IFIP '65

DESIGN OF A GENERAL-PURPOSE SCIENTIFIC COMPUTING FACILITY

co-authored by
F. V. WAGNER
Vice President, Informatics, Inc.
Sherman Oaks, California

DESIGN OF A GENERAL-PURPOSE SCIENTIFIC COMPUTING FACILITY

F. V. WAGNER*

Vice President, Informatics, Inc.
Sherman Oaks, California

INTRODUCTION

We will define a General-Purpose Scientific Computing facility to be one which serves a variety of customers with differing needs, and which is not used primarily for business data processing.

Any computing facility is built of all elements which serve its users. In addition to its hardware, the facility includes its software, procedures, constraints, accounting systems, training and reference materials, documentation and the management. Therefore, the design of the facility must consider everything in the facility which helps fulfill the needs of the facility user.

The first step in design is to define the objectives of the installation. After the necessary characteristics of the proposed facility are understood, the detailed requirements of the facility's design can be pinned down. We will review the trends in facility design in the past and show how these have evolved into the current trends. The results of recent innovations in hardware and software will be described and their value assessed. This will form the basis of a forecast of trends which will be significant in future designs.

GENERAL CHARACTERISTICS

Scientific computing is done in facilities with widely varying goals and personalities. A computing facility does not exist only for the sake of computing. Its purpose is to support some other activity. The objectives of the organization supported must be given primary consideration. These objectives cover a wide range. At one end of the scale we can find a pure *production* scientific computing facility. The computing facility of any large aerospace manufacturing firm comes close to this. There, computing is done only to advance the design and production of aircraft, spacecraft, reactors, propulsion units, electronic systems or whatever the particular firm is developing and manufacturing. Of little importance are education, research or advancement of computing methods, except as they contribute to the primary goal of the installation—to support the firm's objectives.

At the other end of the scale, an installation, equally large, may support the computer sciences department of a university. The objectives of such a department may be to advance the state of the art, by seeking new truths and new approaches and by educating scholars in a new discipline. Clearly, such a facility will have different goals and may require a different design.

Facilities can be characterized in many ways. One of these is by the relative emphasis placed on the three fundamental functions which are carried out in nearly every scientific computing facility. *The first is analysis.* Analysis includes all means by which a physical situation is modeled mathematically. The nature of the physical situation is investigated on the computer, by experiments with the mathematical model. Such activity is usually the basis for establishment of a scientific computing facility. Requirements stemming from analysis weigh heavily in the design of the typical facility.

* *Co-author: Jackson Granholm*

But most such facilities, to some extent, perform a second function, called *data reduction*. Here physical tests are performed. Measured quantities resulting from the tests are transformed to the proper engineering units and put in the format best suited for analysis and evaluation. In general, the requirements for data reduction parallel those for doing analysis. However, there are certain areas of conflict.

A third function often found in a scientific computing facility is some form of *on-line operation*. For example, the facility hardware may also be used for simulation. Here it is attached directly to other hardware and it must operate in feedback mode. Or, facility hardware may be directly attached to scientific instrumentation to record test data in real time. Perhaps the computer simultaneously serves as a communications switching device. Clearly, the requirements for a facility which incorporates such functions must differ from those for an installation in which the computer is never attached on-line to other hardware.

Ten years ago an installation's size was considered a dominant characteristic. Size has gradually decreased in importance. What was feasible only for the largest installations ten years ago is now commonplace even among installations with minimal budgets.

REQUIREMENTS

After the characteristics of a general-purpose scientific computing facility are assessed and understood, the next step in design is to translate these into detail requirements. Requirements should be analyzed and specified in detail. Requirements for production computing most be thought about differently from development requirements.

To establish production requirements, the designer must face the conflict between the need to be economical and the need for the facility to be responsive to its users. In the early days of computing, the hourly cost of the central processing unit dominated the thinking of designers. As time went by, it became apparent that there are other costs of equal importance. Among these are costs of the associated electro-mechanical devices like tapes and printers, as well as the costs of operating personnel at the facility.

A more subtle source of cost must not be overlooked. The installation must be designed to minimize the time of users. A properly designed facility can cut the time the user spends in preparing his input or programs. A fact often overlooked in design is that many production programs which run day after day are not truly static. The very fact that they have been in use for a year or more normally causes them to produce variations and configurations which differ slightly, and which may be changed for each production run.

User time can be economized by making output results easy to interpret. Such obvious features as careful identification of different cases are usually taken care of, but often little consideration is given to encouraging the user to increase his productivity by copious use of graphic output. Such output can have dramatic results in calling to the user's attention the real meaning behind the results computed.

Another criterion for scientific computing is installation responsiveness. Commonly called "turnaround time," this is a major parameter in the installation design. The quantity and the number of levels of priorities which will be considered is a major specification for the installation.

The requirements of economy and responsiveness apply in equal measure to the development cycle of the production programs. Development includes all those activities needed *before* a program goes into production status. Requirements for program development differ if one gives consideration to the time the developed program will be in production. The time a program will be in development is influenced by several features of the facility design. From a programmer's point of view, language convenience is certainly important. Increased ease in implementing the mathematical model can produce remarkable economies. The reduction in pure coding errors that resulted from the abandoning of the use of symbolic assembly languages has dramatized the fact that much development time stems from errors in the mathematical model itself.

A difficult decision must be made about the degree of physical access to the computer given the programmer. Should facility design permit physical access to the machine or should it prevent it? Trends have varied over the years.

The quantity and quality of aids to debugging must be faced up to in facility design. This is perhaps the area which has been given more lip service and less attention than any other. Once again it is useful to recognize that there are two kinds of development: of the program and of the mathematical model. What is useful for the one is frequently not useful for the other.

DESIGN TRENDS

Until about 1955, the design of general-purpose scientific computing facilities was based on requirements that were summarized thus: "Scientific computing is the equivalent of numerical analysis. It has very little input and very little output, but has a large amount of arithemetic in-between." This choice bit of misinformation, together with the fact that it was fun for designers to work on the arithmatic units and internal storage circuits, led to the development of early scientific computers with extremely poor input-output capabilities. As soon as the fallacy of this reasoning was recognized, design trends started which have continued to the present day. A very early recognition of the need was when North American Aviation commissioned IBM to equip their 701's with 727 tape units. This permitted input-output to be done tape-to-tape, like UNIVAC I, using off-line card-to-tape and tape-to-printer equipment. This naturally led to development of a rudimentary monitor system on the 701. In turn, it led to the General Motors-North American-Convair 704 monitor which gained wide acceptance in the scientific computing industry. It was the first of the currently widely used batch monitors. Today we find such programs used on the largest systems and at the same time available on systems as small as the SDS 910.

Another important trend in facility design came from the recognition that scientific computing is *not* the equivalent of numerical analysis. Through the last half of the 1950's, there came into prominence such non-numeric applications as natural language processing and the stochastic simulation of physical systems like air battles, and telephone networks. Their behavior patterns are characterized more by logic than anything properly called numerical analysis.

Since 1960, design trends have been influenced by hardware not previously available. Such hardware falls in four classes:

1. Random access bulk storage devices such as disks, drums, magnetic cards, magnetic strips, and soon, large core storage.
2. The increase in speed and decrease in cost of the central computing unit for even the smallest computers.
3. The growth in sophistication of input-output facilities, such as complex interrupt logic and large numbers of overlapped channels.
4. The proliferation of communication equipment suited for data transmission.

These developments are evident in system configurations which dictate radical changes in installation procedures. New system configurations have been made possible by mass random access devices and by the sophisticated input-output facilities. Multiprogramming and multicomputer configurations lead the way to the future. Remote input-output stations have reached a stage of development where the facility designer must give serious consideration to their use.

As distinguished from the hardware system configuration, the general operating philosophy of the installation must nowadays take into account the trend to use a wide variety of high order languages. Only the system programmers and detail computer specialists are now wide users of assembly languages. In the last five years we have seen the applications programmer begin to dominate. He uses such procedural languages as FORTRAN, COBOL, ALGOL, JOVIAL, LISP, etc. These languages are general-purpose in nature. There is now emerging a third class of users. These express their needs to the machine in *Third Level* problem-oriented languages such as APT for numerical control, COGO for coordinate geometry, SIMSCRIPT for stochastic simulation and now some emerging languages whose sole purpose is interaction with graphic input and output devices. Only a foolhardy installation designer of the future will not take into account the needs of this newer class of computer users.

RECENT RESULTS—HARDWARE

A significant and dramatic trend in system configuration is the evolution in the last two or three years of what may be called Continuous Flow Systems. They are aimed at reducing turn-around time. Their approach is to automate completely the handling of a job from the time it is loaded until the printer or plotter outputs results. They are based on multiprogramming techniques, and frequently involve multicomputer configurations. With such systems input can be kept to a minimum. Everything previously introduced into the system can, at least in theory, be stored and available whenever needed. Such a system can be balanced by having just the right number of input hoppers and just the right number of output devices to be kept in continuous motion. Jobs can be handled in the order specified by a priority algorithm, static or dynamic. Currently, the greatest weakness in these

systems is the relative unsophistication of status display, which could enable the operator to help the system fulfill its functions.

Most new computers are designed to operate in a multicomputer environment. To date, however, facility designers usually ignore the fact that all main frames need not be in the same room. Communications links are now sufficiently developed to make feasible a multicomputer network with computers in diverse locations. In the last eight years, much experience has been acquired with tape-to-tape and core-to-core connections via both microwave and land-line connections. Today most multicomputer networks are in military systems such as the Pacific Missile Range Real Time Data Handling System, the SAGE Air Defense System and the Navy OPCON Center System. However, it appears that economics will soon permit remote multicomputer configurations to be used for scientific computing.

Remote input-output consoles have now reached a state of maturity. There are two basic types. One is the personal console. The user, seated at it, interacts with the computer in a conversational mode. The term *time sharing* has come to be the common descriptor of this mode of operation, although its semantic precision leaves much to be desired. Pioneering work was done at a few installations. The most publicized were Project MAC at MIT, the JOSS System at Rand, and the Time Sharing System at System Development Corporation.

Early commercial application of this technology emphasizes graphical output, at System Technology Laboratories and at General Motors. The first large system to be used like a public utility is the IBM Quicktran System. It shares its consoles not among individuals within the same organization, but on a service bureau basis to anyone in the geographical area.

The second class of remote input-output stations is less glamorous but possibly more useful. These consist of a conventional input device such as a card reader or paper tape reader and an output line printer, all at a remote facility. They are particularly useful when connected to a multiprogrammed computer so that it can simultaneously handle a number of them. A pioneer in this type of operation was the PDP system of Adams Associates. Current installations include the UNIVAC 1004's attached to several computers at the NASA Manned Space Center in Houston, and similar stations, more distantly located from the master computer, and a UNIVAC 1107 at the Computer Sciences Corporation in Los Angeles.

The most complex system to date is the INTIPS system at the United States Air Force Rome Air Development Center. It combines multiprogramming, multicomputing, remote personal consoles, remote graphical input-output devices, and remote conventional I/O devices. It is a lineal descendant of the original RW400 Polymorphic Computer developed by the Ramo-Wooldridge Corporation.

In all of the foregoing examples, communications equipment plays a dominant role. In fact, it may be the key to modern on-line computing. Common carrier lines are in use on a routine basis. Teletype lines are sufficient to service equipment of the typewriter class. More than two thousand bits-per-second can be carried over voice grade telephone lines. For greater bandwith, the common carriers can provide microwave channels in the mega-bit per second class. Unfortunately, the costs of these are proportional to line length, and go up astronomically for microwave over long distances.

Very significant results have been achieved in the last five years in graphic input-output devices. During the 1950's a few commercial scientific installations used the hard-copy equipment then available. For the most part these were mechanical Plotters, very slow and frequently unreliable. You could get your unreliability in the highspeed regime with the IBM 740 Cathode Ray Tube Recorder. In the last five years such equipment has been supplanted by more versatile and reliable devices. The larger, high quality multi-pen XY Plotters are available for those who need them. Many a small installation makes very profitable use of relatively inexpensive digital XY Plotters, such as the CALCOMP. The larger installations, with a greater volume of work, have found the SC 4020 CRT output an indispensable service to the user.

The personal CRT console, with function keys and light pen input is just on the horizon. Pioneered by the Ramo Wooldridge Corporation, we now find them in industrial scientific computing installations such as General Motors Corporation and STL. The announcement of a pair of such devices commercially available as part of the IBM 360 indicates that they must be taken into consideration in facility system design.

RECENT RESULTS—SOFTWARE

From the point of view of the facility manager, the most important part of the facility software design is the executive program. The most recent

trend in executive design, pioneered by the Burroughs B5000, is to give the executive complete control of the machine at all times. Thus, the programmer is not really looking at a machine but at its executive. A significant part of the language used by the programmer, therefore, is a control language actuating certain functions of the executive. The executive is usually multipurpose. It must be designed with a balance between the conflicting requirements of (1) continuous flow or batch processing, and (2) control for a demand processor in case time-sharing consoles should be attached. In addition, it usually has facilities for on-line control—in particular for communications switching.

Under the control of the executive we normally find a number of language processors. Such language processors typically include an assembly language, an algebraic language such as FORTRAN, and a commercial language, usually COBOL. A number of special-purpose processors are usually present, such as a Sort Generator, a Report Generator, or a full-fledged File Management System. Most recently, we find special-purpose, *Third Level*, problem-oriented languages such as APT or SIMSCRIPT. Although the New Programming Language may eventually supplant other algebraic and business languages, and serve as well for real time programming, it would appear that the popularity of these special-purpose, *Third Level*, problem-oriented languages, like APT, will continue to pose a strong requirement that facility design cope with a multiplicity of language processors. Special mention must be made of the newer languages designed to be used at a personal console. At present, these include only algebraic languages for use with typewriter-like consoles, and a few experimental languages designed for graphic input-output consoles. The requirements for console languages will pose a formidable problem for facility designers of the future.

A very significant development in software, and one which must be given serious attention by the facility system designer, is the relatively new concept of *Data Base Management*. We find here the influence of the military system designer, who must plan to control large quantities of information. He has been led to innovations in thinking about how to manage this data base. These ideas are now finding their way into the design of scientific computing facilities.

The first such concept is to treat every piece of data in a standard way under the executive control. Another concept is the realization that programs should be treated as though they were blocks of data. Programs in any language are operated upon by other processors. They can be identified, stored and handled consistently with the data base management philosophy of the particular executive routine. Particularly where large amounts of data are kept in mass storage, the facility designer must give careful thought to the problems of maintenance, updating and purging of this data base.

All foregoing trends have led to another concept —that of the single level store. Operating system design would like to presume that the machine should appear to the programmer *as though it had only primary storage of infinite size.* The management of secondary storage which has plagued programmers for many years should be handled automatically by the total software system. The designers of the software system for the ATLAS Computer were pioneers in attacking this problem. We find the designers of most large modern software systems gingerly probing at the same problem.

EVALUATION OF RECENT RESULTS

As might be expected, there is conflicting evidence about the value of recent developments. However, in a few cases the evidence tends to be fairly definitive. We will examine a few of these.

Continuous Flow Systems

All evidence to data indicates that Continuous Flow Systems are successful. They have achieved their primary design objective of reducing turnaround time—in some cases very dramatically. Some facilities report that average turn-around time has dropped by a factor of three or four—from eight or ten hours down to two or three hours. Though they operate with only a first generation of executive control programs, little complaint is heard that these use too much machine time. Reliability of the mass random access store is reported as excellent.

Remote Consoles

Wherever remote consoles are used, we find the users enthusiastic. However, they always hasten to add that there is much to learn about the design of executives and processors for console languages. Remote input-output stations using standard card reading and printing equipment, although few in number to date, seem to have been quite successful. On the other hand, the various models of personal consoles seem to have been designed with differing

objectives. Where the objective has been to encourage *noncomputing* professionals to use the computer readily and easily, results have been very successful. Otherwise, though technically sound, evidence of their economic value is inconclusive.

Graphic Input-Output

If we consider only hard-copy graphic output, the results have been outstanding. Small installations who have used digital XY Plotters find them almost indispensable. Large installations with massive requirements for graphic data have found cathode ray tube outputs, recorded on either microfilm, photographic paper or both, to be so valuable that many customers have drastically cut down their printed output. The jury is still out, however, on graphic input-output in a personal, interacting manner. What small evidence is available from the General Motors experiment and the STL installation, seems to promise success.

Software

Results are mixed on the new look in software since 1960. Great strides have been made in the acceptance of procedure-oriented languages. Their processors have reached the stage of development where very few people complain about the compilation speed of recent designs. The first few of the new breed of executive monitors, however, received a very hostile reception at their first introduction. But, as the control system designers took their medicine and went back to the drawing boards, improvements have been made. It appears that the modern, general-purpose executive, controlling most of the functions of the installation, is here to stay. However, the attack on the single level store concept must be accounted a failure thus far. No executive or software system to the authors' knowledge has solved this problem in a generally acceptable manner.

FUTURE TRENDS

Based on the rate of development to date, it seems safe to make a few forecasts. First, it is clearly evident that Continuous Flow Systems are bound to dominate the next five years. Few computer users will be willing to put up with off-line preparation of data and slow turn-around time when modern technology makes fast turn-around time possible *at the same cost*. This, in turn, will lead to more and more multiprogramming and multicomputer installations. With modern communication links in their present accelerating stage of development, it certainly appears that remote multicomputer networks will be common by the end of the decade.

This will stimulate a great growth in computer-controlled communication systems. Probably most organizations with more than one major installation will each have all of their computers tied together into a network, unless line lengths make it economically prohibitive. It is desirable to have a computer at each site, to handle communications switching, emergency situations, and for insurance against catastrophes. Thus it seems improbable that central massive computer installations will be very popular.

There seems to be no reason other than economics which will prevent a great proliferation of remote consoles. It is virtually certain that the remote input-output station using conventional equipment will become the most common way of getting work in and out of the Continuous Flow System. At the moment, it is difficult to predict whether remote *personal* consoles can be economically justified to the same extent that technological advances will make them feasible.

It seems inevitable that graphic input-output will continue to grow. Surely, pictorial representations of results are a most desirable thing from the customer's point of view. The technology is here, and the economics are not forbidding. Consequently, there will be great growth in graphic output in the next five years.

It seems that unified, standardized Data Base Management will become commonplace. No difficulties have been encountered, and the advantages are obvious.

Although attacks to date on the problem of a single level logical store have failed, it is our belief that this is because not enough pressure has been brought to bear on the problem. The attacks have been infrequent, and not prosecuted with vigor. The requirements of general-purpose executives and multicomputer, multistore Continuous Flow Systems, together with the great growth in higher level languages, all will bring powerful pressures to bear on this problem. Hence, we forecast that it will be solved in the next five years and all users will only face a primary storage that appears infinite in size.

Finally, what choice will the installation facility system designer have when he selects the languages to be available? The New Programming Language gives every evidence that, in a scientific computing installation, it will serve the purpose of an algebraic

language, a language for on-line applications; and, to a significant extent, a language for the systems programmer. Does this mean it will be the only language available to the users of the installation? Far from it. The paradox is, that the very forces which will make NPL a single, standardized language will, in turn, *permit development of an even wider variety of special-purpose languages.* It will be a poor installation in 1970 which will not support 10 or 20 special-purpose, *Third Level*, problem-oriented languages.

CONCLUSION

In summary, the designer of a general-purpose scientific computing facility today must give serious consideration to the degree to which his facility will incorporate desirable answers to the following questions:

1. Should it be a Continuous Flow System where the input is captured as early as possible and is processed completely automatically, entirely within the hardware, without waiting to be batched with other jobs?
2. Should there be remove input-ouput stations? If so, should these be conventional card readers and printers or personal consoles, or both?
3. Should the installation have multiple computers or provide for their future incorporation? Should some of these be remote from one another? What communications line capacity is needed?
4. How much graphic input-output should be provided? Should there be hard copy or only personal consoles with graphic I/O capability?
5. How sophisticated should the executive control program be? Must it cope with various on-line systems as well as general-purpose scientific jobs? How rigorous should be the procedures of Data Base Management?
6. What languages should be used? To what extent will emphasis be placed on assembly languages, procedural languages, special-purpose problem-oriented languages? Can any or all of them provide the programmer with an apparent single-level storage?

It is the authors' opinion that, in the next few years, good facility design will result from providing positive answers to as many of these questions as the facility's budget will permit.