EDP ANALYZER
Editor and Publisher: RICHARD G. CANNING

EDP ANALYZER

JUNE, 1981
VOL. 19, NO. 6

SUPPORTING END USER PROGRAMMING

With the continuing scarcity of data processing professionals that is creating numerous staffing problems, the thought of end users 'programming' some of their own applications can be tantalizing to data processing management—and, at the same time, ominous. Interestingly, we found some large and small companies that are not just supporting this idea but encouraging it. They are receiving benefits much broader than even they anticipated. They started out with a trial effort, and now a number of their non-data processing employees do some programming as part of their daily work. Here is what we learned from these companies.

IBM Canada Ltd. has headquarters in Don Mills, Canada, a suburb of Toronto. They employ some 11,000 people in manufacturing plants, a laboratory, and sales and service offices across Canada. Current annual sales are about $1.5 billion.

In 1974, the IBM Canada information systems division decided to experiment with the concept of encouraging end users to perform some of their own programming. At the time the information systems division was not able to respond to user requests as rapidly as they would like—partly because they were spending 70% of their time maintaining existing programs.

So in 1974 the division formed the Information Centre, with four analysts and one manager, plus two software products—GIS for queries and APL for data handling. This small group was placed close to users in the main headquarters building. Their express purpose was to teach non-data processing employees how to use either GIS or APL to access the computer for their own needs. Using GIS, users could, on their own, query databases created and maintained by the information systems division. GIS queries could be made on-line, but batch requests were sufficient for most users.

Although it took awhile to grow, the information centre now serves over 800 users, whose usage accounts for 25% of the total computer usage at IBM Canada. The staff has grown to eleven analysts who assist these users.

ISSN 0012-7523. Multiple copy prices listed on last page. Photocopying this report for personal use is permitted, providing payment of $2.00 fee *per copy of report* is made to Copyright Clearance Center, Inc., 21 Congress Street, Salem, MA 01970; please include identifying fee code 0012-7523/81/060001-12$2.00.

The centre now offers interactive products in text processing, financial planning, and education as well as query and APL-based packages. In addition, it provides a number of services, including education, user assistance, planning, product evaluation, and consulting, which management feels are basic to successful end user programming.

Education. Each new user is trained on the software products he or she expects to use. The introductory GIS course, for example, lasts two and one-half days and is aimed at people who have no data processing knowledge. During this course, each attendee creates at least one program to solve a real-life problem he faces in his job. As the centre adds new products, at least one staff member becomes knowledgeable on that product, so that he or she can lead training courses and provide user assistance. Each member of the staff specializes in a couple of products, and users direct their questions to those experts.

User assistance. Since the major purpose of the centre is to get other employees to use the computer on their own, the staff does not do any programming themselves. But they do a lot of 'handholding'—sitting beside users as they perform new types of queries or programming—and troubleshooting—helping these users when they run into difficulties using the computer. And they encourage experimentation. The centre's staff tends to be more people-oriented, rather than technically oriented.

Planning. The centre acts as the broker between the computer center, which provides the raw computing resources, and the end users. Annually, the centre's staff informs computer center management how much of each computing resource the users will need for the next two years. Although the accuracy of estimated usage varies by department, the estimated total is very close to the actual usage.

Product evaluation. The centre makes a formal evaluation of new products, as they are announced, and adopts those which fill user needs.

Consulting. The centre provides advice, as needed, to users in the design of new applications, to minimize both development and processing time.

Security. The centre established sound security practices when it started in 1974. Some of these are enforced by the system. For example, the system will automatically disconnect a terminal after it has remained unused for ten minutes. And it will not allow a user to log-on if he has not changed his password within the past 30 days. The centre monitors security compliance and keeps track of all of the passwords.

Marketing. The centre manager also directs the centre's marketing program. At first, the centre grew very quietly. The manager would visit one small group at a time and describe the centre's services. During the last two years, however, the centre has advertised its services more aggressively throughout all of IBM Canada. For example, in 1979 they held a three-day open house at the centre and encouraged all employees to stop by for a demonstration of the products and services.

The people at IBM Canada see the Information Centre filling a need that was not previously being served by the information systems division. Users are comfortable making their one-time requests for information, all by themselves. Interestingly, the people in the information systems division have been able to shift the bulk of their energies away from program maintenance (formerly 70% of their efforts, now 32%) and into major high pay-back application areas.

The information centre has proven to be a very significant productivity tool at IBM Canada. User acceptance has been very positive. The centre is also in a position to support the new office system technology.

Mouser Electronics

Mouser Electronics is a wholesale electronics distributor, located in Lakeside, California, a suburb of San Diego. Mouser supplies over 9000 electronics parts to original equipment manufacturers, hobbyists, and schools. The parts are used in computer, consumer, automotive, and other products.

In 1978 the vice president in charge of operations (and data processing) realized that the Singer System Ten they had acquired in 1975 would not accommodate their expected growth. He began looking for a computer system that:

(1) was user-friendly, (2) was easily expandable, and (3) could be connected with RS-232-compatible peripherals. After eight months of searching, he narrowed the choice down to four mini-computer systems, and finally selected a Prime 400 computer with the 'Information' operating system. The system was purchased through a San Diego distributor who also supplied Mouser with a distributor application system called INFOFLOW. In addition, the Information operating system provides Mouser with many data management facilities—interactive query, report generator, an English-like non-procedural language, security and recovery, data dictionary, file management system, and so on.

Being a small company, Mouser has only one programmer. So providing many Mouser employees with the capability to use the computer directly was very important. The Information operating system has given Mouser this capability.

The Prime system was installed in January 1980. At that time, it included 256K bytes of memory (expandable to eight megabytes), one 96 megabyte cartridge drive (expandable to eight drives), ten Adds terminals (expandable to 63), and three Printronix printers.

After the system was installed, the vice president took several training courses at Prime and studied several documentation manuals. He instructed the managers at Mouser on the use of the English-like non-procedural language and how to run the pre-programmed application programs. The managers, in turn, trained their support people. The vice president and the programmer wrote many standard report programs. He selected all of the computing resources, and he chose the various software options, such as the menu entries that end users see on the system. He also assigns passwords and controls users' access to the system; some have only query privileges, others have query and update privileges. In addition, he controls data dictionaries. Any authorized user can write an ad hoc procedure to, say, generate a new type of report. If that user wants that program stored on the system, it is given to the vice president who assigns it a name and catalogs it on the system. In addition, he expects to install a charge-back program shortly, for charging computer usage to the users.

The Mouser system now has over 800 programs; many of these are short programs created by employees. Mouser is very pleased with its system, because, through the Information operating system and its data management features, they are able to control their business much better because of increased use of the computer. Much of what they have done could not have been accomplished as quickly if they had had to wait for a programmer to write all of their application programs. Their computer has become an office tool that many Mouser personnel use directly to obtain ad hoc information for decision-making.

Lincoln National Life

Lincoln National Life, with headquarters in Fort Wayne, Indiana, is the seventeenth largest insurance company in the United States, according to *Fortune* magazine. They have $4.3 billion in assets and employ some 3,500 people.

In 1978 LNL embarked on a long-term program for distributed processing and office automation. They set as their 1984 goals: (1) reducing the use of paper by 90%, (2) establishing a distributed system, and (3) providing programming capabilities at end-user work-stations. In the January issue we discussed the organizational issues involved in this effort. Now we turn to another aspect—the introduction of end user programming facilities.

After some study of how end users liked existing offerings, they decided to offer end users five facilities in one system: (1) electronic mail, (2) word processing, (3) electronic filing, (4) records management, and (5) an on-line tutorial. The system was to operate on Prime departmental computers. Lincoln National chose Prime's Office Automation System software to handle the electronic mail, word processing, and filing functions. They chose INFO, a file management system and programming language, from Henco, Inc., for the records management facility. And they wrote the tutorial and a 'help' facility themselves. These five functions are accessed through a menu-based front-end that was also written at LNL. Thus users see all of these capabilities as

one system and only need to use one common set of commands. A typical hardware system consists of a Prime mini-computer with 2M bytes of memory, 900M bytes of on-line storage, and up to 400 terminals.

The first automated office system (AOS) was installed in 1979 in the data processing department. Since then, additional systems have been installed in (for instance) the law, travel, and human resources departments. Many other departments have asked to get their people on AOS as well.

Introduction of AOS is the responsibility of the AOS department, with help from a couple of people in the data processing research and development department. In total there is one manager and three staff members who perform the training and user assistance. There are currently 500 AOS users. New users are given a two-hour briefing on how to use AOS to perform all five functions. Users interested in actually writing programs using the INFO non-procedural language can take an additional one-hour course.

Following are just three typical examples of how employees at LNL use AOS to perform end user programming.

In May 1979, one of the law department library staff members began using the system to send messages to library users about new articles of interest; this soon became a regular newsletter. In the introductory class, she had also learned how to define a file, add records to the file, manipulate data from the file, and then list its contents. For her first few inquiries, the AOS staff would explain the commands she could use, and would demonstrate short programs for her.

In June, after one month of use, she wrote her first program—to select a subject, sort on that subject, and report the results. The program allows users to get a printout of abstracts on a particular subject. Her program prints out the pertinent information about the references plus their abstracts and where the complete books, papers, or articles can be found at LNL. So, using AOS, this staff member was able to provide other LNL employees with a library service in her area of responsibility that did not exist previously.

In another instance, the chief financial officer of the company was responsible for compiling the quarterly report on 'critical measures.' The report was about two inches (five centimeters) thick and contained information on that quarter's sales commissions, agent production, staff levels, expenses, mortality and morbidity indicators, and so on—critical measure information that the top level LNL executives and staff members need. This officer decided that the entire report was too cumbersome to create and use; it would be more timely if it were on-line. So, using AOS, he created the file formats, entered the initial data into the files, and programmed some standard output reports. Now, not only is the information available on-line, but it is more up-to-date, because it is entered by the departments that create it, for those departments that have access to AOS.

The third case began last year, when the data processing department was developing a large suspense reconciliation system for the accounting department. This system would allow accounting to better keep track of money between the time it was assigned for disbursement and the time it was actually disbursed—for example, during the period between a policy holder's death and the payment to his or her beneficiaries. The system was estimated to require sixteen work-months of effort using COBOL and CICS (IBM's data communications product). The system had been designed and was being programmed when one of the programmers found out about AOS. He received permission to create a prototype of the system while the production system project progressed as planned. Within two weeks he had created the complete system using the INFO non-procedural programming language. For the next three weeks, the accounting department used the prototype and requested a number of changes and enhancements. Then they asked to use the prototype until the production version was ready. When the production system was finally delivered, the accounting department tried it out—and didn't like it. It did not have the enhancements that had been incorporated in the prototype. So now the accounting department is evaluating the prototype for continued production use.

The demand for AOS at LNL has been overwhelming, we were told. They can not install the systems fast enough. Most of the files are small, and the programs are short. But employees find the system so friendly that they are using it to replace a great variety type of paperwork.

Encouraging end user programming

As we discussed last month, end user programming occurs when non-data processing employees work directly with a computer using programs that they write themselves, with the help of some new software products. These end users could be sales managers, personnel administrators, financial analysts, secretaries, or any other employees who need ad hoc computer services.

Encouraging and supporting end user programming does not appear to be very difficult, judging from what we have seen in several companies. It does take a change of management emphasis, though—a shift away from 'doing everything for the users' to letting end users perform some of their own application programming work.

In some cases this programming is based on the need for new types of reports. If users can design and generate their own reports, they get the results faster than if they had to go through data processing. And most programmers would just as soon have someone else do this type of work anyway.

In other cases the programming that end users can perform is work that does not appear on any data processing backlog list. It is work that is too specialized, too varied, or 'too trivial' for data processing to be bothered with. Generally these applications involve small files which employees would use to keep track of the status of their own work. These files would be very useful to them, but not very useful to many other employees. Data processing generally cannot justify working on such specialized applications.

We have not heard any proponent of end user programming suggest that end users take over the programming work on the larger, more complex applications currently being done in data processing. These should be left to the programming staff.

End user programming is now possible because of some relatively new products. A fairly common name for these products is 'data management systems (DMS). DMS use either a data*base* management system or a file management system, to manage the file data, and have peripheral capabilities, such as an interactive query facility, report generator, data dictionary, restart and recovery routines, perhaps a graphics generator, and maybe some statistical and financial analysis packages. And, very importantly, these products have a language (often non-procedural) which non-data processing employees can use to create new programs.

We have found a growing number of data management systems to be available, from mainframe manufacturers, from mini-computer manufacturers, from national time-sharing vendors, and from numerous software houses. For a listing of DMS products, see Reference 1.

In some organizations, user departments have acquired their own computers to perform their own processing—and they did not necessarily consult with the data processing department beforehand. This can open a 'Pandora's box,' leading to fragmented departmental data, poorly written user programs, and hardware that cannot be connected to a company's data network. Therefore, we suggest that data processing guide the use of end user programming by providing the computing resources, support group, and management.

Here is what we found some companies doing to support end user programming.

Setting up an end user support group

The group that supports end user programming appears to have five functions: (1) selecting the computing resources, (2) providing end user education and assistance, (3) obtaining an end user front-end, (4) developing standards, and (5) providing support administration. No matter how large or small the company, all five functions need to be performed. In small companies, they may be performed by just one person.

Selecting the computing resources

The computing resources that users need include software, hardware and communications.

Software. What end users want is to be able to access data on their own, and then manipulate that data, when the need arises. They may want to print out a report, draw a graph, perform some statistical analysis of the data, check on the status of something, answer a query, or compare data values. Last month we described the various capabilities of DMS that provide many of these desired services. This month we look at another aspect of these products—the end user language.

Since end user systems cannot be forced upon employees, their willingness to try these systems will depend upon how 'friendly' the systems actually are. If the end user language is poorly designed, cryptic, demoralizing, or tedious to use, most employees will give up and just not use the system at all. On the other hand, if the system is well designed, it will increase employees' confidence in using computers, and they will *choose* to use the computer in place of more traditional office procedures. The end user language has a lot to do with the users' acceptance of the system.

The end user language needs to be *easy to comprehend, use, and remember.* Most companies that offer end user languages today point out how quickly employees can learn to use them—usually within a few days. This ease of learning is important, because end users may refuse to use systems that require them to remember complex commands or procedures.

Interestingly, the language also needs to be *powerful.* By this we mean that each user entry should perform a lot of work. The entry could be a single digit, a short command, a sentence, or the touch of a finger or light pen to a CRT screen. Whatever it is, it should convey a lot of information to the system.

The language should also allow combinations of these powerful instructions. Soon after learning to use such a language, end users will want to perform rather complex analyses and calculations. And they will want to be able to store these routines and use them again and again. The more powerful their entries can be, the more employees will use the system, and the more information they will be able to obtain from it.

From the standpoint of data processing, this power has some drawbacks. For one thing, a user can ask a seemingly simple query that actually requires a lot of computing resources. Also, because they are so flexible, end user languages permit users to write inefficient programs.

One company we talked with felt that since these languages make employees so much more productive, such minor machine inefficiencies are unimportant. However, they do try to get their users to write efficient procedures from the outset. One vendor told us that they see many inefficient programs, some of which need to be 'tuned' by data processing professionals. So some basics of efficient programming should be stressed early in the end user education.

End user languages can take many forms. We have selected four of the more common (and rudimentary) ones to discuss here: *menus, commands, dialogs* and *forms.* These are most frequently used to enter queries and report requests. Most of the time these will be used in an interactive mode, where the user sits at a keyboard and 'converses' with the computer directly.

Menus are a very common technique for allowing end users to communicate with computers. A menu lists a number of options, and the user need only indicate which option he desires, either by typing the number of the desired line or using a light pen. In many systems, there is a hierarchy of menus, each increasingly more specific on what functions the user can perform. For example, the first menu might list: (1) word processing, (2) data processing, (3) electronic mail, and (4) filing, as the functions available on the system. If the user chooses word processing, the next menu might list: (1) create new text, (2) edit an existing file, (3) print an existing file, and (4) back to first level.

Menus are especially appropriate at the beginning of a session on the computer, because they allow users to quickly narrow down their options until they get to the point of indicating what file they want to work on.

Menus are popular because they are easy to use—they actually lead users through the system. And they require only short responses from users. Poorly designed menus, however, can be

both tedious and annoying to use, because they slow users down. Experienced users can become very impatient with the hierarchical structure of menus, so the system should allow some shortcuts for users who do not want to wade through several menus.

We have seen condensed 'reminder menus' displayed in one corner or on one line of a CRT screen. These remind the user of his options in performing his task. They are especially good for infrequently used options.

Sometimes menus are created in-house to pull together several products or functions within one system. Other times menus are part of a vendor's package.

Commands are single words that are typed by the user to indicate what function he wants to perform. Typical end user programming commands are: *erase, add, move, sum, table,* and so on. They convey a lot of procedural information in one word.

Once users are at the point where they want to perform some work on the system, then they will generally use such 'data handling' commands. Often these commands may be used either individually or strung together to perform a number of functions sequentially. Some products allow users to add conditional logic between commands. And some even provide synonym dictionaries that allow users to use single letters or even mis-spellings.

Another form of commands are found in *function keys*, where commands are issued by depressing both the terminal's 'control' key and another key. This is even easier than typing in a command.

Dialogs ask for some variable information at each step. Menus are often referred to as a type of dialog. Dialogs are especially helpful for infrequent or new users, because they prompt the user for the required information at every step. The system may even validate each answer before proceeding.

A few systems allow users to create, store, and reuse their own dialogs. One such dialog might be for specifying reports. The user would create a program that, when run, asks the following sequence of questions: (1) enter type of report, (2) for which division? (3) enter starting and ending years, (4) any other selection criteria? and (5) online or off-line? This program could be stored under a name and then run by typing in its name.

As with menus, experienced users can become impatient with dialogs. They would prefer to specify everything in one line. Once again, shortcuts should be allowed so as not to discourage experienced users.

Forms are another type of data handling language for end users. Here users enter non-procedural commands and data field names into table grids. A prime example of this technique is IBM's Query By Example (QBE). QBE is a database management language. For instance, a user who wishes to use it to query an employee data file would first enter the file name. The system then displays a table, where the column headings are the field names—employee name, department, salary, manager, etc. Then the user enters his query by making entries in the appropriate table columns—such as department '301'; salary greater than 15,000 and less than 20,000; and which fields to print out. This is the total query. Data file creation as well as insertions, deletions, and other operations can also be performed using this table format.

For brevity, this discussion has centered mainly on the retrieval characteristics of end user languages. In addition, however, these languages must have facilities for specifying application logic, used when creating new applications. The same 'ease' and 'friendliness' features are just as important here.

Hardware. Next there are the hardware considerations. There are currently three options. One is to install the DMS on the corporate mainframe. This is a very attractive alternative where it allows users to access corporate data maintained under a DBMS on the same machine. In addition, the DMS offered for mainframes are usually quite a bit more versatile and powerful than those now offered on smaller machines.

The second option is to use a time-sharing service. A number of the DMS that operate on mainframes are available through time-sharing services. This alternative is often used when companies, or user departments, want to experiment with the DMS—either to decide whether

they want to bring it in-house, or to build up usage to a point where it becomes cost-effective to install the product in-house. Use of a time-sharing service does not require as large an initial investment as do the other alternatives. But this option may not be as attractive if the users need to access the corporate database, rather than just create and maintain their own files.

The third alternative is to bring in a departmental mini-computer (or even a micro-computer). There are a substantial number of DMS now available for minis, and some for micros. This alternative is attractive where users plan to create and maintain their own files. If they also wish to access corporate files, they will need some communication facilities.

Communications. The third concern of the support group is providing communication facilities. For the various options just mentioned, the communication requirements vary. For the mainframe service, there need only be communication facilities for terminals. For the time-sharing option, dial-up or leased line service needs to be arranged. For the departmental computer, more elaborate communication facilities will be needed when users begin to want to access other computers.

Providing communication facilities may not be an immediate requirement of supporting end user programming, but will surely become important in the future. End users will want to do more than run their own programs. They will want to send electronic messages, access corporate files, and maybe even use outside services, such as bibliographic search services. For these expanded needs, a company network could be required. It would be best to guide the acquisition of computers with a set of data communication standards, so that the machines can be linked in the future.

Providing end user education and assistance

An immediate requirement of end user programming is the need for education and assistance. This really implies some new types of data processing personnel—people who like to teach and help users. Many programmers would not be too excited at this change of assignment; they prefer to solve technical problems on their own. What is needed for the user support group are in-house 'consultants' who have become proficient in the use of the software being offered to end users. They then can provide the initial education and the follow-up assistance.

Some companies have located the few in-house consultants they need in data processing; others have found them in user departments. There really need not be many of these consultants. At IBM Canada, for example, their eleven consultants serve over 800 end users, because most of these users need very little help once they have used the system for awhile.

Larger companies tend to have these in-house consultants take the training provided by the vendor. Then the consultants create an in-house training course and user manual, for training their users. Smaller companies tend to rely on the vendor's training and/or user manual, for training their users. The support group manager might also want to create a short presentation describing the services his group offers, to give at company meetings.

Most importantly, these consultants need to be 'on call' to help users when they have difficulty. Providing telephone 'hot-line' assistance is a key to getting employees to really use the system, we were told. Generally this means that the consultants respond to requests from users. However, at one company, the consultants monitored the use of the system and contacted users whose use fell off. This is probably a good idea, particularly as the more reluctant employees are introduced to the system.

One suggestion we heard was that these consultants should not write programs for users very often. They might want to develop some standard report programs. But for specialized requests, they should only help users write their own programs.

So organizing a user support group takes some forethought. The consultants need to be carefully chosen and trained. An in-house course and user manual may need to be created, and some initial often-requested report programs will need to be written.

Obtaining an end user front-end

At the companies we visited, the DMS provided most of the needed capabilities for their end users. Even so, some felt they required a front-end to supplement the DMS. In some cases this front-end was minimal; in other cases it performed numerous functions.

Most often the front-end consisted of at least one menu that listed the various functions on the system. The menu was used either to identify the functions available in the DMS, or to link the DMS to additional functions, such as word processing, electronic mail, and electronic calendars.

Sometimes the front-end enhanced the DMS, by providing one or more capabilities not available in the DMS; in one such case, a front-end procedure was written to allow users to store their programs in a library. This particular DMS provides the capabilities to reuse the stored programs, but it depends on the operating system to perform the storage. This user company wrote some code to make the program storage function easy for the users to perform.

The front-end may also include an on-line help facility, or a synonym dictionary for using abbreviations of commands.

The front-end can also be used to monitor security measures. For instance, a company may decide that terminals should be disconnected from the system after (say) ten minutes of non-use, or not allow a user to log on if he has not changed his password within the past 30 days. These functions could be performed by the front-end, if they are not provided within the operating system. Likewise the company might want to enforce some good programming practices in the front-end.

The companies we talked with told us that creating the front-end was not a lot of work, but it was something that tied the system together and made it appear tailor-made for the end users.

Developing standards

The major reason why data processing should become involved in encouraging end user programming is so that they can guide the end user efforts. This is especially needed in two areas—communications and data—where creation of standards is important to the company's future ability to manage its information resources. If users begin installing their own departmental computers, without following any centrally-developed guidelines, it is quite possible that within a few years: (1) it will be impossible to connect all of these incompatible machines into one network, and (2) the systems will not be able to share data even if they could be connected.

Users are sure to want inter-connection, whether it is for electronic mail or to obtain services provided on other computers. But in order to provide such connections in the future, data processing (or some other department) needs to establish communication standards to which all company computers adhere. Corporate management should see to it that these standards are developed and then followed.

The data administration function should also have control over corporate data definitions. Here again, the purpose is to allow future commonality among users and systems. Data that must cross organizational boundaries should definitely have standard data definitions. Individual end user work files may not require this control. The distinction between organizational data and local data should be made as early as possible. (Next month, we will discuss some aspects of this subject.)

In addition, it should be decided which databases or files users will be allowed to query and update. The data administration function would be the most likely unit to enforce these decisions. Certain files may be sensitive and access to them should be restricted to specified users.

Administration

In the companies we visited, the end user support group also priced the service, allocated the resources, and marketed the service.

Since the major objective of the support group is to get end users to use the computer, some companies created pricing schemes that encouraged experimentation. One company has given one year of free service to end users. Another has not yet started to charge users. Another inducement would be a fixed fee, no matter what the usage.

The support group manager also needs to estimate how much computing power his users will require during the next year. This is best determined by the users themselves, based upon their past record. In cases where there really are not enough resources to meet all needs, the manager would become the negotiator between the end users and the data processing department; this situation could occur if resource requirements grow more rapidly than anticipated. So careful forecasting will be important.

And third, the new service needs to be advertised. Some companies take the cautious route. They choose the first users of the system—employees who will learn the system easily and set a good example for other employees. But once the support function is running smoothly, the companies offer the service to all interested employees.

These then are the functions we found companies performing when setting up an end user programming support group. These are the more immediate concerns; things that need to be done now. But what about the future? Could end user programming change the way the data processing department operates in the future? We consider this question next.

Future implications for data processing

How might end user programming affect the data processing organization in the future? This is a question a number of people in our industry are pondering. As a sampling of their thoughts, we have chosen three authors. They see the data processing organization evolving into something quite different from what it is today. These changes may or may not come to pass, but they are worth considering. The three authors are: Roger Sisson of Mathematica, John Zachman of IBM, and Franz Edelman of RCA (now in private consulting).

Roger Sisson of Mathematica (Reference 2) believes that the use of DMS in companies will have a decided effect upon the work of the data processing department. He has seen DMS used in four ways.

The first is to produce analytic ad hoc reports from a database. By allowing end users to perform this operation on their own, Sisson has seen this type of reporting performed some forty times faster than if the users had to go to a programmer, explain the request, and then wait for it to be programmed in a procedural language.

Second, DMS are used by end users to write simple applications, which they can run over and over again.

Third, they are used by data processing professionals to create prototypes of complex systems, which users can actually run and test out. A prototype can be developed very quickly, often in days, and can be modified just as quickly until it is to the users' liking. Then it can be used to create the specifications for the system, which is then written in a procedural language. Sisson notes that this process takes less time and is more accurate than writing programming specifications from scratch. We plan to discuss prototyping in the near-future, because it looks like a powerful application development technique.

And fourth, Sisson sees DMS used by data processing professionals to write complete, complex application systems—in one-fifth to one-tenth the time it would take using a procedural language.

Because computer programs—from ad hoc reports to complex systems—can be created in a much shorter time using data management systems, companies will be needing a lot more computer resources than in the past, says Sisson. For example, it is not uncommon for two data processing professionals, who are expert in using a DMS, to be able to write and install five to ten full-scale applications in the two years it would normally take them to install one application using a procedural language. Further, since these applications are accessed by more users for analysis and reporting than the one application would be, at the end of the two years, the demand for computer resources could be five, ten, or even fifteen times what it would be 'traditionally.'

So Sisson sees data processing being called upon to supply a lot more raw resources, especially machines, communication networks, and data. This will cause data processing's role in the organization to change. Data processing will need to create more standards in these areas

than they have in the past. They will need to control and administer more databases and more data definitions. They will need to create and enforce data communication standards. And they will need to create some application standards, such as good end user programming practices. He expects data processing to function more like an in-house information utility which concentrates on systems work rather than on individual applications.

Drawing on Richard Nolan's work on the stages of growth in computer usage, John Zachman of IBM (Reference 3) hypothesizes about stages four, five, and six. He sees stages one, two, and three (initiation, expanded use, then control) as the ones where data processing management learns about managing the *computer.* He notes that during the first two stages, computer applications have generally concentrated on transaction and operational control systems. But during the third stage, a new type of system begins to be requested—the management control system. Management wants to see the data from different angles, hence it begins to request new types of reports. And since the pace of the business environment is accelerating, management wants the information faster. Unfortunately, this has led to a lot of programming (and program maintenance), and larger backlogs of data processing work—which has led to longer (not shorter) responses to user requests. This is where most companies find themselves today. Conventional development methods are not giving management the responses they want, so data processing is looking for new approaches.

One approach is database technology, because it allows data processing to identify a group of related data elements—say, financial data—put that data into a database, and let users access the data whenever they need it. Data processing retains control over the data; users simply make use of the data. This approach, Zachman says, denotes a subtle, yet very important, change in the role of data processing. He sees it leading to stages four, five, and six.

Zachman sees stages four, five, and six as the ones in which data processing learns how to manage *data*—which is different from managing the computer. He believes that about half of the larger companies are just beginning to make this transition into stage four. He expects their learning curve to follow the classic curve—initiation, expansion and then control. Once end users find out how much the direct use of the computer can help them, they will want more—a whole lot more, says Zachman—until management steps in to control the situation. This progression represents Nolan's stages four, five, and six.

The impetus for data processing to manage data rather than computers is coming from management, says Zachman. For one thing, the world has moved from a period of natural resource plentifulness (oil, water, minerals, etc.), where growth was stressed, to a period of resource shortages, where resource use optimization is more important. This requires management to formalize its organization's planning and control apparatus, which will increase management's perception of the value of data. Therefore, Zachman sees executives asking for consistent views of their organization's data which they can manipulate in order to make decisions. He also expects management to begin asking for models—business models, models of the company's external environment, and so on. These types of applications are very different from transaction systems.

So Zachman sees the data processing department becoming, in stage six, the manager of the data resource, just as the finance department has become the manager of the cash resource. And just as finance creates a chart of accounts and develops and enforces policies for managing cash, data processing will be asked to develop a data architecture upon which systems can be designed and built. And they will develop and enforce policies and procedures for managing data. The responsibility for processing much of the data could well move out to the users in this future environment.

Looking at the evolution of data processing, Franz Edelman of RCA (Reference 4) suggests that, from the mid-1940s until the mid-1970s, the concern in data processing was computing systems and computer programs. Thus the programmer was the key person who created applications to automate manual paperwork procedures.

But information is now being recognized as a prime asset of organizations, second only to human resources. Organizations are spending a lot more on managing their information than people realize. After several years of study, people at RCA discovered that they spend about *10% of their gross revenues* on information management; this breaks down into 1% for traditional data processing, 2% for administrative systems, and 7% for their information (white collar) workers, but not including managers and professionals. With all of this effort expended on managing information, the information function should broaden its perspective, he says. It should begin to concentrate on supporting the managers who run the business, rather than on the clerical transaction operations. He feels this change in emphasis began taking place in the mid-1970s, when a few companies began to discover the value of transaction data to support their managerial, rather than their clerical, processes.

Edelman believes the availability of three technologies—interactive systems, database management systems, and non-procedural languages—has made this change possible. These technologies allow companies to get the computer applications middleman—the programmer—out of the picture and let end users access the data directly.

But what organizational structure will support this broadened perspective? Edelman see a 'bi-partite organization' as the current best solution. A bi-partite data processing organization separates information production and distribution from information application. That is, one part of the data processing organization continues to focus on the computer and communication technologies—becoming the corporate information utility. A new organization within data processing takes over the applications end, and concentrates on solving business problems. This organization could very well move into the functional areas of the business.

Edelman says this organizational structure will work only so long as top management plays a leading role in (1) resolving conflicts between these two organizations, and (2) helping the applications organization develop its requirements for business planning, control, and operational systems.

Edelman sees end user programming as 'inevitable' and will bring with it the need to change the data processing organization, if not the profession itself.

As we noted last month, end user programming is coming, because it offers just too many benefits to end users. We believe data processing should encourage and guide the trend so that the organization will benefit from standard communication interfaces, common data definitions, and more productive employees. In turn the trend could affect what data processing does in the future.

REFERENCES

1. For a free lisiting of DMS products that we have come across, write to EDP ANALYZER, 925 Anza Avenue, Vista, California 92083.
2. Sisson, Roger, "Solution systems and MIS," *Proceedings of the Twelvth Annual Conference of the Society for Management Information Systems*, September 22-24, 1980, Philadelphia, Pennsylvania, SMIS (111 East Wacker Drive, Suite 600, Chicago, Illinois 60601), pp. 177-182; price $16.50 prepaid.
3. Zachman, John, "Information systems: The decade of the 80s," *Proceedings of SHARE 55*, August 17-22, 1980, Atlanta, Georgia, SHARE (111 East Wacker Drive, Chicago, Illinois 60601); pp. 204-214; price $50 prepaid.
4. Edelman, Franz, "Management of information resources: A challenge for American business," *MIS Quarterly*, SMIS (address above), March 1981, pp 17-27; price $8.

Prepared by:
Barbara C. McNurlin
Associate Editor

SUBJECTS COVERED BY EDP ANALYZER IN PRIOR YEARS

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
9. The Automated Office: Part 1
10. The Automated Office: Part 2
11. Get Ready for Major Changes
12. Data Encryption: Is It for You?

1979 (Volume 17)

Number

1. The Analysis of User Needs
2. The Production of Better Software
3. Program Design Techniques
4. How to Prepare for the Coming Changes
5. Computer Support for Managers
6. What Information Do Managers Need?
7. The Security of Managers' Information
8. Tools for Building an EIS
9. How to Use Advanced Technology
10. Programming Work-Stations
11. Stand-alone Programming Work-Stations
12. Progress Toward System Integrity

1980 (Volume 18)

Number

1. Managing the Computer Workload
2. How Companies are Preparing for Change
3. Introducing Advanced Technology
4. Risk Assessment for Distributed Systems
5. An Update on Corporate EFT
6. In Your Future: Local Computer Networks
7. Quantitative Methods for Capacity Planning
8. Finding Qualified EDP Personnel
9. Various Paths to Electronic Mail
10. Tools for Building Distributed Systems
11. Educating Executives on New Technology
12. Get Ready for Managerial Work-stations

1981 (Volume 19)

Number

1. The Coming Impact of New Technology
2. Energy Management Systems
3. DBMS for Mini-computers
4. The Challenge of "Increased Productivity"
5. "Programming" by End Users
6. Supporting End User Programming

(List of subjects prior to 1978 sent upon request)

PRICE SCHEDULE (all prices in U.S. dollars)

	U.S., Canada, Mexico (surface delivery)	Other countries (via air mail)
Subscriptions (see notes 1,2,4,5)		
1 year	$48	$60
2 years	88	112
3 years	120	156
Back issues (see notes 1,2,3,5,)		
First copy	$6	$7
Additional copies	5	6

NOTES

1. Reduced prices are in effect for multiple copy subscriptions and for larger quantities of a back issue. Write for details.
2. Subscription agency orders are limited to single copy subscriptions for one-, two-, and three-years only.
3. Because of the continuing demand for back issues, all previous reports are available. All back issues, at above prices, are sent air mail.
4. Optional air mail delivery is available for Canada and Mexico.
5. We strongly recommend AIR MAIL delivery to "other countries" of the world, and have included the added cost in these prices.

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_____

DEVELOPING APPLICATION SYSTEMS FASTER

Two conference sessions that we attended in early May highlighted a point of interest to all data processing managers: how to get new application systems developed faster and cheaper. One session discussed the steps needed in "conventional" application development; even for a small system, the number of steps is imposing and time-consuming. The other session discussed the development of applications by using an application generator package. To this second category we would add "data management systems," which we have been emphasizing in recent issues.

The case of the Lincoln National Life accounting system, described in this issue, illustrates what these new tools can do for application development. What took 10 work-months using conventional methods needed only five weeks elapsed time (and even fewer work-weeks) to create by prototyping. What is more, the prototype version served user needs better than did the so-called production system.

But the benefits of application generators and data management systems do not stop with faster and cheaper development. From our talks with users of these tools, we get the message that maintenance time and costs also are slashed.

Life may not be completely rosy for new users of data management systems or application generators. Why? One main reason seems to be that programmers resist these new tools, because they fear the tools will decrease their marketability.

We will return to the subject of system development by prototyping in a near-future issue. We believe the subject is so important that it should be near the top of the priority list for most data processing executives.

Editorial: Richard G. Canning, Editor and Publisher; Barbara McNurlin, Associate Editor. While the contents of this report are based on the best information available to us, we cannot guarantee them.

Missing Issues: Please report the non-receipt of an issue within one month of normal receiving date; missing issues requested after this time will be supplied at the regular back-issue price.

Microform: EDP ANALYZER is available in microform, from University Microfilms International, Dept. P.R., (1) 300 North Zeeb Road, Ann Arbor, Mich. 48106, or (2) 30-32 Mortimer Street, London WIN 7RA, U.K.

Prices: Prices of subscriptions and back issues on page 13.