EDP ANALYZER

AUGUST, 1978
VOL. 16, NO. 8

THE CHALLENGES OF DISTRIBUTED SYSTEMS

With the rapid improvement in micro processor technology and its consequent dramatic reduction in costs, one thing seems clear: small computers will soon penetrate business in a big way. The day of the 'work station processor' is not far off. In some cases, these small computers will be part of multi-level systems designed and installed by the central data processing staffs. In other cases, they will be small, stand-alone computers that will, however, have the capability to communicate with the corporate data center. In either case, these 'distributed systems' pose challenges not only to the data processing function but also to user management and staff. Here are some steps to take to help mitigate the problems.

Ford Motor Company, headquartered in Dearborn, Michigan and with almost one-half million employees world-wide, has been using computer technology since almost the first days of the computer. In addition, the various divisions within the North American operations have had a good deal of autonomy in designing the systems and in selecting the equipment they use. It is not surprising, then, to find that units within Ford have explored in some depth the application of just about every new advance in computer technology. That has been the case, for instance, in the emerging new area of 'distributed systems.'

Some time ago, Mayford Roark, Executive Director of Ford's Systems Office, addressed a group of European data processing executives. His talk dealt with Ford's early experiences with distributed systems; he recently updated the information for us. Roark stressed an overriding philosophy: with distributed systems, as with other advances in computer technology, the question addressed was "how best can we organize available computer resources to accomplish a particular mission?" He then singled out four illustrative experiences within Ford.

The first case, said Roark, concerned the order processing system within the Ford division, which connected 35 district sales offices in North America to division headquarters. In 1968, a decentralized system was in use, with IBM 360/20s installed at each office. But the decentralized system was not meeting management's needs. Management wanted the ability to query the sales order files for details that were not available in the general office's file. Further, the district office installations had to be staffed around the clock, for doing the processing and for communicating with headquarters; costs were thus higher than desired and were growing.

In this instance, said Roark, the division concluded that a centralized system would be a better solution. So, in 1972, the application was

converted to a centralized system, with remote batch terminals located in each district office. Not only could management now query all sales order files but it was also possible to reduce the overall operation to two shifts and with fewer operators, he said.

But the system structure for this application has not stopped there. The Ford division is now in the process of installing new mini computers that will handle working files at each district office, to provide vehicle order status and sales data. The master records will still be stored at the general office. The working files, updated daily from the master files, will be used locally on an interactive basis, for scheduling orders and for analyzing sales trends.

In the second example, at about the same time that the Ford division put in the centralized system, the automotive assembly division saw a need to restructure its material control application, for 20 assembly plants in North America. This division also considered a centralized system but rejected it, the main reason being that the greatest reporting needs were at the plants, not at the general office. In addition, the division had found that if the people at the various plants looked upon the data files as 'our' files, the integrity of the data improved.

So, in this second case, the division chose to locate the master stock status files at the plants and to maintain only a summary file at the general office. The plants would send only those data items needed by the general office, on a daily basis. In addition, the computers at the plants were used for many other applications of a strictly local nature.

For his third instance, Roark cited the experience of the Ford parts division that sells about 200,000 service parts from 20 depots throughout North America. The accounting functions for this division were centralized in Dearborn around 1961 and a centralized real-time inventory control system was installed in 1968.

In 1973, however, this division developed a dealer order entry system, designed to give spare parts availability information to dealers on a fast response basis. For this system, inventory files at the depots were more economical than putting the central inventory file on a fast response basis. So the system that was installed includes a master parts file at Dearborn and depot inventory files at each depot. The depot files are loaded daily from the central file. In structure, the system is quite similar to that of the automotive assembly division, except that the flow of data between the master files and subsidiary files is reversed.

The fourth example concerned Ford's electrical and electronics division. This division has been a leader in the use of mini computers for industrial control applications, said Roark. In addition to their control functions, the minis are being used to provide data input for such applications as attendance reporting, time reporting, production testing, and quality audits. This data flows to a central computer at each plant, which in turn updates the files and processes queries.

In short, said Roark, the various divisions within Ford are making use of a variety of system structures, including both centralized and distributed. Further, the divisions have 'gone in both directions'—that is, from decentralized to centralized, and from centralized to distributed. Each system has been designed to put the computer resources where they would do the most good for accomplishing a particular mission.

Bank of America

The Bank of America, with headquarters in San Francisco, California, is the world's largest bank, measured in terms of assets (over $80 billion). The banks's main operations are in California, where it has some 1,092 branches and two data processing centers—in San Francisco and Los Angeles.

The bank has operations at many other places, however, such as in seven major cities of the U.S. and in 101 countries outside of the U.S. In fact, Bank of America processes data electronically at about 70 sites.

Two characteristics of Bank of America's operations—the huge volume of activity within California plus the large number of processing centers outside of California—have led to two interesting projects in distributed systems. One of these might be called 'clustered mini computers' and the other 'data collection and reporting.'

Clusters of mini computers. Drake, in Reference 1b and at the conference of which this reference is the proceedings, pointed out the reasons behind the idea of clusters of mini computers.

No single computer, regardless of size and speed, can cope with the bank's large volume of transactions in California, he said. Thus, the bank operates many large scale computing systems to provide the needed batch throughput to handle the high volume of daily posting activities.

But there are three main shortcomings of these batch systems that led the bank to evaluate on-line account inquiry systems. One shortcoming is the high cost of producing the printed daily status of customer accounts. Another is the slow process of tellers looking up customer status information or phoning another branch for that information. And the third reason is the exposure to fraud that the batch printouts provide.

An on-line system would have its share of difficulties, though. About 8500 on-line terminals would be required throughout the branches. In order to handle inquiries fast enough to keep customer lines in branches at reasonable levels, a response rate of sixty inquiries per second is required. No existing central DBMS is fast enough to provide that response with a database of 12.5 million loan and deposit accounts. So the solution chosen was a distributed system.

The bank has selected an approach to distributed processing that uses low cost modules of mini computers. The customer loan and deposit account records are assigned to modules. Each branch of the bank communicates with the module that has its data.

For its modular processing capability, the bank has designed a reliable 'processing module' consisting of four mini computers—two for message handling and routing and two for local data base management and application processing. Each of these pairs of minis is loaded to only 50% of its capacity, so that if one mini goes out, the other mini can carry the load.

By way of the software, a module is specialized to handle one type of application or a small number of very similar applications. The goal here was to keep the programs simple and the processing fast.

Communication standards have been set up for communicating between modules.

Currently, five of these modules have been installed in San Francisco, for serving branches in Northern California, and five have been installed in Los Angeles for serving Southern California.

These clusters of mini computers meet the requirement for high reliability and high transaction rate for this transaction oriented on-line system. However, they do not provide for other reporting requirements. The reporting system remains a batch type operation, with its voluminous output and slow process of program development and maintenance.

Data collection and reporting. With the many processors worldwide, the bank's management has been faced with the problem: how to provide the concurrent consolidation of data from more than one processing center and from more than one application? While the various processors are essentially free standing, management needs a means to both get meaningful consolidated reports from the processors and to be able to probe downward for detailed data as required.

The elements of the bank's proposed solution to this problem is as follows. *Local database processing*, at the local machine level, must provide access to data in a manner appropriate to the primary use of the equipment. The access method may be relatively simple in nature, such as indexed sequential, or more complex, with numerous access paths and data relationships. Next comes a *global datamanager*, a hardware and software combination that logically integrates the part of the network for which it is responsible. This global datamanager will contain a global directory of data and a global query language processor. It will serve one or more processing sites. The greater the capability of the local database processor, the more functions the global datamanager will delegate to it. Finally, a *user interface* will provide a natural language capability and will translate human queries into the global query language.

In operation, says Drake, a user will enter a query for information via the user interface facility. The query will be translated into the global query language and sent to the nearest global datamanager. This global datamanager, in turn, will consult its global data directory and will direct a series of messages to other global datamanagers throughout the network, asking for the desired data. The latter global datamanagers will direct the retrieval of the desired data (for those local systems employing only rudimentary access methods) or will turn the retrieval tasks over to

the local DBMS. It will then send the requested data back to the originating global datamanager for delivery to the user.

One of the main potential cost items in moving to a distributed system is the need to totally revise existing application programs. The bank is seeking to avoid this cost—by way of the global datamanagers. The global datamanagers allow communication with existing application systems, ranging from simple transaction systems to complex query systems. They also enhance security because all data flows between sites must go through them and hence be subject to control

General Electric Company

Slone (Reference 1a) has described one approach that his company has tried, in the use of distributed processing.

When manufacturing companies have used a combination of maxi and mini computers in the past, said Slone, there tended to be not much interaction between them. Then gradually, data began to flow between them, perhaps by way of magnetic tape. Today, the maxis and minis are still largely dedicated to their own tasks, and the links between them remain quite casual.

Further, he says, the minis of the past have not been very convenient for application system development. The available software has been quite limited, forcing users to develop very application-dependent software (and, to make matters worse, often in assembly languages). So the application development costs for the minis have been relatively high, in his opinion.

What is needed for effective distributed processing, he says, is a way of using both the maxis and the minis in a hierarchical structure and in a manner the gives the benefits and strengths of each. To accomplish this, he says the following elements are needed: (a) high level programming languages to be used for an on-line environment; (b) effective data base management for both the maxis and minis; (c) data base integrity and recovery facilities for a multi-access environment; (d) integrated transaction processing software; and (e) easy-to-use links to the maxi computers.

To demonstrate the feasibility of this approach, his department has developed a generalized software system (that they call MADTRAN) to run on a hierarchy consisting of a Honeywell H6080 as the maxi and DEC DECsystem-10s as the minis. A pilot application system, involving job time data handling, has been installed on the system.

The minis are essentially free-standing systems that perform transaction processing and the updating of their local data bases. (They can also be used for real-time control of factory equipment, but this feature was not included in the pilot system.)

In the pilot application that has been implemented, job data is transmitted from the maxi to the appropriate minis. This data includes standard production times for the jobs. During the day, actual job data is entered on the minis, showing actual production accomplished and hours expended. At the end of a shift, the mini produces performance reports for the factory foremen. Also, records pertaining to completed jobs are stripped from the local file and sent to the maxi, where labor distribution, payroll, planning reports, and so on, are produced.

To ease the task of creating application software, Slone's group has developed an 'edit and prompt' package. The developer(s) of an application system need only supply descriptions of message formats, editing rules, and prompts for missing fields. The package generates edited, fixed format messages for the application programs.

Another point of interest is that, by means of an input code, the terminal operator can indicate whether he/she wants to be prompted for each input field (for novices) or whether a prompt can call for multiple fields.

Further, the mini may have to call on the maxi for processing or data that is not available at the mini. The transaction code itself might determine that such service is needed, or the terminal operator may signal that connection to the maxi is desired.

We checked to see about more recent information on this system, but no further information is being released at this time.

The push for distributed systems

In the instances discussed above, the pressure to install distributed systems came from two main sources. In some of the cases, the initiative

came from the central data processing management and staff, because they could see the problems of trying to handle a growing workload on large, central computers. In other instances, the initiative came from user department management who sought some benefits that they felt centralized data processing could not offer or offer as economically.

We suspect, from the comments that we have heard on this subject, that most frequently the push to install a distributed system comes from user department management, the end users. The main reasons seem to be (1) dissatisfaction with centralized data processing and (2) seeking benefits not easily available from centralized data processing.

Dissatisfactions. An Infotech report (Reference 2) has caught many of the complaints that end users have voiced about centralized data processing.

Very briefly, these usually hinge upon the belief that centralized data processing is a near monopoly with a captive audience. Centralized data processing is motivated by things other than user happiness, say the critics. Data processing has its own set of priorities as to which jobs to run when time schedules get tight. Also, user management may have little or no control over their data processing expenditures, and they do not know how much they are really getting for their money.

Another complaint is that, with centralized data processing, the risk of loss is greater than it would be with distributed processing. Fires, floods, or disgruntled programmers may destroy the data files and programs. When the central system goes down, all on-line users cease work. Yes, there are practices that can be followed to help mitigate such problems, but those practices may not always be followed. The risk does exist.

Seeking benefits. Typically, the end users prefer to be responsible for their own activities—that is, to run their own system, under their own sets of priorities, subject to their own budget, and using their 'own' data files. End users typically feel, we believe, that they can more easily obtain these benefits with their own computers than they can from sharing a central system.

Another type of benefit comes from the fact that most of today's mini and micro systems provide interactive operation with immediate access to files. Not only does this provide faster turnaround but it also tends to catch errors of input right on the spot.

A *possible* benefit of distributed systems is that of cost reduction. We will have more to say about this point below.

A variety of approaches

While it is commonly recognized, we believe, that the term 'distributed system' is ambiguous, it is worth reviewing briefly some of the different meanings given to the term.

Distributed data entry is a function to which the name 'distributed system' is frequently given. Key-to-tape, key-to-disk, key-to-diskette, and key-to-cassette data entry devices are often marketed under this term.

Distributed processing, central data storage involves the use of intelligent terminals at the remote points, connected through a network to a central processor and data files. Hierarchical distributed systems often have this structure.

Distributed processing, local data storage implies that the local sites have both processing and data file storage capabilities. To make them part of a distributed system, they should have some means of co-operating with each other, perhaps by communicating over a data communications network. This form of distributed systems implies the concept of a *distributed data base*. While the true distributed data base is probably beyond today's state of the art, it is receiving much attention. We will say more about this work later in the report.

Small, stand-alone systems are being marketed today under the name of distributed systems. Under this approach, each department might have its own mini or micro computer based system—but there is no assurance that the various departmental computers will be able to communicate with each other or with a headquarters computer.

As we will point out later, it seems to be in this area of the small, stand-alone departmental computer that most of the concern about distributed systems is concentrated.

Work station systems are in the offing, as we discussed two months ago. They will act as personal terminals and will replace the typewriter,

the hand calculator, and so on. They probably will be marketed as distributed systems.

As yet, slow progress

Even with this variety of approaches, supported by today's technology, one must concede that the acceptance of distributed systems to date has been quite slow. Perhaps distributed data entry is the variety that has received the most acceptance, but even here the growth has been steady rather than dramatic, we gather.

We have observed that a strong, central data processing staff tends to resist the pressure for putting processing power and (particularly) data storage at the locations of the end users. Instead, these staffs appear to argue for the following approach, which has been reasonably successful.

First, centralize data processing, say these people. A lot of organizations still have numerous, perhaps somewhat small data processing installations—a holdover from the '1401 days.' A good amount of program and data incompatibility still exists. So the best step to take in such instances, say these centralists, is to first get things under control. Remove the incompatibilities by putting all applications on a central system. This will make a future conversion to a distributed system much easier.

Second, off-load some data entry functions in order to distribute this workload and to reduce errors of input. If the people who understand the data are the ones who perform data entry, data integrity will improve. This off-loading may be accomplished either via an on-line system (such as a time sharing system) or via key-to-disk or other such devices. But the processing programs and data files remain at the central site.

Third, install intelligent terminals (which might replace the data entry terminals) that have local data storage capability. The main data files would still be centrally maintained, but local sub-files would be used for local processing and handling inquiries. Also, the programming of these terminals might be controlled centrally, perhaps by not providing a local programming capability (as with the IBM 3790 terminals).

Two points stand out in this approach. One, the organization may still be in the process of cleaning up the 'mess' resulting from the proliferation of computer usage in the 1960s. Secondly, an almost total rework of existing application systems may be needed—whether centralizing or distributing. Data processing management can often make a strong point against moving to distributed systems on the basis of such points.

Gathering speed

Maybe progress to date has been slow, but we expect that interest in distributed systems will accelerate rapidly in the next year or two.

We see four main reasons for this likely speed-up. One is the rapid growth in the micro computer technology. Another is the emergence of the public packet data networks. The third is the economics that seem to be favoring distributed systems. And the fourth is an aggressive new attitude toward the development of standards that will support distributed systems.

Micro computer technology. We discussed what is happening in micro computer technology two months ago and so will treat the subject only briefly here. Much more powerful hardware and software are becoming available under this technology. The micro processors are now competing with the mini computers and soon will be competing with regular CPUs. Software development may well proceed almost as fast. Since micros may not be shared among jobs but instead may be dedicated to functions or jobs, the software may be less complex.

What this means for distributed systems, we believe, is the following. User department management will be able to say, "There is no need to revise existing application programs. Just let us extract the data we need from the existing central files and store it locally, where we will have immediate access to it. With our own system, we will also be able to handle local jobs that central data processing does not want."

This will be one strong argument for distributed systems, we think.

Public packet networks. This subject also was discussed in depth, last month, so we will treat it briefly.

Public packet switched networks are becoming available in many countries, for serving many cities and towns in those countries. Further, they are providing interfaces for the most popular

types of computers and terminals. These networks are providing quite efficient data communications, at a relatively low cost, particularly for low volume usage.

As far as distributed systems are concerned, this means that departmental computers can more easily communicate with each other and with central data processing. The argument against distributed systems "because our network won't support them" is becoming less and less valid.

Economics of distributed systems

One of the main arguments for centralized systems has been that of 'economy of scale.' But that argument is being attacked from several sides.

Reynolds (Reference 6) points out that large, centralized installations are losing ground economically in three ways—operating staff costs, development staff costs, and hardware costs. It is true in a batch environment, he says, that centralization has had economic benefits. But with the greater use of on-line systems, operator costs are becoming less significant. Also, in large shops, specialized, rigid operator functions tend to develop; in small shops, one person normally does many functions.

As far as development staff costs are concerned, says Reynolds, the expected savings from 'common applications' that are centrally developed often are a mirage. Accounting systems, run in a batch environment, can and do show savings from centralized development. But this has not proved to be so true of engineering and manufacturing applications, which seem to vary from factory to factory. In fact, batch systems do not serve these applications particulary well.

Economies of scale in hardware have just about disappeared, says Reynolds. He presents some comparative figures from studies that his staff has made. These figures show that some mini computers provide the same processing power as leading maxi computers (such as the IBM 370/155) but at a fraction of the cost. And some micros are showing even greater performance-cost ratios.

Reynolds goes on to discuss a number of other non-economic factors that seem to be favoring the dispersion of computing power and data storage.

Louis M. Branscomb is chief scientist and vice president of IBM. Three months ago, he was the keynote speaker at a conference of which Reference 1c is the proceedings (but the speech is not in the proceedings). In his talk, he reviewed some studies, by IBM and others, of what is happening to data processing and data communications costs. In 1975, said Branscomb, one study showed that the average DP budget was: communications hardware and services, 19%; computer hardware and purchased software, 46%; staff salaries, 35%. Further, studies indicated that the communications costs are falling at an average rate of 11% a year, computer hardware costs are coming down at about 15% a year, and staff costs are *rising* at about 6% a year. If one extrapolates the 1975 figures out to 1983 using these trends, he said, one finds that communications costs will be down to only 10% of the total, computer hardware will be down to 17%, and staff costs will be up to 73%.

There is another factor to consider—the steady growth in total data processing expenditures. This growth averages between 12% and 15% per year. In 1975, the total U.S. expenditures were estimated at $26 billion. By 1983, an estimated $75 billion per year will be spent.

How will this $75 billion be spent? If the above-mentioned 1983 percentage estimates (10%, 17%, and 73%) are used, on the assumption that the 1983 environment would be like the 1975 environment, that would mean some $55 billion for staff alone (73% of $75 billion). But this extrapolation is not valid, said Branscomb. Staff costs will be held down by transferring more of the load to computers and communications. In other words, distributed systems will be used to hold down staff costs. His company thinks the percentages for 1983 will be more like 39% for communications, 37% for computers, and 24% for staff.

So it is just possible that distributed systems will installed as a means for controlling the growth of DP staff costs.

Standards developments

We cite just three examples of some interesting work going on in the standards area. In each instance, something new has been injected into standards work—a real sense of urgency.

West Coast Computer Faire 1978. We discussed the 1978 Computer Faire two months ago, as an illustration of the interest in personal computers. But there was also some noteworthy standards work reported there.

One of the technical sessions we attended, partially covered in Reference 3, dealt with hardware and software standards. The hardware standards discussion dealt with the S-100 bus, originally developed by Intel and becoming almost an industry standard. In theory, any micro processor or peripheral unit that is S-100 compatible can be just 'plugged into' an S-100 machine. However, the original timing protocols were deficient to the point where, as one speaker said, "S-100 compatibility means that the connector has 100 pins." The end users were complaining that the units they were buying would not work as advertised. And the computer stores were upset because their customers were upset.

An IEEE Computer Society committee has been working on an S-100 standard; committee members reported on the work. The session was essentially a discussion of the proposed standard—and members of the audience participated in the discussion. Committee members felt that the new 'standard' bus would turn out to be much better than any individual computer manufacturer now offers.

We asked about when the results of this work might be of benefit to end users. Possibly as soon as later this year, we were told. No, the standard might not be officially adopted by the IEEE by that time. But as soon as the standard is crystallized, many of the micro processor and peripheral manufacturers probably will start using it, because they also are tired of the complaints about non-compatibility.

(We have heard some experts comment that the S-100 basically is not a good design and is really not suitable for standardization. However, it is very widely used, good design or not, and is apparently becoming an ad hoc standard.)

University of California at San Diego. As we discussed two months ago, UCSD has been a leader in the use of both micro computers and the PASCAL programming language for teaching computer science courses. Moreover, they have developed a relatively portable software system which allows both application programs and system software to be moved from one type of micro (or mini) computer to another. Their software is being adopted by a number of micro computer suppliers, possibly to supplant BASIC as the main programming language for the micros.

PASCAL has many of the data naming advantages of COBOL plus the control structures desired for structured programming. Also, it is suitable for interactive program development.

Kenneth Bowles, at UCSD, has recognized that PASCAL has some shortcomings that must be corrected before it can have widespread use. In an attempt to prevent each supplier 'doing his own thing' in making such enhancements (as has happened with BASIC), Bowles decided to take some action.

He organized a two-week working conference that was held last month at UCSD. All interested PASCAL suppliers were invited to attend. The first week was spent identifying the needed enhancements and coming to a tentative agreement on them. Then the attendees went home for a week, to talk over these ideas with their colleagues. They then reassembled for the second week of the conference, to make any revisions in their earlier ideas and to develop more detailed specifications for the enhancements.

It is possible that a relatively 'standard' PASCAL will emerge from this effort and that it will gradually become the most widely used programming language for the micros.

Bowles also argues that users should start demanding 'manufacturer independent' and 'machine independent' software for micro computers. "Don't get locked into one machine family or one manufacturer," he says. Toward this end, Bowles and his associates at UCSD have developed a PASCAL pseudo-machine interpreter. The PASCAL source code is first *compiled* into PASCAL object code. The interpreter then makes the host machine look like a PASCAL machine. The object code runs about five times slower than native code and the interpreter takes about 8K of memory. But if the operating system and other system software are also written in PASCAL, then *everything*—application programs, system software, etc.—can be moved to a different micro just by rewriting the interpreter. In fact, they have done just that at UCSD. For more information on this work, see Reference 7.

Study Group on Distributed Systems. Last month, we briefly mentioned the work of the Study Group on Distributed Systems, under the American National Standards Institute committee on systems planning and requirements. The work of this study group, we believe, will add momentum to the acceptance of distributed systems.

As we understand it, the charter of this study group is to look at the area of distributed systems, create a logical structure for the area, see where standards are needed, see where standards are currently being developed, and then recommend an organization (and reorganization) of the standards work. Further, the study group is expected to move as rapidly as possible; distributed systems are emerging so fast that the normal (slow) standards process is not acceptable.

The chairman of the study group is Charles W. Bachman of Honeywell Information Systems. Other companies represented on the study group include IBM, Burroughs, Digital Equipment, Control Data, Univac, Bell System, and Xerox. The U.S. government has several representatives, and three other organizations are represented.

The study group has developed a logical structure for distributed systems, as requested. The group sees these systems as consisting of networks of co-operating work stations. A work station is a cluster of activities created to do some portion of the work of an enterprise. The work stations co-operate by exchanging information. The study group has been defining several levels of protocols to allow this co-operation among work stations. Some of these protocols may be the subject of international standards efforts. And some of these standards might begin to appear in the next two years or so.

For more information on the work of this study group, see Reference 4.

We view these four areas—micro computer technology, public packet networks, cost trends, and aggressive standards efforts—as supporting a distributed system capability.

Yes, the conditions seem right for much faster acceptance of distributed systems.

But there are some very real problems—problems that could turn 'distributed systems' into 'distributed messes.' Let us take a look at those potential problems and what might be done to mitigate them.

THE CHALLENGES

We cannot remember any major technical advance in the computer field that has been adopted by users without 'pain and suffering' at the outset.

There is good reason to expect, then, that distributed systems will cause their share of pain and suffering. Let us take a look at where the problems probably will lie.

We will concentrate on the situation where user department management wants to install a departmental mini or micro computer system, in order to perform the department's own work. This is the situation that is encouraged by the developments just discussed.

Management concerns

As we indicated earlier in this report, strong data processing executives have, in many cases, resisted the installation of mini and micro systems in end user departments. In the face of strong pressures from user department management, they have been able to convince executive management on the validity of their concerns. Here are some of the concerns that we have heard discussed.

Costs. A departmental mini or micro computer installation might start out as a small expenditure, but costs can then gradually get out of control. The application programs turn out to cost much more than the hardware. Maintenance and enhancement costs turn out to be much higher than expected. Computer staff costs, for programming and operations, begin to appear where none were really expected by user department management. The times and costs for developing new applications turn out to be greater than budgeted.

All of these things have been pervasive in the past. Why not expect them to happen with distributed systems? True, with relatively inexpensive micro systems, it will be feasible to divide big applications into small sub-systems. By reducing the complexity, some of the problems of the past may be avoided. But it would be unrealistic to expect that none of these cost problems will occur with distributed systems.

Quality. The low cost of the hardware may lead user department management to expect low

cost software. Why should one have to pay two, five, or even ten times as much for the software as one pays for the hardware? This line of thinking can (and does) lead user department management to select the lowest fixed price bid for the development of the application software. Since the requirements are essentially *never* stated completely and accurately, this leads to troubles.

What are the 'troubles'? Again, they are the well-known difficulties that most installations have experienced. The development project begins to slip behind schedule and over cost. Corners are cut. Programs are quickly (and poorly) designed. Documentation is sacrificed. Everything is done in a 'quick and dirty' fashion in order to stay within the tight budget. Little or no use is made of corporate standards, with a loss of compatibility between the departmental system and the corporate system, in terms of programs and data. As multiple departments take this same approach, there results a wide variety of computer types, operating system types, data storage media types, programming languages, and so on.

Reliability. When a departmental mini or micro system is brought in, department management has one or a few main applications in mind for it. All attention is directed at getting that or those applications running.

But there are other things in addition to the application programs that must be considered, if the departmental computer is to operate reliably. These include the need for backup facilities, restart and recovery facilities, data security, keeping redundant data files in synchronism, and so on. We suspect that, typically, no budget will be provided for these things if the departments are left to their own devices.

Just because the departmental computers are small is no reason to believe that such needs can be ignored.

Availability of data. As indicated earlier in this report, executive management is coming to expect the ability to probe downward into data files. If some cost figure is substantially higher than expected, for instance, management probably will want to be able to find out specifically what items have caused the excessive cost.

If each department has essentially gone its own way on installing a computer system, there is no assurance that top management will be able to probe the departmental files. These files will thus be unavailable to higher levels of management.

These, then, are the main concerns that we have heard expressed, from a management standpoint, about distributed systems. As some people have said, distributed systems (of the free-standing departmental system variety) represent "the 1401 days all over again."

But, in addition to the managerial concerns, there are some technical concerns to be considered.

Technical concerns

Yes, the technology in support of distributed systems is progressing very rapidly. Yes, the designers of the micro processors can design in all of the features they would like to see in those processors. Yes, work is going on in standards that will help provide compatibility for all of the components of distributed systems. But No, all of the elements of a distributed system have not been completed yet.

Let us take a brief look at some of the as-yet missing elements.

Standards. Standards for distributed systems not only have to be developed, agreed upon, and adopted, they also must be implemented. And there lies a big problem. How well will they be implemented?

Consider, for instance, the EIA RS-232 signal connector interface standard, used for connecting terminals to modems, etc. We have talked to people who have tried to connect units from several different manufacturers. Often as not, they say, something doesn't match. As the number of different suppliers increases, the more differences that come to light. To some, this interface standard "only assures that the connectors will mate, not that the electrical signals will match up properly."

If such a problem exists with something apparently as standardized as the RS-232 interface, think what lies ahead as more complex standards are adopted for distributed systems.

Evolving micros. The very rapid change in micro computer technology is well known. Because this rate of change is expected to continue,

the question arises: how can a company keep from getting locked in to 'old' micro technology?

For instance, on the hardware side, the 8-bit word length micros are common today. The 16-bit micros are beginning to appear (DEC LSI-11 and Intel 8086). In two years, it has been predicted that 32-bit word length micros will appear. (There is some controversy on the need for such word lengths in micros; they often will be used in dedicated fashion, not multi-programmed, and will thus not need the huge memories implied by 32-bit word lengths.) Also, peripherals are continually improving in performance and lowered costs.

Much the same situation prevails in software. Today, BASIC is the most common higher level language for the micros, but it has a lot of shortcomings. PASCAL may replace it as the common language for the micros. Improvements are being made in operating systems, DBMS, communication monitors, etc. for the micros.

So the micros that are installed in the next one or two years might well become technically obsolete fairly quickly. How can distributed systems be designed so that they can evolve with the technology, and not get held back by old hardware and software? This is a challenge—and Bowles solution, mentioned above, is one approach to meeting that challenge.

And then there is the question of distributed data bases.

Distributed data bases

This brief discussion draws on Liebowitz and Carson (Reference 1d), the work of the CODASYL Systems Committee (Reference 5a) and Bachman's commentary on this CODASYL report (Reference 5b). Readers interested in this subject should see these references.

The CODASYL Systems Committee has studied and described the new environment where multiple data bases reside at the nodes of a distributed system. Their report has been published in the hope that others will study it and comment on it.

Bachman, in his commentary, points out that essentially all of today's systems are really distributed ones. There are essentially no truly centralized systems, where just one computer and one data base serve the needs of the whole company (certainly true for larger size companies). There are almost always multiple computers and multiple data bases. So distributed systems, as the term is now being used, really involve connecting those components that are already distributed.

The CODASYL report sees a distributed data base node as consisting of (a) a network access process for interfacing the node with the communications facility, (b) a network DBMS function that includes the more conventional DBMS function, (c) a local data base, and (d) user processes. The committee has presented design alternatives for structuring such systems, as well as technical and administrative issues related to their use.

The committee is asking: "Is this the way these systems are likely to look? Can you suggest more likely alternatives? If we can agree on the likely structure, this will help guide development."

Bachman adds that a mechanism is needed that (a) allows messages to flow between the nodes of the system (between the co-operating work stations), (b) provides restart and recovery facilities, as well as concurrency control, and (c) has a means for reaching agreement that a job has been completed correctly.

Liebowitz and Carson address the question of finding data in a distributed data base system. Each node can have its own directory, which can be for local data only or for the complete system (highly redundant). There can be a central directory, which can be the only directory or which acts in concert with local directories. And there might be regional directories. With redundant directories, the main problems are storage costs, duplicate updating, and keeping the directories in synchronism. If redundant directories are not used, the network will be cluttered with "I need record X, do you have it?" messages.

Perhaps this brief description has pointed out some of the technical challenges associated with distributed systems.

Distributed systems: what to do?

The challenges and the problems of distributed systems are real. So how can an organization go about coping with this new technology?

The answer depends on which approach to distributed systems is taken.

Centrally designed hierarchical system. As far as we can tell, the 'successful' distributed systems

that have been installed to date have been designed and implemented by the central data processing staff. This is the approach that IBM, for one, seems to be currently advocating.

The only advice we have come across for this approach is almost 'old hat.' It is much the same as we presented three months ago in connection with DBMS conversions. The same advice applies for using any new, complex technology for the first time. Take it easy for the first application of a distributed system. Pick a non-trivial but also a non-vital application for the first efforts. Have realistic expectations; a distributed system will be more difficult to install and will cost more than the first estimates generally indicate. Treat the distributed system project like any other system project, using good project management methods. And start immediately to take steps to make future conversions to new technology easier.

Free-standing department systems. This is perhaps the most challenging situation of all for data processing management because it is so hard to control. The end users, in this case, are trying to break their dependence on central data processing and might well resist any effort by the central staff to help them.

Nevertheless, we suspect that most end users will need such help. Here is what we see as needed.

Consulting and training. Earlier in this report, we discussed a number of management concerns about distributed systems—involving costs, quality, reliability, and availability of data. These are the areas where the user departments need help. Central data processing might develop guidelines of good practices, for use by other departments for installing their own computers. And short training courses might also be in order, to be given to department representatives.

Get running on time sharing. This is an excellent approach that user departments might be encouraged to take. If the company has an appropriate in-house time sharing network, perhaps it can be used. If not, consider using a commercial time sharing service to get the application(s) developed and running. Then, when running satisfactorily, bring the job in-house on a departmental computer. There is a better chance with this approach that corporate standards can be followed when developing the application systems. Also, a much better hardware and system software selection might be made under this approach, as opposed to making the selection before the application is developed.

The pressures for distributed systems will be too powerful to resist, we believe. But the challenges posed by these systems must be overcome, or a real 'mess' is sure to result.

REFERENCES

1. Digests of papers presented at IEEE Computer Society conferences and seminars; order from IEEE Computer Society, 5855 Naples Plaza, Suite 301, Long Beach, Calif. 90803:
   a) *Compcon Fall 1977*, price $20.
   b) *Compcon Spring 1978*, price $20.
   c) *Trends and applications: distributed processing*, price $12.
   d) *Tutorial on distributed processing*, 1977, by B. H. Liebowitz and J. H. Carson.
2. *Distributed Processing*, Infotech State of the Art Report. Order from Infotech International Ltd. (Nicholson House, Maidenhead, Berkshire SL6 1LD, U.K.); in the U.S. order from Auerbach Publishers Inc. (6560 North Park Drive, Pennsauken, N.J. 08109); write for price.
3. *Conference Proceedings*, Second West Coast Computer Faire, 1978. Order from Computer Faire (P. O. Box 1579, Palo Alto, Calif. 94302), price $15.
4. For information on the work of the ANSI Study Group on Distributed Systems, write Charles W. Bachman, MS 897A, Honeywell Information Systems, Inc. 300 Concord Road, Billerica, Massachusetts 01821.
5. *Proceedings of 1978 National Computer Conference*, order from AFIPS Press (210 Summit Avenue, Montvale, N.J. 07645); price $60:
   a) "Distributed data base technology—An interim report of the CODASYL systems committee," p. 909-917.
   b) Bachman, C. W. "Commentary on the CODASYL systems committee's interim report on distributed database technology," p. 919-921.
6. Reynolds, C. H., "Issues in centralization," *Datamation* (1801 S. La Cienega Blvd., Los Angeles, Calif. 90035); March 1977; p. 91ff.
7. For more information on the UCSD work, write Institute for Information Systems, Mail Code C-021, University of California at San Diego, La Jolla, Calif. 92093.

Additional Reading

8. *Computer,* IEEE Computer Society (address above), price $6 per issue.
   a) Peebles, R. and E. Manning, "System architecture for distributed data management," January 1978, p. 40-47.
   b) Maryanski, F. J. "A survey of developments in distributed data base management systems," February 1978, p. 28-38.
   c) Morrow, G. and H. Fullmer, "Proposed standard for the S-100 bus," May 1978, p. 84-90.
9. Bernstein, P. A., J. B. Rothnie Jr., N. Goodman, and C. A. Papadimitriou, "The concurrency control mechanism of SDD-1: A system for distributed databases," *Software Engineering* (IEEE Computer Society, address above); May 1978, p. 154-168, price $10.
10. Faggin, F., "How VLSI impacts computer architecture," *Spectrum* (IEEE, 345 E. 47th Street, New York, N.Y. 10017); May 1978, p. 28-31; price $4.
11. There are a number of other papers on distributed systems, in addition to those listed in Reference 5 above, in the *Proceedings of 1978 NCC.* Also, cassette recordings of all sessions (including session Th100, "Installing distributed systems,") can be obtained from On-the-Spot Duplicators, Inc., 7309 Fort Hunt Road, Alexandria, VA 22307; price $5.50 plus billing and shipping charges.

The automated office field is just beginning to emerge. Some conferences have been held on this subject, a growing number of firms are investigating it, and a barrage of new products is reaching the field. We have been following developments quite closely for over a year. In the next two reports, we will present our findings, to provide 'early warning' for data processing management about a significant new trend.

SUBJECTS COVERED BY EDP ANALYZER IN PRIOR YEARS

1975 (Volume 13)

Number

1. Progress Toward International Data Networks
2. Soon: Public Packet Switched Networks
3. The Internal Auditor and the Computer
4. Improvements in Man/Machine Interfacing
5. "Are We Doing the Right Things?"
6. "Are We Doing Things Right?"
7. "Do We Have the Right Resources?"
8. The Benefits of Standard Practices
9. Progress Toward Easier Programming
10. The New Interactive Search Systems
11. The Debate on Information Privacy: Part 1
12. The Debate on Information Privacy: Part 2

1976 (Volume 14)

Number

1. Planning for Multi-national Data Processing
2. Staff Training on the Multi-national Scene
3. Professionalism: Coming or Not?
4. Integrity and Security of Personal Data
5. APL and Decision Support Systems
6. Distributed Data Systems
7. Network Structures for Distributed Systems
8. Bringing Women into Computing Management
9. Project Management Systems
10. Distributed Systems and the End User
11. Recovery in Data Base Systems
12. Toward the Better Management of Data

1977 (Volume 15)

Number

1. The Arrival of Common Systems
2. Word Processing: Part 1
3. Word Processing: Part 2
4. Computer Message Systems
5. Computer Services for Small Sites
6. The Importance of EDP Audit and Control
7. Getting the Requirements Right
8. Managing Staff Retention and Turnover
9. Making Use of Remote Computing Services
10. The Impact of Corporate EFT
11. Using Some New Programming Techniques
12. Progress in Project Management

1978 (Volume 16)

Number

1. Installing a Data Dictionary
2. Progress in Software Engineering: Part 1
3. Progress in Software Engineering: Part 2
4. The Debate on Trans-border Data Flows
5. Planning for DBMS Conversions
6. "Personal" Computers in Business
7. Planning to Use Public Packet Networks
8. The Challenges of Distributed Systems

(List of subjects prior to 1975 sent upon request)

PRICE SCHEDULE

The annual subscription price for EDP ANALYZER is $48. The two year price is $88 and the three year price is $120; postpaid surface delivery to the U.S., Canada, and Mexico. (Optional air mail delivery to Canada and Mexico available at extra cost.)

Subscriptions to other countries are: One year $60, two years, $112, and three years $156. These prices include AIR MAIL postage. All prices in U.S. dollars.

Attractive binders for holding 12 issues of EDP ANALYZER are available at $6.25. Californians please add 38¢ sales tax.

Because of the continuing demand for back issues, all previous reports are available. Price: $6 each (for U.S., Canada, and Mexico), and $7 elsewhere; includes air mail postage.

Reduced rates are in effect for multiple subscriptions and for multiple copies of back issues. Please write for rates.

Subscription agency orders limited to single copy, one-, two-, and three-year subscriptions only.

Send your order and check to:
EDP ANALYZER
Subscription Office
925 Anza Avenue
Vista, California 92083
Phone: (714) 724-3233

Send editorial correspondence to:
EDP ANALYZER
Editorial Office
925 Anza Avenue
Vista, California 92083
Phone: (714) 724-5900

Name_____

Company _____

Address _____

City, State, ZIP Code_____