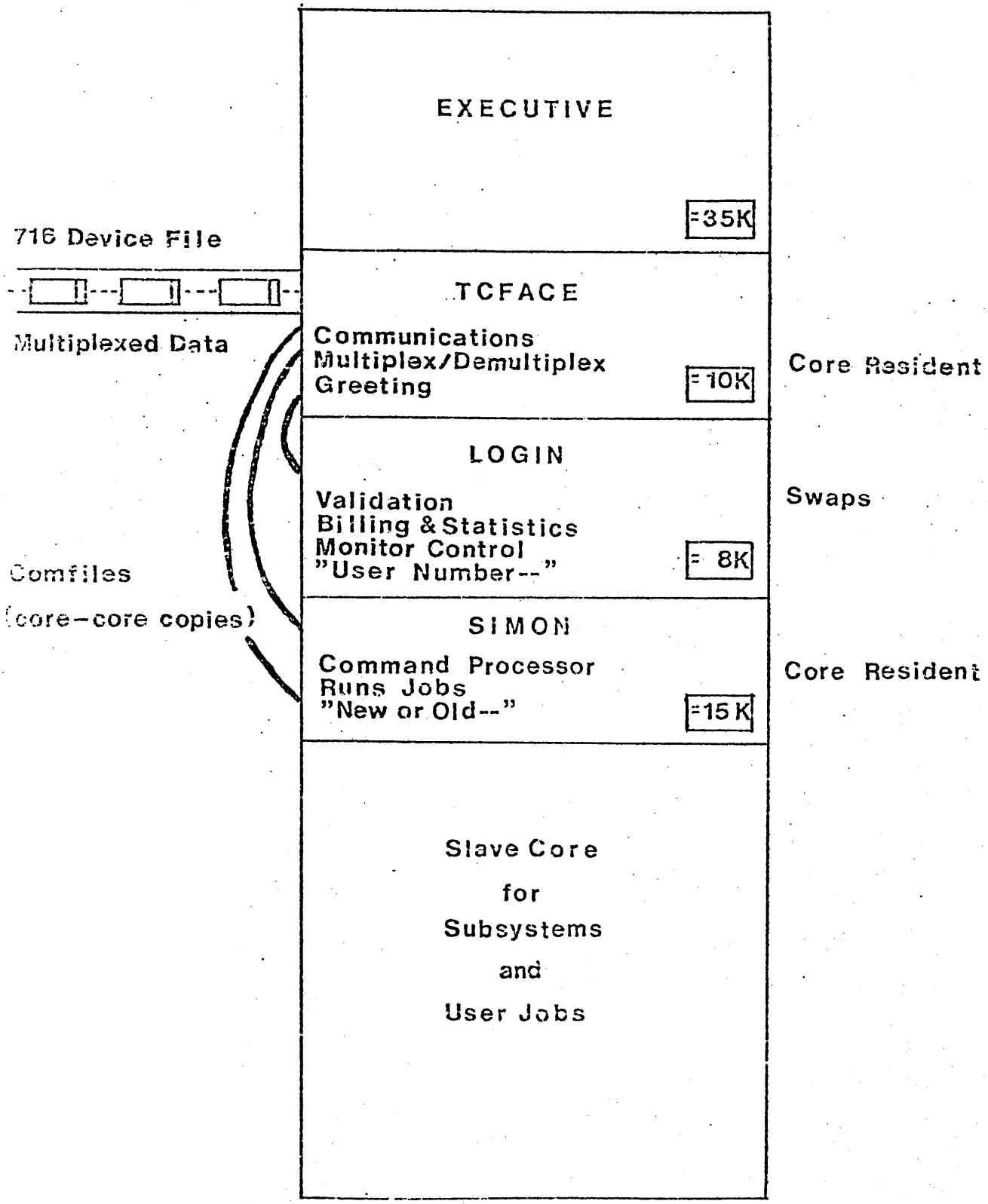


Figure 3.3


digital

DEC TOPS-10

Description: The operating environment of the DEC System-10 is controlled by the TOPS-10 operating system. TOPS-10 controls an integrated environment of time sharing, remote batch processing, multiprogrammed batch, real-time, and multiprocessor configurations. Unlike many mainframe offerings, the DEC System-10 (and hence, TOPS-10) was designed to be primarily a time-sharing system.

General Application: Small- to medium-sized, multiprocess operation. At present, TOPS-10 has very limited business application packages, but does well in educational, scientific, and engineering applications.

Development History: Work on TOPS-10 began in the late 1960s and it was formally introduced in 1970.

Current Release: TOPS-10 revision 6.03A was released in September, 1978.

Structure:

Operating System: Main residence requirements of the operating system vary from 30K to 50K words. One aspect of the operating system, whose value is extremely application-dependent, is that all jobs process according to a predefined time slice (either ½ or 1 second). On the one hand, for a time sharing or similar environment where maximized use of system resources is desired, this is an equitable arrangement. On the other hand, it makes the continuous processing of batch "high-priority" jobs, such as those in commercial shops, almost impossible. This situation is further complicated by the fact that the operating system views the entire batch job stream as one time shared user job.

Virtual memory is an option in the DEC System-10, allowing users to implement the virtual-memory facility where it is best effective. An installation manager can choose the jobs to be run in a demand-paged environment and can even suggest which type of memory-allocation technique (shuffle, checkerboard, or best-fit) is to be used. Digital's operating system is not a full-blown virtual operating system such as the IBM VS.

Languages Supported: COBOL, FORTRAN IV, ALGOL-60, BASIC, APL, MACRO-10 Macro Assembler, CPL (a subset of PL/1).

Number of Installations: As of July 7, 1978, there were 90 TOPS-10/ TOPS-20 users.

File Types/File Naming: ASCII files. There are five parts to a file description: 1) device name, 2) directory name, 3) file name, 4) extension ("subclass"), and 5) version number (enumerates successive versions of file).

example: ① ② ③ ④ ⑤

PS : <JLM> SCAN. FORT ; 2

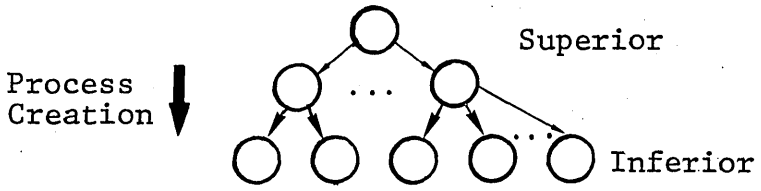
print the second version of the FORTRAN file SCAN that can be found in directory JLM.

Words are 36-bit (9 bytes/bit) with one parity bit appended. The 36 bits can be addressed at the word, byte, or bit level. Byte instructions pack or unpack bytes of arbitrary length anywhere within a word.

File System: The file system is the general I/O mechanism. It performs three subfunctions:

- symbolic file name translation
- file access checking
- actual file operations.

Job/Process Structure: A "job" is just a set of hierarchically related processes. The process is a tree structure:



Superior processes can control (kill, freeze, or resume) inferior processes.

Process communication is via shared memory and a pseudo-interrupt system (called "signals"). TOPS-10 uses a working directory approach similar to UNIX and Multics.

Text Editor: A text editor, TECO (Text Editor and Corrector) was initially implemented at MIT using a graphics display. TECO is character-string oriented and includes the ability to define programs to do general string substitution.

TECO Editor Commands: TECO is a character-oriented editor. It does not require line numbers. TECO requires 5K of core memory, 3K of which is shared in a reentrant system.

A: (append) read in next page from input file without clearing contents of editing buffer.

D: delete character(s) at current character pointer position.

E: end TECO editor session.

I: insert characters immediately following current character position.

J: move character pointer to beginning of the buffer.

L: advance buffer pointer on a line-by-line basis.

P: print contents of editing buffer.

S: search for character string in editing buffer.

Y: "yank" clears the editing buffer and reads next page of the input file into the buffer.

ASCII/BCD: TOPS-10 uses 36-bit ASCII words.

DEC Terminals Supported:

6T40 - Graphic display; 12-inch screen with 2,432 char/ display; PDP-11 control unit; 300-9,600 baud; standalone capabilities; light pen.

VB10C - Graphic display; 21-inch screen; light pen.

VT05 - A/N display; 1,440 characters; up to 2,400 baud.

VT50 - A/N display; same as VT50 except with 1,920 char upper- and lowercase

LA36 - Keyboard printer; 30-cps; 128-character set

References:

- Auerbach reports.
- C.G. Bell et al, "The Evolution of the DEC System-10" Communications of ACM, January 1978, pages 52-62.

DEC TOPS-20

Description: TOPS-20 is a multimode, virtual memory time sharing operating system that can handle up to 64 local or remote terminals. If the GALAXY processor is included, it also handles multistream batch processing. The command language for batch and time sharing modes is the same so that a user can create a program in time sharing mode and run it in batch. All DEC System-20 models are controlled by the TOPS-20 time sharing operating system.

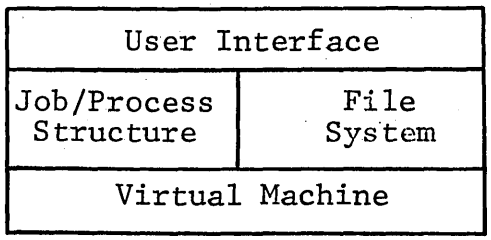
General Application: Digital's traditional markets have been in educational, scientific, commercial, and engineering applications requiring a mix of time sharing, batch, and TP on a single system. TOPS-20 is particularly aimed at multiprocess operation.

Development History: TOPS-20 represents several years of work in merging the virtual storage, time sharing TENEX system developed by the Defense Department's ARPA network by Bolt, Beranek, and Newman (BBN) with DEC's TOPS-10 time-sharing operating system. TOPS-10 was introduced in 1970, the BBN Pager extension was written in 1971, and TOPS-20 was announced (concurrently with the DEC System 20 announcement) in 1976.

Current Release/Release Cycle: TOPS-20 Release 3A came out in October 1978.

Structure: The virtual machine environment allows each user program to be treated as a separate process with its own 256K-word addressing space; the operating system divides programs up into 512-word pages that are swapped back and forth from disk.

Operating System: The operating system architecture is:



Operating System: continued...

The virtual machine separates monitor calls and implements all functions except virtual memory.

Unlike many competitive operating systems, batch is considered subordinate to time sharing in a multimode environment.

Languages Supported: FORTRAN, COBOL, ALGOL, APL, CPL (subset of PL/1), and a Macro Assembler. All DEC System-20 software is re-entrant.

Number of Installations: As of July 7, 1978, there were 90 TOPS-10/TOPS-20 users.

File Types/File Naming: ASCII files. There are five parts to a file description: 1) device name, 2) directory name, 3) file name, 4) extension ("subclass"), and 5) version number (enumerates successive versions of file).

example: ① ② ③ ④ ⑤

PS : <JLM> SCAN. FORT ; 2

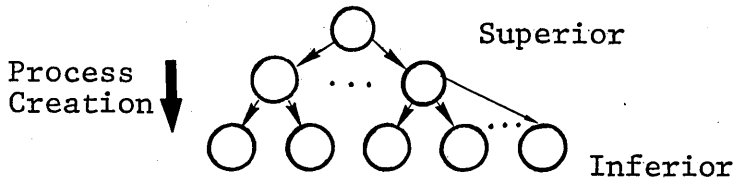
print the second version of the FORTRAN file SCAN that can be found in directory JLM.

Words are 36-bit (9 bytes/bit) with one parity bit appended. The 36 bits can be addressed at the word, byte, or bit level. Byte instructions pack or unpack bytes of arbitrary length anywhere within a word.

File System: The file system is the general I/O mechanism. It performs three subfunctions:

- symbolic file name translation
- file access checking
- actual file operations.

Job/Process Structure: A "job" is just a set of hierarchically related processes. The process is a tree structure:



Superior processes can control (kill, freeze, or resume) inferior processes.

Job/Process Structure: continued...

Process communication is via shared memory and a pseudo-interrupt system (called "signals"). TOPS-20 uses a working directory approach similar to UNIX and Multics.

Text Editor: A text editor, TECO (Text Editor and Corrector) was initially implemented at MIT using a graphics display. TECO is character-string oriented and includes the ability to define programs to do general string substitution.

TECO Editor Commands: TECO is a character-oriented editor. It does not require line numbers. TECO requires 5K of core memory, 3K of which is shared in a reentrant system.

A: (append) read in next page from input file without clearing contents of editing buffer.

D: delete character(s) at current character pointer position.

E: end TECO editor session.

I: insert characters immediately following current character position.

J: move character pointer to beginning of the buffer.

L: advance buffer pointer on a line-by-line basis.

P: print contents of editing buffer.

S: search for character string in editing buffer.

Y: "yank" clears the editing buffer and reads next page of the input file into the buffer.

Reference:

- Palyn Time Sharing System Seminar notes, January 1979
- TOPS/20 User's Guide
- C.G. Bell et al, "The Evaluation of the DEC System 10" Communications of ACM, January 1978, pages 44-62.
- Auerbach Reports.

TOPS-20 COMMANDS

SYSTEM ACCESS COMMANDS

These commands allow you to gain and relinquish access to the system, to change your job's account number, and to release and connect terminals to your job.

ATTACH/DETACH	Connects/disconnects your terminal to a designated job.
ENABLE	Permits privileged users to access and change confidential system information.
LOGIN	Gains access to the TOPS-20 system.
LOGOUT	Relinquishes access to the TOPS-20 system.

FILE SYSTEM COMMANDS

The file system commands allow you to create and delete files, to specify where they are to be stored, to copy them, and to output them on any device.

ACCESS	Grants ownership and group rights to a specified directory.
CONNECT	Removes you from your current directory and connects you to a specified directory.
COPY	Duplicates a source file in a destination file.
CREATE	Starts EDIT for making a new file.
DELETE	Marks the specified file(s) for eventual deletion (disk files only) or deletes the specified files (all other devices).
DIRECTORY	Lists the names of files residing in the specified directory and information relating to those files.
EDIT	Starts EDIT for changing an existing file.
LIST	Prints one or more files on the line printer with or without formatting.
PRINT	Lists one or more files on the line printer.
RENAME	Changes one or more descriptors of an existing file specification.
TYPE	Types the specified files on your terminal.
TMOUNT	Requests that a magnetic tape be made available to the user.

DEVICE HANDLING COMMANDS

These commands allow you to reserve a device prior to using it, to manipulate the device, and to release it once it is no longer needed.

ASSIGN	Reserves a device for use by your job.
BACKSPACE	Moves a magnetic tape drive back any number of records or files.
EOF	Writes an end-of-file mark on a magnetic tape.
REWIND	Positions a magnetic tape backward to its load point.
SKIP	Advances a magnetic tape one or more records or files.

PROGRAM CONTROL COMMANDS

The following commands help you create, run, edit, and debug your own programs.

COMPILE	Translates a source program using the appropriate compiler.
CREF	Runs the CREF program which produces a cross-reference listing and automatically sends it to the line printer.
CSAVE	Saves the program currently in memory so that it may be used by giving a RUN command. The program is saved in a compressed format.
DDT	Merges the debugging program, DDT, with the current program and then starts DDT.
DEBUG	Takes a source program, compiles it, loads it with DDT and starts DDT.
EXECUTE	Translates, loads, and begins execution of a program.
FORK	Makes the TOPS-20 language work for a particular address space.
GET	Loads an executable program from the specified file.
MERGE	Loads an executable program into memory and merges it with the current contents of memory.
POP	Stops a copy of the TOPS-20 Command Language and returns control to the previous copy of the Command Language.
PUSH	Starts a new copy of the TOPS-20 Command Language.
R	Runs a system program.

PROGRAM CONTROL COMMANDS continued...

REENTER	Starts the program currently in memory at an alternate entry point specified by the program.
RUN	Loads an executable program from a file and starts it at the location specified in the program.
SAVE	Copies the contents of memory into a file in executable format. If memory contains a program, you may now execute the program by giving the RUN command with the proper file specification.
START	Begins execution of a program at the location specified in the entry vector.

EDIT COMMANDS

A: Alter contents of specified line(s).

B: Save current file but do not end current editing session.

C: Copy specified line(s).

D: Delete specified line(s).

E: End current editing session.

F: Find next occurrence (after current line) of specified string.

H: Print a list of EDIT commands and options.

I: Input line(s) starting at specified location.

L: List specified line(s).

N: Renumber specified range of lines.

P: Print specified range of lines.

S: Substitute new string of characters for all occurrences of the existing string.

?: Print a list of all the EDIT commands.

Honeywell

GCOS TSS

DESCRIPTION:

Running under GCOS operating system, TSS consists of a time sharing executive, a number of independent processing subsystems under the exec and a common control language.

GENERAL APPLICATION:

Multimode (batch, timesharing, and transaction processing) computer users.

DEVELOPMENT HISTORY:

GCOS III was first introduced in 1965-66. In 1971, TPE was introduced and implemented. TDS was developed by CII-HB (1973) and placed under GCOS in 1974. In 1971 the EIS instruction set extension was made.

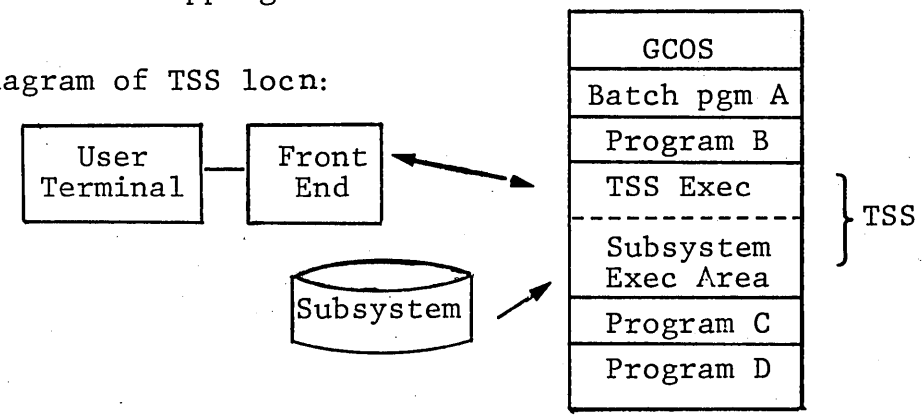
CURRENT RELEASE - RELEASE CYCLE: 4/J

The release cycle is approximately annual.

OPERATING SYSTEM:

TSS as a master mode GCOS program w/fixed dimensions (real memory with memory limitations). GCOS can concurrently run TSS, Batch, TP, etc. as distinct nonoverlapping subsets.

Diagram of TSS locn:



STRUCTURE:

TSS is a master mode program residing on top of a series of subsystem programs. TSS allocates subsystem space. The core-resident Time Sharing Executive is structured as a slave program to GCOS. It performs the functions of selecting, allocating, dispatching, and swapping time sharing user programs. In addition, it suballocates memory and subdispatches the processor to the individual time sharing programs. The executive resets the base register to protect the other programs in memory.

All terminal input/output security arrangements are controlled by the executive. It also accounts for the resources used by the individual time sharing users. The same priority scheme for memory swapout applies.

THE SUBSYSTEM INCLUDES:

ABACUS: Simple algebraic "calculator"

ACCESS: File management system interface. Through ACCESS, a user can create, list, modify, delete, and add password restrictions on catalogs & files.

BPRINT/BPUNCH: TSS file contents can be diverted to central system printer or punch.

DATABASIC: Simple way to create, use, and maintain a data base. No record descriptions are required. DATABASIC Uses a BASIC-like language to interface with IDS/I.

FDUMP: Inspection & maintenance facility for random or sequential files (not necessarily TSS files).

HELP: User information files.

CARDIN: Method of submitting a TSS program as a batch job.

JOUT: Means of inspecting batch job output at a terminal.

TEXT EDITOR (EDIT): Facility for building and editing TSS pgms.

RUNOFF: Text file formatting.

LANGUAGES OFFERED:

BASIC, FORTRAN, COBOL-74, APL, ALGOL, JOVIAL, PASCAL, LISP/66, GMAP.

NUMBER OF INSTALLATIONS USING TSS:

File Types/File Naming: File names can be up to 8 characters in length. File types can be the following:

ASCII (linked file for variable length ASCII records);
BCD (linked file of variable length records);
Random (random file of fixed length records).

VOCABULARY:

TSS Command Language

ABC: Call Abacus - evaluate given expression.

ACCESS: Adjust access rights on file(s).

APRINT/BPRINT: Print ASCII/BCD file on system printer.

ASCBCD: Convert ASCII file to BCD

AUTO/AUTOX: Begin automatic line number generation.

BSDASC: Convert BCD file to ASCII

BPUNCH: Convert file to BCD and punch out on system punch.

BYE: logoff TSS system

CATLOG: List catalog and/or file information.

GET: Access specific file and make available for TSS use.

LINELENGTH: Set terminal linelength.

NEWUSER: Logout current user - do not disconnect line

PURGE: Delete specified permanent file(s) and release/overwrite file space.

RELEASE: Remove specified permanent file(s) from file system. (Does not overwrite file space)

RUNOFF: Text formatter.

SCAN: Read specified batch output file.

STATUS: Print user's current usage statistics.

SYSTEM: Exit from current subsystem; go to named subsystem or return to subsystem selection level.

Edit Commands

BACKUP: Move search pointer up a specified number of lines.

BUILD: Append text to file (enter input mode).

CUT/PASTE: Move specified text from one place in file to another.

DELETE: Delete from file indicated lines.

DONE: Exit from EDIT.

INSERT: Go into input mode; line input begins following current line.

PRINT: Print specified line(s) of file

SAVE/RESAVE: Copy contents of current file to permanent file. SAVE is used when perm. file is created. RESAVE used once a SAVE has been made.

REPLACE: Delete specified text; enter input mode.

RESEQ/SEQ: Resequence/begin sequencing line numbers.

RUN: Execute selected program or system.

VERIFY: Print line if specified string is found.

ASCII/BCD: Will handle and convert to/from either BCD or ASCII.

Interfaces:

TSS & File Management System:

ACCESS allows TSS user to create, list, modify, delete, and add password restrictions to catalogs & files.

TSS & Batch:

TSS created and compiled programs can be run as batch jobs through the CARDIN facility. The associated JOUT command-allows a user to monitor the batch job and direct the program results back to his/her terminal.

GCOS SOFTWARE SIZE ESTIMATOR

Software Module	Resident Required (36-Bit Words)
Basic GCOS Monitor	32K + 0.5 to 1.0K/User
Time-sharing Supervisor (TSS)	37K (minimum)
Transaction Processing Supervisor	14K
Transients	1-14K
File Management Supervisor Runtime Module Utilities: File Create/Delete/ Modify/Catalog	5K/Program 32K 4K/Program
Network Processing Supervisor (NPS) (DATANET 6600 Interface)	5K
IDS/I COBOL Preprocessor Runtime Processor	 32K 4-12K/Program
Data Query (under TSS)	4-7K
Management Data Query (MDQ) Under TSS Under Batch GCOS (for Creation of information used on-line)	 4-5K 16-32K 13K (minimum)
Transaction Processor (entry)	13K (minimum)

P&PM-MAS
2 MARCH 79
REV. 0

MULTICS

DESCRIPTION:

Multics (the Multiplexed Information and Computing Services) is a time-sharing-oriented operating system. Originally built as a general purpose computer utility, it has been expanded to handle large-scale users who desire an environment rich in program development, word processing, and data base management tools. Major editors include edm (most simple, easiest to learn), and qedx (an extensive editor and text language).

CURRENT RELEASE:

MR 7.0. The release cycle is approximately annual with a new release introduced each October.

INTERFACES:

A Multics user's "logs into" the time sharing environment. Time sharing (and batch) users communicate with the system through a command processor, which interprets user's commands and eliminates the need for users to deal with a job control language. Multics displays full interlanguage transparency. (It is quite common to call a PL/1 program from a FORTRAN program, then call a BASIC program, etc.) MRDS databases can be accessed from command level via subroutine calls from all programming languages supported by Multics. GCOS can be encapsulated and run under Multics.

DEVELOPMENT HISTORY:

1965 - research began between MIT, GE, and Bell Labs
1973 - introduced commercially
1978 - became part of "standard" product base.

MULTICS STRUCTURE:

1) Built around a segmented and paged virtual memory
2) Selective, controlled sharing of all data and programs via three dimensional access mechanism.
3) File Access Sharing/Restrictions are hardware enforced.
4) World's most secure commercially available system (according to Mitre Corporation study).
5) Remote Terminals Normal Mode of Entry - Time Sharing.
6) Flexible User Interface.

JLM
P&PM-MAS
04-26-79
Rev. 1

OPERATING SYSTEM:

1) 95% in PL/1; the remainder is in ALM (Multics Assembler Language). Virutally all system programming is done in PL/1.
2) Files are called SEGMENTS. Catalogs are called DIRECTORIES.

VIRTUAL MEMORY:

1) 1K Pages
2) Uses "least-recently used" replacement algorithm
3) Hardware addressing in terms of a page within a segment and on offset with a page.

LANGUAGES OFFERED:

BASIC, PL/1, FORTRAN, COBOL, ALM (Assembly Language), and APL. All compilers produce pure re-entrant code.

NUMBER OF INSTALLATIONS: As of YE79, approximately 24 sites.

FILE ("SEGMENT") TYPES:

There are three file types:

1. Ordinary Segments contain user information data. All structure and interpretation is supplied by programs (by naming convention. For example, a.fort would be interpreted as FORTRAN source program a.).
2. Multisegment files: extended length ordinary segment.
3. Directory Segments provide the system mapping between the user's directory and other subdirectories or files. A directory segment is an ordinary segment with special access privileges and information about other segments.

FILE NAMING:

Segments may have an arbitrary number of names and each name can be up to 32 characters in length.

JLM
P&PM-MAS
04-26-79
Rev. 1

MULTICS continued...

FILE PROTECTION:

on segments - read, execute, and/or write
on directories - status, modify, and/or append

OTHER FUNCTIONALITY:

- All supervisory functions are done by subroutine (hide implementation idiosyncrasy)
- Extensive system metering
- High user control of sharing of programs, directories, or data. User is able to specify who may access his segments and how each user may access.
- Automatic file backup and retrieval facility.
- Dynamic system reconfiguration
- Ring structure: very sophisticated protection mechanism
- Unattended operation (with auto reboot)
- Patch-free operating system
- No system edits or generations
- On-line installation of software
- Dynamic linking (no loading)
- Library management tools
- On-line accounting and billing

MULTICS COMMANDS:

abbrev:	gives user mechanism to abbreviate parts of command lines
accept_messages:	permit messages to be received immediately
archive:	combines an arbitrary number of separate segments into one single segment
bind:	create single bound object segment from one or or more unbound object segments
compare/compare_ascii:	compares two segments and lists their differences
compose:	type segments in manuscript form
copy:	copy contents of one segment into another segment
debug or probe:	invokes interactive debugger tool
delete:	delete specific segment from system
dprint:	route output to system printer
dpunch:	route output to system card punch
edm/qedx:	invoke respective editor desired
exec com:	execute command lines found in an "exec-com" segment
gcos:	invoke GCOS environment
io_call:	command interface to i/o system
list:	list segment or directory information
mail:	send mail to another system user
print:	print a specified segment
send_message:	send message(s) to another system user
who/whom:	list users currently on system.

qedx EDITOR COMMANDS:

a(ppend)	append lines after specified line
c(hange)	delete current line, and enter input mode
d(elete)	delete line(s)
e(xec)	execute system command without leaving editor
i(nsert)	insert line(s) before current line
m(ove)	move lines from current buffer to another buffer
p(rint)	print line(s)
q(uit)	quit editor
r(ead)	read segment(s) into editor
w(rite)	write specified lines into specified segment
x	give status of all buffers in use
=	print current line number
g(global)	
v(exclude)	
b(buffer)	

Most editing requests are preceded by an address specifying the line(s) in the buffer on which the request is to operate. Lines in the buffer can be addressed by absolute line number; relative to the "current" line addressing; context; or by a combination of these.


IBM

CP-67/CMS

Description: CP-67 and CMS work together - CP-67 creates a "virtual machine" to each user and CMS creates a timesharing user interface. CP-67 handles the virtual memory management, I/O control, and processor allocation such that a CMS user feels he/she has control of the entire machine. A virtual machine is defined (by IBM) as a functional simulation of a computer and its associated I/O devices. CMS, in turn, handles the file system and interactive interfaces for the user and also handles limited device management.

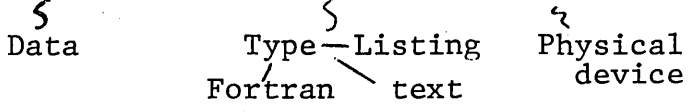
Number of CP-67/CMS/VM370 Installations: (June, 1978) 135

CP Description: CP is a virtual machine monitor. It creates individual machines by multiplexing one actual machine. Although it is usually run with CMS, it can support other IBM operating systems.* Each virtual machine has its own console (terminal) and can be individually controlled. Consoles are typically mapped onto one or more actual console devices. Each virtual machine has up to 16 M Bytes of virtual memory. Total "real" memory is typical ≤320KB for CP only. Paging is accomplished via dynamic address translation hardware.

CMS Description: CMS is designed for one user in a VM environment. To handle multiple timesharing users under CP, there must be multiple copies of CMS. Because of this, sharing of other user's files is difficult. VM/370 files have to be rewritten to run CMS-67.

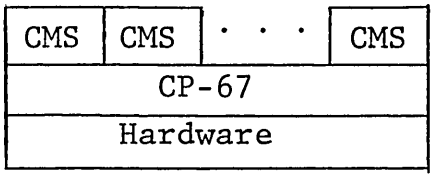
Current Release:
Release Cycle:

File Types/File Naming: <file name> <extension> <device>



* Including OS/360, DOS, RAX and DOS/APL

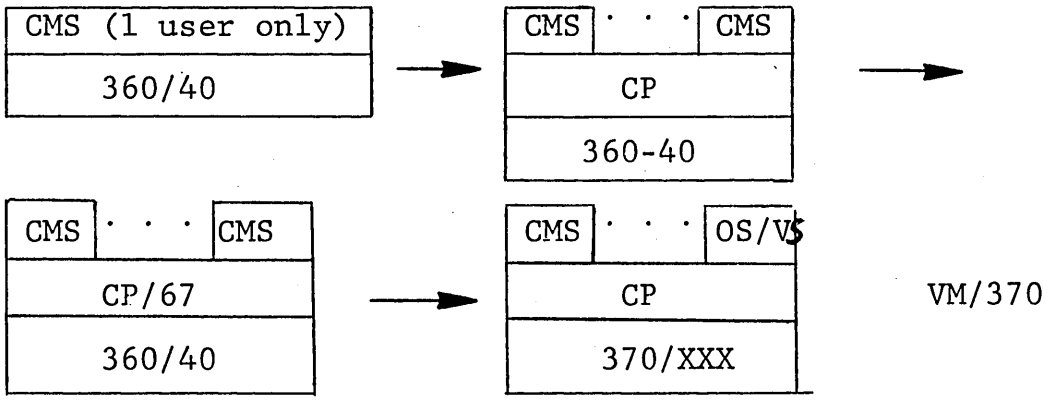
Interfaces:



CMS handles file system and interactive interfaces. It cannot handle any memory or processor management. CP handles virtual memory management, I/O control, and processor allocation; it cannot touch the file system or any user interfaces.

CP-67 functions can be executed directly from CMS by using the CP Function Command.

Development History:



Structure: (See Development History)

Processor Management: Goals are to limit thrashing and give preference to interactive jobs. This tends to give larger programs higher preference than small ones. CP must support a priority system to run OS/VS systems.

I/O Allocation & Control: invisible to user and unaccessible through CMS. Three device management strategies are used:

1. spooled - supports I/O to unit record devices
2. dedicated - magnetic tape. Contradictory to virtual machine concepts to have a dedicated device!!
3. shared - disk and communications.

Languages Supported: (same as OS/360): Assembler, FORTRAN IV, PL/1, SNOBOL, BRUIN (Brown University Interpreter, an interpretive language).

CP Console Commands

BEGIN: begins execution at specified address.

CLOSE: releases spooling areas for input from card reader or output area for printer or card punch.

DETACH: releases specified device from user's virtual machine configuration.

DISCONNECT: allows a user to disable terminal and leave his virtual machine running.

DUMP: print contents of specified register(s), core location(s) or psw on offline printer.

IPL: simulates the Initial Program Load sequence on specified unit. (prior to configuring device to user's virtual machine.)

LINK: dynamically attaches virtual disks.

LOGOUT: release user's virtual machine.

MSG: type message at terminal of person whose USERID is specified.

PURGE: erase spooled input or output files on device.

QUERY: types out information about users on the system.

SPOOL: directs spooled output and controls the reading of spooled input.

XFER: controls the passing of card-image files between users.

CMS Commands

ALTER: change filename, filetype, and/or filemode of user's permanent file.

ASSEMBLE: use the OS/360 F level assembler.

BRUIN: invokes Brown Univeristy Interpreter (desk calculator and stored program mode).

COMBINE: concatenate permanent files into new file specified.

COMPARE: compare two disk files.

CPFUNCTION: issues CP console functions without leaving CMS.

CNVT26: convert BCD files to EBCDIC.

CVTFV: convert fixed length record files to variable length records.

DEBUG: enter interactive debugger.

EDIT: enter editor.

ERASE: delete permanent file.

EXEC: run an EXEC file containing one or more CMS commands.

LOGOUT: logs off CMS session and returns to CP level.

OFFLINE: creates disk files from card input - then prints an offline printer or punches an offline punch.

PRINTF: display on terminal contents of perm file.

SETERR: set error overrides.

START: begins execution of loaded program(s) at specified (or default) entry point.

TAPE: write contents of CMS disk file onto magnetic tape.

EDIT Commands: blank: removes chars. starting from column 1 from current line.

change: replace character string with a second character string.

file: write file to perm file

find: (similar to locate): find first occurrence of string and print line.

insert: insert one line immediately following current line.

locate: print line containing first occurrence of string that is below current line.

print: print specified line(s) - default is print current line.

top: place line printer immediately before first line of file.

References:

- CP-67/CMS User's Guide, IBM Scientific Center Report #320-2015, Oct., 1967.

- ACT/TAG presentation material, January, 1979.

- Meyer & Seawright, "A Virtual Machine Time Sharing System", IBM Systems Journal J, volume 9, #3, 1970.

- 360/67 Cambridge Time Share System CP-67, May, 1969 CL #I006702032, IBM Scientific Center #320-2032.

IBM ETSS/II

DESCRIPTION:

DOS/VS Entry Time Sharing System/II (ETSS/II) is a time sharing system designed for entry level System/370 Computers (Model 115 through 148) and the 4331 and 4341 E-Series machines.

DEVELOPMENT HISTORY:

ETSS was originally designed as a Field Developed Program (FDP). In late 1977, ETSS/II became available as an extension of ETSS.

CURRENT RELEASE/RELEASE CYCLE:

Release 3.0 (before E-Series announcement - possible new release(s) at that time) released Oct. 20, 1978.

STRUCTURE:

The DOS/VS ETSS/II runs under the DOS/VS Control Program (CP) and was tested and released under DOS/VS Releases 33 and 34. One of the following programming products is required to provide a terminal control facility:

> MTCS/VS (Release 1.1), CICS/VS (Release 1.2, 1.3, or 1.4), or Terminal Control System for ETSS/II. ETSS/II is not supported for use with the Terminal Control products MTCS/370 or Terminal Control 5798-CGC which support the original ETSS.

ETSS/II requires a System/370 capable of running DOS/VS. The minimum real storage requirement is 160K for a non-CICS/VS-based system and 192K bytes for a CICS-VS/based system. All system components are written as Assembler Language Macros.

The virtual storage requirement for the control program will generally vary between 44K and 130K, depending on system options, the number and size of file buffers, and the number of command processors that are resident.

The virtual storage requirement for the pseudo-partitions will depend on the number of pseudo-partitions generated and the size of the compilers and other programs being run in these areas. For example, three pseudo-partiitions each running FORTRAN compilations would require no more than 192K (3 X 64K) virtual.

The amount of real storage that must be page-fixed for the control program will generally vary between 8K and 12K, depending on ETSS/II options. Note that this does not include the real allocation requirement of the CICS/VS, TCS, or MTCS/VS system, which must be added to the ETSS/II page fix requirement unless using the Dual Partition Execution option in which case the CICS/VS real requirement would actually be a part of a separate DOS/VS partition.

OPERATING SYSTEM:

Foreground vs. background: All system commands and context editor commands are executed in what is defined as a foreground sub-task. On the other hand, all compilations and program executions are performed as a background sub-task. The foreground sub-task and the background sub-task(s) operate within a single DOS/VS partition. Under MTCS/VS and under the ETSS/II standalone terminal controller, certain commands that execute in the foreground would take too much time and, thus, exclude other terminal users. The ROLLOUT facility of ETSS/II automatically alleviates most of these delays.

A job submitted for execution is queued for execution until a block of virtual storage becomes available. The block of virtual storage is called a "pseudo-parition." An optional feature of the DOS/VS Entry Time Sharing System/II allows pseudo-partitions to be protected from each other by their own storage protect keys. Pseudo-partitions have their own DOS/VS communication region, work files and GETVIS area.

Time-sliced execution: Once a job has been successfully scheduled into a pseudo-partition, it will contend for task resources with other active execution requests in a time-sliced manner.

The library file: Each user of the DOS/VS Entry Time Sharing System/II has access to at least one library within the library file. He or she may own this library exclusively or may share it with other users. Even though the library file may be a large, multi-volume data file encompassing dozens of user libraries, an individual terminal user has access to only the PRIVATE library or libraries assigned to him, the system common library, and any PUBLIC libraries. Each library has a directory which may contain as many entries as are allowed by the local installation manager.

LANGUAGES SUPPORTED:

Assembler, BASIC, FORTRAN, COBOL, PL/1, and RPGII.

OTHER FUNCTIONALITY:

- Submit-to-batch capability via POWER/VS allows user to initiate jobs from the terminal for later batch processing.
- Interactive programs possible.
- A full screen context editor that allows user to create, edit, and/or save file(s) directly on IBM 3270 video screen.
- Command list processor with logic capabilities (allows user or installation to build their own commands).
- Library auditing (monitor of additions, changes, and deletions to library file).

NUMBER OF ETSS/II INSTALLATIONS:

FILE TYPES/FILE NAMING:

EBCDIC files.

COMMANDS:

System commands: System commands, such as /SAVE,/PURGE or /TABSET, tell the system to perform such functions as saving members in the library, purging members from the library, or setting logical tab positions.

Context editor commands: Context editor commands such as LOCATE, CHANGE, or INSERT generally relate to the location or mainpulation of data that the user has entered.

Job entry statements: Job entry statements look very much like system commands. However, the effect of a system command is realized immediately, whereas the effect of a job entry statement is not realized until the job is actually run as a background task. Job entry statements appear in job streams specified by the terminal user.

Dump commands: Dump commands may be entered if the dump option has been set and a pseudo-partition terminates abnormally. These commands allow the terminal user to display registers and storage areas and to locate data areas within the pseudo-partition.

Procedural commands: Certain pre-defined procedures are made available with the ETSS/II system. These procedures provide functions such as compiling programs, loading and executing programs, storing object decks, sorting library members, and several other utility functions. The supplied procedures also may be used as examples for user-written procedures.

ETSS/II ENHANCEMENTS OVER ETSS:

- Full screen editor capability allows user to:
  - Create one or more new files.
  - Make file changes by altering corresponding records on 3270 display screen.
  - Edit multiple files concurrently on same display screen.
- New File Change Reporting Facility.
- The terminal control (CICS/VS, MTCS/VS, or the ETSS/II stand-alone terminal controller) system generation procedures are completely separate, unlike ETSS where modules are link-edited together.
- If CICS/VS is used as the terminal control program, an optional feature called the Dual Partition Executive option, allows CICS/VS to run in one DOS/VS partition even through CICS/VS continues to service all ETSS/II terminal activity.

TAPE SUPPORT:

Tape drivers are not required for operation of ETSS/II. However, one tape drive is required for installation of the system.

TERMINAL SUPPORT:

The following terminal types are supported:

- IBM 2740 Model 1 Communication Terminal with record checking and station control (both features are required).
- IBM 2740 Model 2 Communication Terminal with record checking (required). If the buffered receive feature is installed, only the largest buffer size (440) will be supported by the Time Sharing System. If this feature is not installed, the minimum buffer size is adequate. The buffered receive feature will improve overall performance if multi-dropping IBM 2740/3767 terminals on a single line.
- IBM 2741 Communication Terminal is supported under CICS/VS only.
- IBM 3767 Communication Terminal, when equipped with the appropriate features to make it appear as one of the supported IBM 2740 or 2741 configurations.
- IBM 3277 Display Station (Model 2 only) through the IBM 3272 or 3271 Model 2 Control Unit with, optionally, one or more IBM 3284/3286/3288 Model 2 Printers.
- The recommended keyboards for the IBM 3277 are the EBCDIC typewriter keyboard with the 12 program function keys or the operator console keyboard with the 12 program function keys. While the program function keys are not required, their inclusion is recommended.
- IBM 3275 Display Station with control with, optionally, one IBM 3284 Model 3 Printer.
- The system console is supported as an ETSS/II terminal under CICS/VS only.
- Sequential devices such as card reader, line printer, tape, and disk are supported as ETSS/II ter-inals under CICS/VS only. Under CICS/VS, the main system printer (perhaps via POWER/VS) may be specified as a hardcopy terminal destination.

block multiplexer

The IBM 3272 control units may be locally attached to a multiplexer, or selector channel. The IBM 3271/3275 control units may be remotely attached via non-switched communication lines under CICS/VS or MTCS/VS. In addition, under CICS/VS the IBM 3275 may be attached to dial-up switched lines. The IBM 2740 (and 2767 in 2740 mode) may be remotely attached via non-switched communication lines. The IBM 2741 terminal is supported on switched or non-switched lines only under CICS/VS and only as point-to-point devices.

TERMINAL SUPPORT...continued:

Remote terminals/control units (IBM 3271, 3275, or 2740) of the same type may be multi-dropped from the same communication line. However, if multi-dropping IBM 2740 (or 3767 in 2740 mode) terminals, it is wise to use the buffered (2740 Model 2) terminal. When including remote 3270s in the configuration, a line speed of 4800 BPS or higher is desirable, especially if multi-dropping 3271s or 3275s.

IBM ICCF

Description: ICCF (Interactive Computing and Control Facility) is a limited time sharing facility available on the IBM 4300 series computers.

General Application:

DEVELOPMENT HISTORY: ICCF is a replacement of the field developed Entry Time Sharing System (ETSS).

CURRENT RELEASE / RELEASE CYCLE: The initial announcement of ICCF was made following the 4331/4341 announcement in early 1979.

STRUCTURE:

While it is running, ICCF occupies a single DOS/VSE partition, which is divided into the four logical areas listed below. These areas are shown in Figure

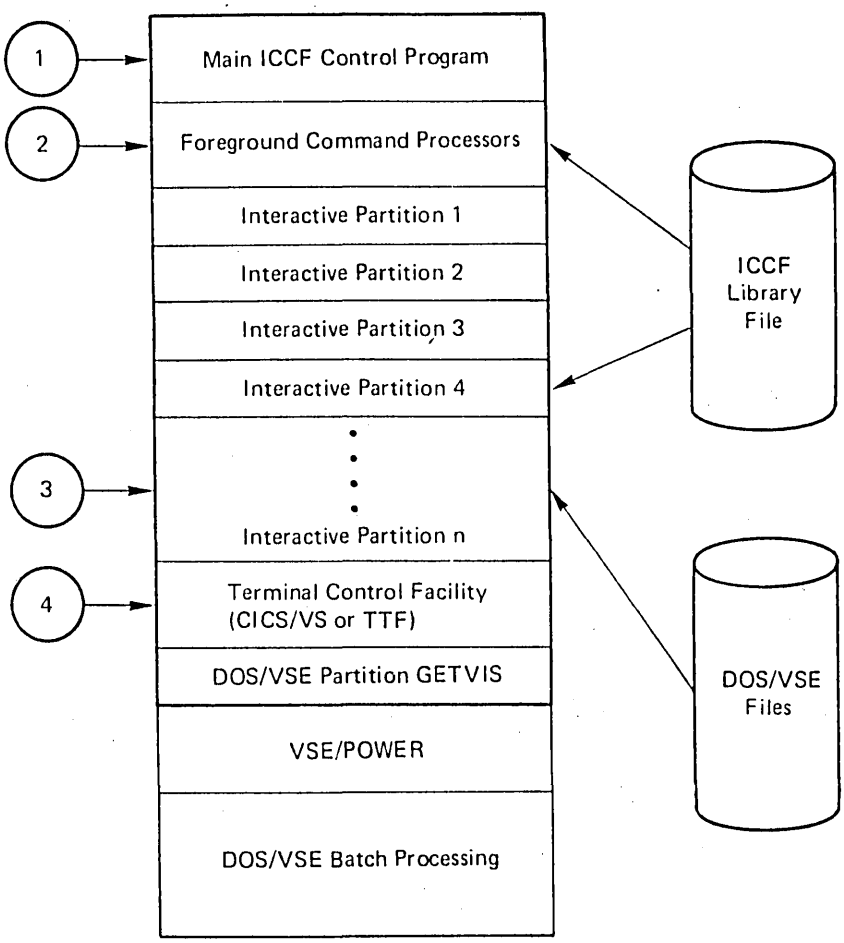
1 Main Control Program

2 Foreground Command Processors

3 Background Interactive Partitions

4 Terminal Control Facility

STRUCTURE: continued...



Logical Components of ICCF.

Main Control Program

The main control program supervises the activities of all ICCF components, except terminal I/O activities, which are carried out by the terminal control facility. The main control program is always in storage. Its responsibilities include attaching and monitoring subtasks, scheduling jobs, intercepting SVCs, performing library I/O, and allocating buffers.

Foreground Command Processors

The command processors read and interpret commands and data from the terminal and carry out requested functions such as saving members in libraries, printing a member on the terminal, or entering a line of input.

Terminal Control Facility

Terminal control in ICCF is handled by either Customer Information Control System/VS (CICS/VS) or by the Terminal Transaction Facility (TTF), a terminal control program that is part of ICCF.

STRUCTURE: continued...

Background versus Foreground

Foreground processing in ICCF refers to the execution of all system and editor commands. To maintain an adequate level of response to terminal users who are doing command work, foreground execution has a higher priority than background execution.

Background processing is concerned with the compilation and execution of user programs. When the interactive facility receives a request for compilation or execution of a user program (via an /EXEC,/RUN or /ENDRUN command), it queues the request for execution.

Interactive Partitions

The background partitions in ICCF are called "interactive" partitions. Interactive partitions have similar characteristics to DOS/VSE partitions: they have their own communication region, their own storage protect key, and their own GETVIS/FREEVIS areas. Interactive partitions may be associated with up to four classes, which you may use to direct certain kinds of jobs to certain partitions. The size, number, and other characteristics of interactive partitions are determined at generation time but may be altered at ICCF startup. ICCF can support up to 35 interactive partitions, plus one for the terminal control facility.

Time-Sliced Execution

Once a job has been scheduled into an interactive partition it will contend for task resources with other active execution requests on a time-sliced basis. When it begins execution it continues to run until one of the following events occurs:

- A conversational read is encountered, which causes job execution to be suspended until the conversational input has been entered, after which the job is again made eligible for execution;
- The print buffer area is full, at which time the job is suspended until all the print lines have been transferred to the terminal. Then the job is made eligible for execution again;
- The interactive partition's time slice elapses, which causes the execution to be suspended until a new time slice becomes available;
- The job terminates, at which point the remaining output is transferred to the terminal and the interactive partition is made available to another job.

OPERATING SYSTEM: ICCF runs under the DOS/VSE System Control Program (SCP) with the VSE/Advanced Functions. VSE/Power is required for a submit-to-batch facility. VSE/VSAM is supported.

LANGUAGES SUPPORTED: VS/BASIC, FORTRAN, DOS/VS COBOL, DOS/VS GPR-II, PL/1, and assembler.

ADDITIONAL FUNCTIONALITY:

- VSE/POWER allows time sharing jobs to be executed in batch environment.
- DOS/VSE Interactive Debug Package (a FDP) can be used with ICCF for online debugging.

EDITORS: ICCF provides two editor programs: a context editor, which can be used with any input/output terminal, and a full screen editor, which can only be used with the IBM 3270 Information Display System. The editor commands in ICCF are carried out at the time they are entered.

The context editor is a text manipulation program that enables users to create and modify data sets either in his/her library or input area. The context editor consists of commands and macros that allow one to:

- Locate strings of information within a member of a library;
- Move a line pointer from one record to another;
- Perform control functions, such as setting tabs;
- Replace or insert data lines or character strings;
- Edit members of data sets by line number.

The context editor allows global changes to be made to a line, to a number of lines, to a column (defined zone of a line, or lines) or to an entire file. With the editor stack facility, one is able to store frequently-used sequences of commands and execute them with a single command.

The full screen editor is designed to use the full range of 3270 terminal features and offers the following advantages:

- Data can be changed anywhere on the screen;
- Multiple data sets can be edited using split screen control;
- Only changed data is transmitted (faster data transmission);
- Minimum CPU activity (multiple functions per interrupt);
- Wide range of user defined PF key options.

The full screen editor is a context and sequential-number oriented editor, primarily designed for maintenance of program source material under ICCF, although it can also be used to edit other data type members. Data can be modified anywhere on the display simply by positioning the cursor and entering the desired change. Only after ENTER has been pressed does the editor check the modifications to the screen and make permanent copies of these changes within the member. The full screen editor also has a split screen facility to divide the full screen into logical "sub-screens".

REFERENCE: VSE/ICCF General Information Manual 1979.
CL #I4300S6066M Program #5746-TS1

IBM MUSIC

DESCRIPTION: The McGill University System for Interactive Computing (MUSIC) is a standalone, time sharing system. It operates in both the VM/370 and dedicated machine environments. MUSIC is an IBM Field Developed Program (FDP) offered on an "as is" basis.

GENERAL APPLICATION: Educational environments where time sharing and batch jobs are run concurrently.

DEVELOPMENT HISTORY: Developed at McGill University, Montreal.

CURRENT RELEASE / RELEASE CYCLE: MUSIC Release IV released December 1978.

STRUCTURE: MUSIC can run on the System/370 models 115 through 168 CPUs. On the 3031, 3032, and 3033, MUSIC can only be run under VM/370. Minimum storage requirements is 256K which supports about ten terminals and batch. Each additional terminal adds 0.8K to this requirement. Up to 250 terminals can be supported. VS APL requires about 160K additional storage. MUSIC requires one channel multiplexer (with channel address of 0) and one disk (2 spindles) device to run.

MEMORY REQUIREMENTS: MUSIC requires 110K for system code and buffers. A 240 KB S/370 can support up to 60 users while a 384 KB S/370 can support up to 120.

LANGUAGES OFFERED: FORTRAN IV, VS BASIC, ANS COBOL, VS APL, Assembler, PL/1. APL has disk calculator. Programs prepared under MUSIC are compatible with and execute under OS/VS.

ADDITIONAL FUNCTIONALITY:

- STATPAK (interactive statistical problem-solving tool)
- MUSIC's word processing capability is provided through MUSIC/SCRIPT which is very similar to CP/67-CMS SCRIPT.

NUMBER OF INSTALLATIONS:

FILE NAMING/FILE STRUCTURE: MUSIC has a common file structure (unlike CP-67/CMS).

TERMINALS: Terminals supported are:

- IBM 3277/3278 Model 2 Local Display Terminals (1920-character version).
- IBM 3767 Communications Terminal operating in 2741 compatibility mode.

TERMINALS: continued...

- IBM 2741 Communication Terminal with either EBCDC or correspondence code configurations. The IBM Communicating Mag Card Selectric Typewriter (CMCST) is supported.
- IBM 1050 Data Communications Adapter with the 1052 Printer-Keyboard and optionally the 1054 Paper Tape Reader, 1055 Paper Tape Punch, 1056 Card Reader, and 1057 Card Punch.

References: MUSIC IV Interactive Operating System, September 1978, CL #0370 0 6068A.

P&PM-MAS
03-20-79
Rev. 1

IBM TSO

Description: TSO is a general purpose time sharing capability. It runs within the foreground with OS/VS2 (MVS or SVS), or Tone Software Corporation's OS/VS1 in background operation. The MVS implementation accommodates interactive processing but not 360/TSO because of real-memory based design incompatibilities. TSO runs under OS/VS2-1 in a similar manner to OS/MVT. VSPC (TSO's successor) offers superior performance and function than TSO and will eventually succeed TSO.

General Application: General purpose time-sharing for all IBM markets.

History: The current 370/TSO is a carry-over from 360/TSO.

Structure: TSO, less optional components, requires about 240KB exclusive of operating system requirements. A TSO environment requires about 1MB, exclusive of operating system requirements. In SVS, multiple timesharing users share a timesharing region and their active pages (working set) are swapped (block paged) to the paging data sets. In MVS, each timesharing user is assigned to an individual virtual address space.

Performance under MVS has been less than desirable. VSPC requires less real storage than TSO and offers superior performance. Under OS/VS2, TSO can handle a maximum of 42 timesharing users OR 63 batch users. Mixed mode operation degrades both factors.

Operating System:

Current Release/Release Cycle:

Languages Offered: BAL, FORTRAN, COBOL, PL/1, VS BASIC (compilers are separately priced).

Additional Functionality: TEST debugging environment allows controlled execution of a program, interrupting it at dynamically specified points.

P&PM-MAS
02-26-79
Rev. 0

File Naming/File Types: EBCDIC Files.

EDIT Commands:

BOTTOM: set line points to last line

CHANGE: replace specified, character string in specified lines within second string

DELETE: delete specified line(s) from data set.

DOWN: move line pointer toward bottom of data set the specified number of lines.

END:

FIND: find first occurrence of specified character string and print that line. (Search begins at current line.)

INPUT: same as INSERT for non-line numbered data sets.

INSERT: insert one or more lines following current line.

LIST: print specified line(s) of the data set.

RENUM: assign or renumber line numbers in a data set.

RUN:

SAVE: store data set.

TOP: set line pointer to zero (before first line of data set)

UP: move line pointer toward top of file the specified number of lines.

Terminal Supported:

Switched or nonswitched lines: 3767 Communication Terminal, 5100 Computer Systems.

Switched or nonswitched point-to-point lines: 2741 Communication Terminal, 1050 Data Communication System.

Nonswitched lines only: 2845 Display Control, 2848 Display Control.

Reference: OS/VS2 TSO User's Guide.
Auerbach Reports

TSO in VS2 differs from its MVT counterpart in several respects:

- The number of regions that can be assigned to foreground jobs is increased from 14 for TSO in MVT to 42 for TSO in VS2.
- The paging supervisor executes all swaps.
- The LSQA is no longer taken from the region; each TSO region is allocated one segment of LSQA.
- Swapping is a process by which the valid pages (addressable pages) in a user's region are usually block paged to external page storage and always block paged from external page storage.
- Background regions must have 100 percent backup in external page storage. For example, a background job that requests a 128K region is allocated 128K (32 pages) of external page storage as backup.
- Most TSO jobs do not continually need an amount of external page storage equivalent to the region sizes requested. Consequently, foreground jobs are backed up by an installation-specified amount of external page storage that is a percentage of the total amount of external page storage required.

While TSO functions much the same under MVS, a few modifications to the time-sharing option are necessary to take advantage of the facilities offered by the new operating system. These modifications include:

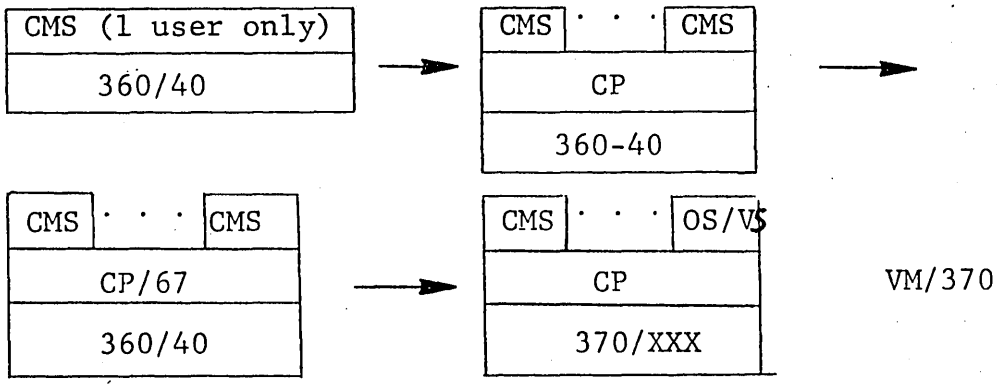
- A separate address space for each user, assigned at log-on
- Extended data-handling commands, allowing allocation of multivolume and multiunit, non-direct access, VSAM and virtual I/O, and concatenated data sets other than VSAM or ISAM. The installation can selectively authorize time-sharing users to allocate data sets requiring volume mounting; under installation control, time-sharing users can also direct SYSOUT data sets to remote stations
- Frequently used time-sharing commands or service routines can be placed in the pageable link-pack area (PLPA) without reducing the amount of real storage available. When time sharing is inactive, the time-sharing programs in the PLPA do not require real storage
- External page storage is used for swapping, eliminating swap data set. It is possible, however, to designate a separate swap data set for those pages marked "fixed" by the operating system

IBM VM/370

Description: VM/370 is both an operating system (system control program) and a multiaccess time-shared facility that permits users to develop virtual machines.

General Application:

Development History: VM/370 is the successor to CP67-CMS.



Current Release/Release Cycle:

Release 6

Structure: VM/370 has a minimum real storage requirement of 240K bytes; a minimum 512KB system can support the VM/370-CMS environment. Layered overhead resulting from VM/370's hierarchical structure is alleviated by VM/370 Assist (microcode @ no charge) feature available with the 370/135, 145, and 158. VM/370 consists of two major components: the Control Program (CP) and the Conversational Monitor System (CMS). Through these the user creates and controls the virtual machines he desires. Remote spooling capabilities are provided by a multitasking supervisor called RSCS (Remote Spooling Communications Subsystem).

VM/370 works on the System/370 Model 135 through 168, except for Models 155 and 165.

Processor Management: Goals are to limit thrashing and give preference to interactive jobs. This tends to give larger programs higher preference than small ones. CP must support a priority system to run OS/VS systems.

I/O Allocation and Control: Invisible to user and unaccessible through CMS. Three device management strategies are used:

1. Spooled - supports I/O to unit record devices
2. Dedicated - magnetic tape. Contradictory to virtual machine concepts to have a dedicated device!!
3. Shared - disk and communications.

VM/370 CP: CP manages the resources of the System/370, including CPU time, to create and control multiple concurrent virtual machines that can run under different operating systems. In effect, each user appears to have access to the complete functional capabilities of the System/370 including virtual storage of 8K to 16 million bytes.

I/O devices that are part of a virtual machine's configuration normally require real device equivalents, except for unit record devices and virtual 2311 Disk Storage Drives. The former can be simulated by CP using disk, while the latter are mapped by CP onto 2314 or 2319 disks. Up to two full 2311s can be mapped onto a 2314 or 2319 in this manner.

VM/370 CMS: CMS operates under the direction of CP and provides a general-purpose conversational time-sharing capability suitable for problem solving, program development, and general conversational work. Included are commands for program compilation and execution, file and utility manipulation, and control and debugging.

Work done by each virtual machine is scheduled and controlled by the selected operating system running under VM/370. The time and system resources required to do the work are managed by the virtual machine facility.

A CMS enhancement under VM/370 now allows users to remotely read DOS sequential data files. Access is limited to reading, however.

Operating Systems: Users can select any of IBM's System/360 operating systems - including DOS and OS - or any of the new operating systems to run under VM/370. There are, however, some major restrictions:

- No machine or program timing dependencies can exist.
- No use can be made of the DIAGNOSE instruction for machine control or of the Read Direct or Write Direct instructions.

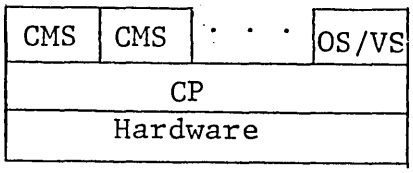
Operating Systems: continued...

- No channel program can be changed by the CPU or the channel during the interval between execution of the Start I/O instructions and the channel end interrupt, except that performed by the OS indexed sequential access method.
- No DOS emulation is permitted. VM/370 users can, however, run DOS and one or more of the OS systems concurrently in virtual machine mode. DOS sequential data files can be remotely read.

Language Supported: COBOL, FORTRAN, BASIC, PL/1, and BAL. There is no charge for CMS systems support, but compilers and optional components are separately priced.

Number Of Installations:

Interfaces:



CMS handles file system and interactive interfaces. It cannot handle any memory or processor management. CP handles virtual memory management, I/O control, and processor allocation; it cannot touch the file system or any user interfaces.

CP functions can be executed directly from CMS by using the CP Function Command.

CP Console Commands

BEGIN: begins execution at specified address.

CLOSE: releases spooling areas for input from card reader or output area for printer or card punch.

DETACH: releases specified device from user's virtual machine configuration.

DISCONNECT: allows a user to disable terminal and leave his virtual machine running.

DUMP: print contents of specified register(s), core location(s) or psw on offline printer.

IPL: simulates the Initial Program Load sequence on specified unit. (prior to configuring device to user's virtual machine.)

LINK: dynamically attaches virtual disks.

LOGOUT: release user's virtual machine.

MSG: type message at terminal of person whose USERID is specified.

PURGE: erase spooled input or output files on device.

QUERY: types out information about users on the system.

SPOOL: directs spooled output and controls the reading of spooled input.

XFER: controls the passing of card-image files between users.

CMS Commands

ALTER: change filename, filetype, and/or filemode of user's permanent file.

ASSEMBLE: use the OS/360 F level assembler.

COMBINE: concatenate permanent files into new file specified.

COMPARE: compare two disk files.

CPFUNCTION: issues CP console functions without leaving CMS.

CNVT26: convert BCD files to EBCDIC.

CVTFV: convert fixed length record files to variable length records.

DEBUG: enter interactive debugger.

EDIT: enter editor.

ERASE: delete permanent file.

EXEC: run an EXEC file containing one or more CMS commands.

LOGOUT: logs off CMS session and returns to CP level.

OFFLINE: creates disk files from card input - then prints an offline printer or punches an offline punch.

PRINTF: display on terminal contents of perm file.

SETERR: set error overrides.

START: begins execution of loaded program(s) at specified (or default) entry point.

TAPE: write contents of CMS disk file onto magnetic tape.

EDIT Commands: blank: removes chars. starting from column 1 from current line.

change: replace character string with a second character string.

file: write file to perm file

find: (similar to locate): find first occurrence of string and print line.

insert: insert one line immediately following current line.

locate: print line containing first occurrence of string that is below current line.

print: print specified line(s) - default is print current line.

top: place line printer immediately before first line of file.

IBM VSPC

Description: VSPC (Virtual System, Personal Computing) is a set of program products that can operate concurrently with DOS/VS, OS/VS1, or OS/VS2 (MVS) in background operation. It is a replacement for TSO wherein implementation is virtualized (VTAM, VSAM) and more integrated to provide better performance and added flexibility. A person running VSPC must have a working knowledge of DOS/VS, OS/VS1, or OS/VS2 (MVS).

General Application: Special-purpose software for novice users, particularly managers. It will handle a maximum of 12-14 concurrent users.

Development History: VSPC is the successor to TSO and it offers superior performance and functionality than TSO.

Current Release/Release Cycle:

Operating System: VSPC operates under DOS/VS (Release 33 or later) on System/370 models 135, 138, 145, 148, 155-2, and 158 with at least 256K bytes of real storage to support three terminals. It can run in a degraded performance mode on the 370/115 and 370/125. Under DOS/VS, VSPC requires VTAM for terminal access, VSAM for the device - independent access method for VSPC library support, and SAM. POWER/VS is required for VSPC conversational RJE. NCP Release 4.1 or later is required for remote terminals.

VSPC will run under OS/VS2 (MVS) on all System/370 models supported by OS/VS2 (MVS) with at least 2048K bytes of real storage to support three terminals. Under OS/VS2 (MVS), VSPC requires HASP VTAM and VSAM. NCP Release 4.1 or later is usually required. Under OS/VS2 (MVS), JES2 or JES3 (Supervisor 1 or 2) selectable units are also required. (JES = Job Entry System). For MVS, VSPC can be used on MP/AP configurations.

Virtual Storage Requirements:

DOS/VS, OS/VS1 - The size of the VSPC interactive partition is dependent on the number and type of active terminals, the sum of the active users workspace, the VSAM external file requirements, and the initialization options selected. Additional virtual storage is required for any user-installed foreground processor.

Virtual Storage Requirements: continued...

OS/VS2 - The VSPC interactive environment virtual storage requirement is dependent on the number and type of active terminals, the sum of the active users' workspaces, the VSAM external file requirements, and the initialization options selected. Additional virtual storage is required for any user-installed foreground processors. Note: VSPC under OS/VS2 (MVS) utilizes multiple address spaces to extend the total amoung of virtual storage available to interactive VSPC.

VSPC Real Storage Requirements - The real storage requirement for VSPC is dependent on the number and type of terminals, the functions selected, the installation's other processing (if any), expected traffic, and the desired response time characteristics.

Language Supported: VS APL (also available under CMS), VS BASIC (available under CMS and TSO), VS FORTRAN IV (also available on CMS), VSPC PL/1. Compilers are separately priced and additional to system.

Additional Functionality:

- VSPC Library Manipulation provides a set of facilities to share data files and to keep them secure, as desired. (This function is standard on some other systems.)
- The Full Screen Management facility allows use of 3270 display devices. (The installation must write to own foreground processor.)
- Shared Storage Management provides communication between programs, both in the VSPC foreground and in the batch background.

File Types/File Naming: EBCDIC files.

Number of Installations: (As of November 1977): approximately 25.

Terminal Support: According to the requirements of VTAM, VSPC can be used with the following terminals. For remote terminals, an IBM 3704 or 3705 Communications Controller with NCP (Network Control Program) is required.

Terminal Support: continued...

- IBM 3270 Information Display System
- IBM 3770 Data Communication System (interactive mode only)
- IBM 3767 Communication Terminal, Models 1, 2, and 3 (interrupt capability required)
- IBM 2741 Communication Terminal
- IBM 1050 Data Communication System.

Reference: IBM VSPC User's Guide - CL # IO370L0070

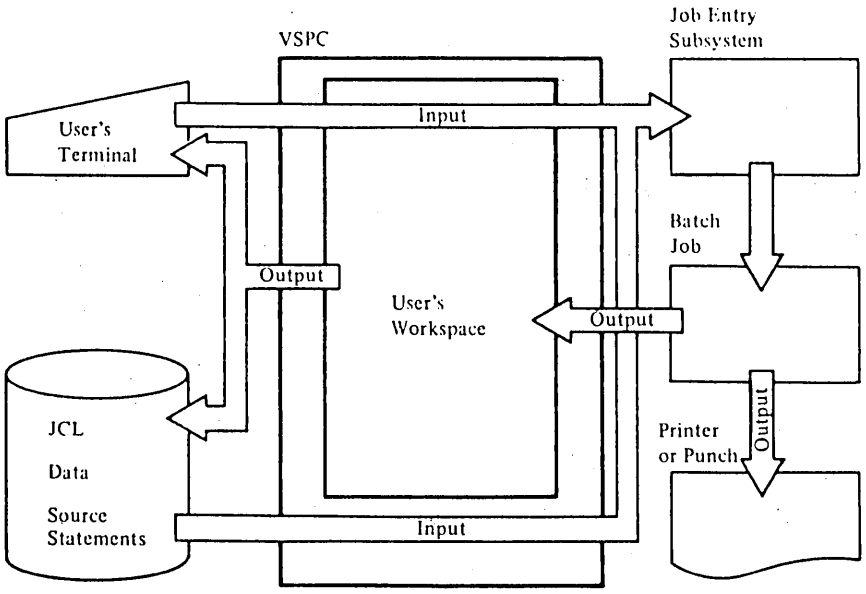


Figure 1. Submitting Batch Jobs and Receiving Output

VSPC Commands

Objective	Terminal Entry	Terminal/Computer Response
Log on to VSPC.	The user types **vspc id=** and his user number.	(At his terminal) VSPC prompts him for his logon password, if he has one. (At the computer) a workspace becomes available.
Prevent messages from other users or the VSPC operator from interrupting his work.	Types **message block.**	(At the computer) no messages are sent to his terminal.
Receive messages.	Types **message open.**	(At his terminal) the last message, if any, broadcast by the VSPC operator is displayed, and new messages are displayed.
Send a message to another VSPC user.	Types **send**, his message, and the other user's number.	(At the computer) the message is available for the other user. (At the other user's terminal) the message is displayed as he indicates.
Learn the names of the files in his library.	Types **query library.**	(At his terminal) the name of each file is displayed.
Delete an unwanted file.	Types **purge** and the name of the file.	(At the computer) the file is removed from his library.
Assign a password to a file.	Types **protect**, the name of the file, and the password.	(At the computer) the password is associated with the file.
Make a file available to certain other problem solvers.	Types **share** and the name of the file.	(At the computer) the file is made accessible to whoever supplies the password, if any.
Submit job(s) for batch processing.	Types **submit** (when the workspace contains the JCL and data for the job(s)). . .or types **submit** and the name(s) of the file(s) that contain the JCL and data.	(At the computer) VSPC verifies that he is authorized to submit jobs, and the job(s) are scheduled to be processed. (At his terminal) the name of each job is displayed.
Inquire whether a job has been processed.	Types **status** and the name of the job.	(At his terminal) the status of the job is displayed.
Acquire output from a job.	Types **load output** and the name of the job.	(At the computer) the output is placed in his workspace.
	Types **list.**	(At the terminal) the output is displayed.
	Types **save** and a name for the output.	(At the computer) the contents of the workspace are stored in his library.

Figure 3 (Part 1 of 3). Example of a User Session with VSPC

Objective	Terminal Entry	Terminal/Computer Response
Run a program or a VSPC command list interactively.	Types **run** (if the workspace contains the program or command list). . .or types **run** and the name of the file that contains the program or list.	(At the computer) if the file is: *An object program,* it is executed. *A source program,* it is compiled, and the resulting object program is executed. *A VSPC command list,* the commands are processed. (At the terminal) messages and other output from the program or VSPC command list are displayed—the user and the computer interact.
Gain access to a file in his library.	Types **load** and the name of the file.	(At the computer) the file is placed in his workspace.
See what the workspace contents look like.	Types **list**.	(At his terminal) the workspace contents are displayed.
Edit	Add lines to the contents.	Types **input**, then types each line.	(At the computer) each line is added to the workspace.
		Types **merge** and the name of another file in the library.	The new file is placed at the specified position in the workspace (or it is, optionally, merged by line number into the workspace).
	Edit the workspace contents.	Types **move** and line numbers.	(At the computer, in his workspace) the indicated line(s) are moved to the indicated position.
		Types a line number without any data.	The indicated line is deleted.
		Types **delete** and line number(s).	The indicated line(s) are deleted.
		Types **find** and a string of characters.	(At the terminal) the line number (and optionally the text) of each line that contains the string of characters are displayed.
		Types **change**, line number(s), old string of characters, and new string of characters.	(At the computer, in the workspace) the old string of characters is replaced by the new string of characters in the indicated line(s). (At the terminal) the line number (and optionally the text) of each changed line and the total number of changed lines are displayed.
	Renumber the lines in the workspace.	Types **renumber**.	(At the computer) the lines in his workspace are renumbered. (At the terminal) the number of lines renumbered is displayed.

Figure 3 (Part 2 of 3). Example of a User Session with VSPC

	Objective	Terminal Entry	Terminal/Computer Response
Edit Commands	Put the workspace contents back in his library.	Types **save.**	(At the computer) the updated workspace contents replace the original file in the library (or they are optionally stored as a new file).
	If the workspace contains a source program, compile it and place the object program in his library. . .	Types **store** and a name for the object program.	(At the computer) the source program in the workspace is compiled, and the resulting object program is stored in the library.
	. . .or run it immediately.	Types **run.**	(At the computer) the source program in the workspace is compiled, and the resulting object program is executed. (At the terminal) the user and the program interact.
	Receive an explanation of an error message sent by VSPC.	Types **?.**	(At the terminal) an explanation of the immediately preceding message is displayed by VSPC AID.
	Erase the contents of his workspace for entering new data.	Types **clear.**	(At the computer) the contents of the workspace are erased.
	Name the new collection of data.	Types **name** and the new name.	(At the computer) the name is associated with the workspace and becomes the name of the collection of data about to be created by the user.
	Log off VSPC, automatically saving his workspace contents until he logs on again.	Types **off continue.**	(At the computer) the system resources held by this user are reclaimed, and his workspace contents are saved in his library under the name CONTINUE. (At the terminal) VSPC displays the length of time the user was logged on and the amount of time he used the CPU (the computer's central processing unit).

Figure 3 (Part 3 of 3). Example of a User Session with VSPC

UNIVAC

UNIVAC
HIGH VOLUME TIME SHARING (HVTS)

Description:

HVTS utilizes the 1100 executive transaction interface package and communication management system (CMS) to provide high volume time sharing. It is designed to handle 50 to 2000 active terminals.

General Application:

High volume, "one run" time sharing.

Development History:

Current Release/Release Cycle: Level 2R1 released in 1978.

Structure:

HVTS operates through the TIP/CMS interface. Because HVTS is a standard 1100 software component, enhancements have been made to other 1100 OS elements to accommodate it in the operating system. TSS, Quota, CMS, and the EXEC itself accommodate (as of June 1977) the HVTS requirement.

HVTS has been designed to provide optimum performance in a dedicated system but it can also operate with batch, demand, real time and transaction programs running in the same system.

Operating System:

HVTS is a part of the Univac 1100 Operating System. The 1100 Executive Operating System is an extension of the EXEC 8 operating system that was first delivered in March, 1968. Three types of program operations are recognized (in order of decreased importance): real-time, demand (timesharing), and batch.

Configuration Guidelines:

Since HVTS provides the capability of handling a range of 50 - 2,000 active terminals, it is obvious that hardware configurations will vary greatly across the range. Several sample configurations are shown to give a general indication of hardware requirements.

EXAMPLE 1 This represents the minimum dedicated HVTS system intended primarily to operate only HVTS, with no demand and only a small amount of concurrent Batch work.

Configuration Guidelines...continued:

EXAMPLE 1
continued

System	1100/11
Main Storage	262K
Disk Drivers	Two (Note: 8433 equivalent, does not include user library storage)
No. Of Active Terminals - 50	

EXAMPLE 11 This represents a small user with a requirement for more terminals and moderate demand and batch concurrent with HVTS.

System	1100/21
Main Storage	393K
Disk Drivers	Two (See note above.)
No. Of Active Terminals - 100	

EXAMPLE 111 This represnets a somewhat larger user - more active terminals and a moderate demand & batch concurrent with HVTS.

System	1100/12
Main Storage	393K
Disk Drivers	Three (See note above)
No. Of Active Terminals - 150	

According to Univac, the terminal user can expect a median response time of three seconds, with 90% of HVTS responses returned within seven seconds. These response times are predicted on a reasonably "typical" workload profile.

Language Supported:

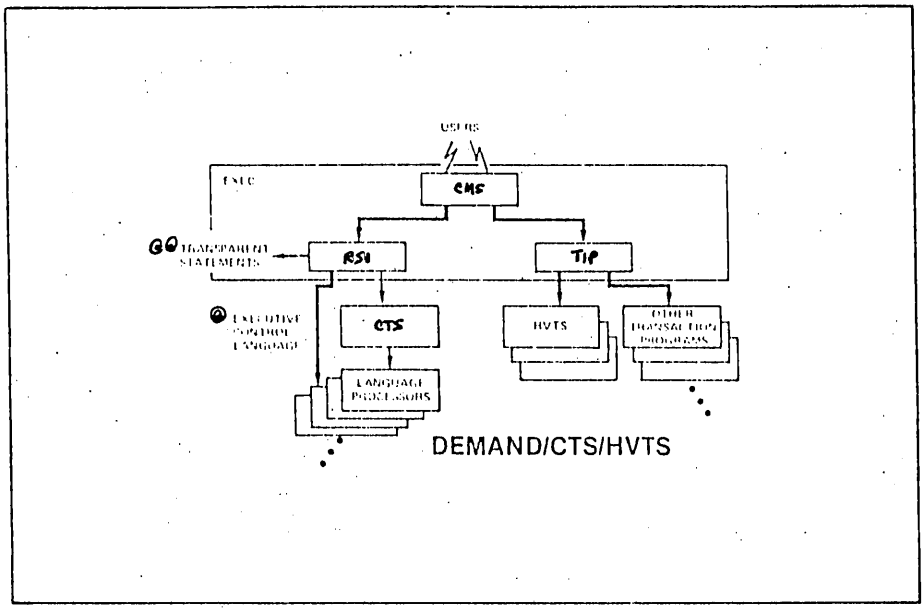
FORTRAN V, BASIC, APL (interactive pgms. possible)

Additional Functionality:

A keyword of 6 or more characters can be abbreviated by typing: <first letter>: <last letter>. For example, "A:N" is equal to "ASSIGN".

Word Size:

Univac 1100 Systems are word-oriented; a word consists of 36 bits. Double-, single-, half-, third-, sixth-, and quarterword operands can be used.



DEMAND/CTS/HVTS

This diagram illustrates how HVTS fits into the 1100 picture. That which you see enclosed in the large box is the Operating System with CMS or the Communications Management System controlling the communication hardware and TIP (Transaction Interface Package) controlling the various HVTS transaction programs.

HVTS Commands:

Common file and data entry commands available to the user include:

- DEBUG
- STATUS
- CATALOG
- PERMIT
- EXPORT
- IMPORT
- LOOK
- NEW
- NUMBER
- SCRATCH
- REPLACE
- OLD
- SAVE
- COMPILE
- UNSAVE
- HELP
- CREATE
- GET
- PUT
- UPDATE
- ATTACH
- DROP
- CONVERT

In addition, the following commands provide the user with editing capabilities:

- RESEQUENCE/MERGE
- GO
- LIST
- LOCATE
- CHANGE
- DELETE
- BRIEF

Both command and editing commands generally have the same syntax and invoke the same functions in conjunction with BASIC, FORTRAN and DATA modes. This means that the DATA mode user can manipulate his data file by using any of the HVTS commands except the three specifically associated with language operation: RUN, COMPILE, and DEBUG. APL is an interpretive language and uses its own command set.

UNIX

UNIX

Description: UNIX is a multi-user time sharing system for DEC PDP 11/40 and up machines. It supports several languages and can replace DEC's RSTS. The Programmer's Workbench (PWB) is a program development facility that was created at Bell Labs.

No field support is available from Bell Labs for UNIX.

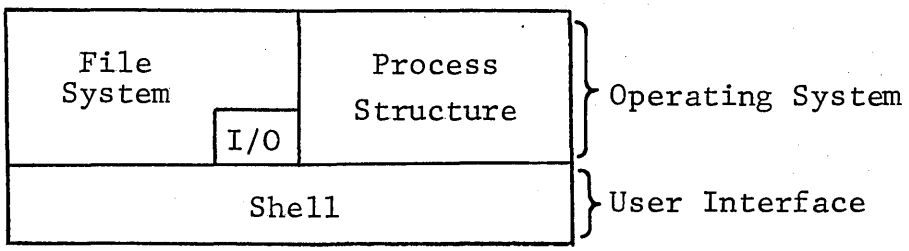
General Application: University community - research environment. Schools use it as a teaching tool because students can interact with UNIX very easily, and also because of the scientific/mathematic functionality UNIX offers. Within the Bell system, operating companies embrace UNIX to do a variety of engineering and administrative applications.

Development History: Bell Telephone Labs participated in the Multics program with G.E. and MIT from 1965 to 1969. When they dropped out, BTL staff member, Ken Thompson, felt some ideas planned for Multics would be worth trying on a small computer. Using a discarded PDP-7, he developed an initial version of UNIX in two years worth of evening work. The initial version was also installed on a PDP-9 computer at BTL and proved successful enough to attract the attention of other BTL staffers. All told, 15 to 20 man-years of BTL work went into the "released" version of UNIX.

Current Release - Release Cycle: PWB 1.0 was released in July 1978.

Weaknesses: No real time applications; lack of flexibility in piping.

Operating System: The UNIX operating system kernel is very small. It consists of 10K lines of "C" and 1K lines of assembly code (200 lines for efficiency reasons, 800 lines to handle hardware interface). The operating system executes the system calls, maintains the file system, and manages the system's resources.



Explanation of the Shell concept: The Shell, or command language interpreter, serves as a user interface to the operating system. It constitutes a full programming language allowing a user or a command procedure to:

- Supply arguments to and run any executable program.
- Very easily redirect standard input and/or standard output.
- Compose compound commands using the following operators:

;	for sequential execution.
|	for simultaneous execution with output of one process "piped" to input of another.
&	for asynchronous operation.
()	parentheses for grouping.
|| and &&	for left-to-right conditional evaluation.

- Control the order in which directories are searched for commands.
- Trace execution of commands for debugging.
- Execute Shell procedures, which are command scripts with substitutable arguments.
- Construct arguments that denote all file names that match a specified pattern (regular expression).
- Use the Shell itself as a command (recursively).
- Collect command usage statistics.

Structure: PWB/UNIX runs on a DEC PDP-11/45 or /70 with at least the following equipment:

- 96K words of memory (2 bytes/word), with memory management.
- Disks: rp03, rp04, rp05, rp06, or equivalent.
- Console typewriter terminal.
- Clock: KW||L or KW||PP.
- Tape: tu10, tu16, or eqivalent.
- Floating point: FP||B or FP||C.

The following equipment is strongly recommended:

- DH|| communications controller(s) with full modem control.
- Full-duplex 96-character ASCII terminals.
- Extra disk drive for system backup.
- DQS B communications controller(s) for RJE.

The minimum memory and disk space specified is enough to run and maintain PWB/UNIX. More will be needed to keep all source on line, or to handle a large number of users, big data bases, diversified complements of devices, or large programs. PWB/UNIX does swapping and sharing of reentrant user code in order to minimize main memory requirements. The resident PWB/UNIX operating system uses 40-48K words, depending on the configuration.

The central processing units (CPUs) for PWB/UNIX are the Digital Equipment Corporation's (DEC) PDP-11/45 and /70 computers. These computers, because of their price and word size (16 bits per word), are really large minicomputers. Although PWB/UNIX can run on hardware costing as little as $60,000, a typical PWB/UNIX system costs about $120,000 and can support 24 simultaneous users with ease. Larger systems can support twice that number. The cost per user-hour of PWB/UNIX is significantly lower than that of most other interface computer systems. PWB/UNIX typically runs unattended.

A large PWB/UNIX configuration (PDP-11/70, 256K words of main memory, fixed and moving head disks) can generally provide reasonable service to between 40 and 48 simultaneous users.

Virtually 100% of the operating system is written in "C".

Languages: "C" (high-level structured language), BASIC, FORTRAN, SNOBOL.

Additional Functionality:

PIPES: interprocess communication - utilizes system read and write calls to allow one executing program to "talk with" another and run in parallel.

FILTERS: extend the pipe notion to the command language interpreter (the "Shell").

UNIX File System:

- The file system is also the I/O interface. It is simple and at the center of the UNIX structure.

File Types:

There are three file types:

1. Ordinary Files contain user information data. All structure and interpretation is supplied by programs. (By naming convention. For example, a.fort would be interpreted as the FORTRAN file "a".)
2. Directory Files are rooted trees that provide the system mapping between the user's directory and other subdirectories or files. A directory file is an ordinary file with special access privileges.

Files Types: continued...

3. System Directories: The system root directory points to all user directories. The System Command Directory contains all system commands and programs.

File Naming:

- File names $\leq$ 14 characters
- Path name specified by:
  directory 1 / directory 2 / ... directory n / file name.
  clarification

File Protection:

on files: read, write, and/or execute.

Number of UNIX Installations: approximately 300.

UNIX COMMANDS

System Commands

comm: print lines common to two files
cmp: compare two files
cp: copy one file onto a second
db: use debug facility
dc: desk calculator
dsw: delete interactively
ed: enter text editor
find: find file(s)
if: conditional command
ln: create a link (link creates link to a file)
login: sign into UNIX
ls: list contents of directory
mail: send mail to designated user(s)
mesg: permit or deny messages
mv: move or rename a file
newgrp: log into a new group
nroff/troff/roff: format text
passwd: change login password
pr: print file
rev: reverse lines of a file
rm: remove (unlink) files
sh (shell): use command interpreter
sleep: suspend execution for an interval
stty: set typewriter options
sort/usort: sort or merge files
wc: word count
who: list users on system

ASCII/BCD: UNIX uses ASCII naming conventions. Word size is 16 bits.

References:

- UNIX Programmer's Manual, Sixth Edition, K. Thompson and D. Ritchie, May 1975, Bell Labs.
- ACT/TAG Time Sharing Lectures notes, January 1979
- Busch, RW, UNIX/Multics Comparison, May 1977.

Special Capabilities:

RJE to IBM System/370 is available on Program's Workbench (PWB/UNIX) version developed at BTL.

UNIX is highly portable (usually three months is all that conversion time required to implement UNIX on a new system).

DATAMATION, December 1978

- RESULTS -

Time Sharing System:	UNIX
Company:	Western Electric Company
Users Reporting:	10
Overall Satisfaction:	3.4
Throughput/efficiency:	3.4
Ease of Installation:	3.1
Ease of Use:	3.8
Documentation:	2.8
Vendor Tech Support:	2.3
Advantages:	flexible
Modifications:	usually necessary - must be done by the user.
Disadvantages:	no maintenance offered.