EDP ANALYZER

JANUARY, 1978
VOL. 16, No. 1

INSTALLING A DATA DICTIONARY

During the 1960s, the most computerized application systems were designed with their own data files. Both the data and the responsibility for the data were fragmented—and many of those systems are still operating today. During the 1970s, one major trend has been toward the use of the shared-data data base that serves multiple applications. Numerous benefits are being obtained from data bases, including the reduction of undesired redundancies and incompatibilities of data. But at the same time, data bases have brought into focus the need for assigned responsibility and control of the organization's data resources. The data administrator function is being set up and given this responsibility. And data dictionaries are being installed as tools for the data administrators to help them perform this function. Here are some user experiences in putting in data dictionary systems.

Data dictionaries are systems and procedures—either manual or automated—for the storing and handling of an organization's data definitions. In theory, their reason for existence need have nothing to do with computers. Well-managed organizations should have clearly specified definitions for all of the data items used by the organizations, according to theory. But this is not the way things have worked out in practice.

In practice, the units of an organization—for instance, the departments—have developed systems and procedures to meet their needs, over the years. Each organizational unit has developed its own data definitions. Only when certain data items must be regularly exchanged among organizational units have standard definitions emerged. The data definitions for financial data are the ones most frequently subject to standardization, since financial data is collected from all organizational units.

The result has been that, in most organizations, the same data items have been given different names and different definitions by the different units of those organizations. Similarly, the same names have often been given to quite different data items. Because of this, management has found it very hard to compare data from the various units.

A data dictionary might very well be a paper-based system for pulling together all of the data definitions used by the units of an organization, in an attempt to reduce undesired redundancy and remove inconsistencies. Or a data dictionary could just as well be a computerized system to do this same function. There are numerous other functions that a data dictionary might perform, as we will discuss.

One thing stands out in the above discussion. A data dictionary does not, in itself, generally produce operational data for the organization. It does not produce invoices, or job orders, or product catalogs, or such. Instead, its purpose is more

to eliminate the errors of understanding, the ambiguities, and the difficulties in interpreting the data. As such, a data dictionary is an overhead item. And unless the costs of those errors, ambiguities, and difficulties are clearly evident, it may be hard to justify the installation of a data dictionary.

The installation of computerized data bases and data base management systems has been bringing into focus the need for data dictionaries. Our discussion will center on mechanized data dictionaries that support computerized systems. Let us now look at some of the advantages and some of the problems associated with mechanized data dictionaries.

Advantages of a dictionary

Lefkovits (Reference 1), Leong-Hong and Marron (Reference 2), Ehrensberger (Reference 3), Schussel (Reference 5), and EDPACS (Reference 6) provide good discussions of the advantages of data dictionaries, from which the following is drawn.

Control of the data. By identifying the different definitions of the same data item, through the use of a data dictionary, an organization can standardize on one definition for that data item. Further, the accepted definitions for all data items will be found in the dictionary. Hence, unintentional redundancies will be largely eliminated. Further, discipline will be imposed on the introduction of new data items or changes to existing data definitions.

The data dictionary can be used as an enforcement tool. It can help enforce the use of the standard data definitions. It can be used to enforce security safeguards, both for data entities within the data bases as well as for the data dictionary itself. And it can be used to enforce the use of standard edit and validation routines.

In addition, the data dictionary can be used as an audit tool. It can help auditors—internal auditors, external auditors, system reviewers, etc.—gain an understanding of the systems to be audited. Further, the dictionary can provide the data standards against which the audits are to be performed.

Improved system development and control. By providing centrally maintained data definitions, the data dictionary can support programming standards and naming standards. Both systems design and programming can be done faster because the data is already defined. System and program maintenance are made easier because complete documentation of the data is provided. By displaying all uses of a given data item, the full impact of a proposed change can be assessed. Further, the time that will be required to make the change can be more accurately determined.

By providing both user-oriented and computer-oriented definitions for each data item, the data dictionary can help provide better communication between users and the system development staff.

Automatic generation capability. The data dictionary can be interfaced (bridged) to other software systems, so as to automatically generate input for those systems. Program data definitions (such as COBOL data divisions) can be supplied by the dictionary, as can program input-output area definitions. The definitions required by the data dictionary associated with the data base management system can be automatically generated. The dictionary can generate reports about the data definitions, as well as documentation about the data base.

All in all, then, a data dictionary offers *many* potential benefits for the user. But all is not milk-and-honey. There are some problems, too.

Problems with dictionaries

Perhaps the first problem that confronts the potential user of a data dictionary might be termed the what-where-when problem. What will it be used for? Where will it be obtained from? When (in relation to other activities) will it be installed?

Here are some of the factors that have to be considered in making these decisions. *Basic type of dictionary:* will the dictionary be manual or automated? As mentioned above, we are assuming that an automated dictionary is being considered. However, many of the same considerations apply to manual data dictionaries. *Uses:* will the dictionary be used to support system development, system maintenance, productive use of the data base, or a combination of these? *What type of mechanized dictionary?* must the dictionary be independent of any particular data base management system, or may it require the use of a

specific DBMS? *Source of the dictionary:* will the dictionary be purchased or will it be developed in-house? *Timing of the installation:* will the dictionary be installed prior to the installation of a DBMS or after it? *Scope of the project:* will the dictionary be used for all data, forms, and processes used by the organization, or will it be limited to just the mechanized data, or will it apply only to data base data?

Hopefully, this brief listing of factors will give some idea of the inherent complexity of the what-where-when problem. It is true, of course, that many prospective users will make the same types of decisions that numerous dictionary users have made in the past. These are the so-called "practical" decisions; for example: "We want an automated dictionary to support our current data base applications; we do not care if it uses the same DBMS that we are using or not, but we will buy it, not build it; we want it to support the productive use of our data bases, and we will use it for data base data only." As "practical" as these decisions seem to be, we will attempt to show in this report that the planning for a data dictionary really ought to take a much broader view.

Lefkovits (Reference 1) points out some other problems associated with data dictionaries. In order to achieve the benefits, the user organization must have (a) a good degree of commitment by management, users, and data processing personnel, (b) an effective data administration function, and (c) an effective method for planning the introduction of change into information systems.

Even if these conditions are met, says Lefkovits, the characteristics of the data dictionary selected can lead to installation problems. For instance, some dictionaries require the use of short (too short, in Lefkovits' opinion) data names, while others allow for longer "natural" names. Some support mechanized data in conventional files (tape or disk) much better than others. Some use free-form commands while others use fixed-form. In addition to the ability to define data entities, some systems also allow the definition of process entities (systems, programs, or modules), and usage entities (users, terminals, etc.). These are only a very few of the significant differences pointed up by Lefkovits. If you are considering a data dictionary, or are considering replacing your current one, we strongly urge you to read Reference 1.

Field experience problems

We talked to Keith Setzer, an executive at University Computing Company, Dallas, Texas, about field experiences with data dictionaries. UCC has developed and is marketing the widely-used UCC 10 data dictionary system. We asked Setzer about some of the things to do and not to do, when installing and using a data dictionary, based on what his company had encountered.

But first, a few words of background about UCC 10. UCC obtained IBM's IMS data base management system in 1969, for in-house use. Very soon, they saw a need for a data dictionary, to help them get their data definitions under control. The UCC 10 was developed in 1970. Originally it was a batch system, in which the data definitions were printed out. These printouts could be quite voluminous—and expensive, when an entire printout had to be obtained just to get one change. So in 1972, an on-line query and update facility was added. The on-line query ability is available for both production and test status data, but on-line update is limited to test status data definitions only. In 1973, UCC started marketing UCC 10. It is designed for IMS data bases and works with IMS and IMS/VS. So Setzer's comments were based not only on what UCC representatives had observed in the field but also upon UCC's own use.

Situations that typically lead to problems. Setzer identified three situations that typically lead to serious problems or to outright failures to install a data dictionary. *A political ploy:* The acquisition of the data dictionary may be a part of an actual or an apparent political ploy. Perhaps the reason given for the acquisition is that the computer center is running short of disk space and wants a dictionary to help eliminate redundant data. But user departments may see the acquisition of the dictionary as one more effort to gain more control—in this case, over the data definitions. The users may resist the dictionary, in order to retain control over "their own" data definitions.

Huge clean-up problem: Setzer pointed out that some organizations have been using a DBMS for five or six years, mainly as an access method rather than as a true DBMS. Many data base applications are running, but with little integration among them. Much redundancy and many incon-

sistencies exist in the data definitions, and there is little or no current documentation of those definitions. No data administration function has been established. In such a situation, the organization has a huge clean-up job ahead of it, if it wants to use a data dictionary effectively.

Budget pressure: If the organization's financial problems lead to significant budget reductions, one of the first things that probably would be cut would be the data dictionary project. As we pointed out earlier, a data dictionary is an overhead item, as is the data administrator function. Neither can stand up well in the face of budget cuts.

(We did hear a report that some companies are finding that this function reduces system analysis and design times and costs sufficiently to cover the costs of the function. But we have not talked to users on this point.)

A not-bad situation for a dictionary. On the other hand, said Setzer, if the organization obtained its DBMS about a year previously, and had developed one pilot application and had put it in to production, the chances of success for a dictionary might be much improved. Such an organization might well see the need for controlling the data definitions. In fact, they might want to install the data dictionary before putting on more DBMS applications.

The best situation for a dictionary. Ideally, said Setzer, the data dictionary should be considered at least as soon as the DBMS is considered. If the organization is willing to consider setting up the data administration function, developing data standards, and installing the data dictionary as the first data base application, then the chances of success are much enhanced.

The importance of data administration

The effective use of a data dictionary really requires an effective data administration (or data base administration) function, said Setzer. Here are some of the things that the function must be able to do, in his view.

Clean up the data definitions. The data administration function must have the responsibility and authority to get rid of undesired definition redundancies and inconsistencies. The organization should not allow two or more names to exist for the same data item, nor should it allow the same name to be used for two or more different data items. It may take the data administrator some time to accomplish this, of course.

Control "corporate" data definitions. While data that is used solely by one organizational unit might be considered as "local" and under the control of that unit, data used by two or more units should be considered as "corporate." The data administration function should be able to control the acceptance of and perform the updating of all "corporate" data definitions.

Control the changes to "corporate" data definitions. The data administration function should be charged with analyzing the impact of all proposed changes to "corporate" data definitions. All programs that would have to be changed should be identified before approval to make the change is given. A data dictionary can be of tremendous help here, said Setzer. It provides one place to look for all uses of the data. Otherwise it may be necessary to get printouts of all program and data base data definitions and to manually scan them. Finally, approval to proceed with the change might be held up until all affected programs have been changed, to keep those applications from aborting.

Oversee some security functions. The data administrator must be able to assign security levels and "need to know" clearances for the use of both the data dictionary and the data bases. Even if a user is entitled to access certain data in the data base, the access might be limited to read-only, as opposed to update. Further, the data administrator might be the only one allowed to update the data dictionary; all changes to definitions would have to go through this function.

To illustrate how an organization can approach the installation of a data dictionary in a desirable manner, let us consider the experience of Macy's California.

Macy's California

Macy's California Division, with headquarters in San Francisco, Calif., is a division of the R. H. Macy Inc., a large department store chain. *Fortune* magazine ranks Macy's as 27th in sales vol-

ume among U.S. retailers, with sales of over $1.4 billion annually. The corporation has about 38,000 employees.

Macy's California has 20 stores throughout central California. The data processing for these stores is performed at the division's computer center in San Francisco. The division is using an IBM 370/158 and there are 28 people on the development staff.

While Macy's California is not yet using a data base management system, data processing management recognized several years ago that a DBMS probably would be a desirable acquisition. But it was also recognized that it would be helpful to install a data dictionary system in order to get the data definitions and documentation under control. So management decided that the data dictionary would be considered ahead of the DBMS. A project was set up in mid-1976 to investigate and evaluate alternative data dictionaries. The assistant manager of system development was assigned to the project.

A policy decision made early in the project shaped the whole study. This decision was that the data dictionary should not influence the choice of the DBMS. If a data dictionary works with only one DBMS, it was not to be considered. Also, Macy's wanted a dictionary package that could handle conventional tape and disk files as well as data base data definitions. A number of leading data dictionaries were thus eliminated from consideration.

After looking at the major data dictionaries on the market, two prime candidates emerged. Both were studied in some detail, and users of both were visited and interviewed. About two months after the project started, the DATAMANAGER data dictionary, developed and marketed by MSP, Inc. (of Lexington, Massachusetts) was selected for detailed evaluation.

Having selected what seemed to them to be the most suitable data dictionary for their needs, Macy's decided to use the package's trial period to find out just how well in fact it met their needs. They also wanted to find out what would be required of them in order to make effective use of the dictionary.

The first step in this evaluation was to select one existing application system and put all of the data definitions from that application into the dictionary. In performing this step, they encountered the problems of loading a dictionary. Some of the data definitions could be obtained automatically from COBOL programs. But these definitions were not complete, as far as the dictionary was concerned. And in some instances, the definitions had to be changed. What Macy's learned from this exercise was that it is not wise to jump too fast into the automatic loading of a dictionary.

Another experiment was to test the interfaces between DATAMANAGER and the PANVALET library system they use. The data definitions that were picked up automatically from the COBOL programs were outputted to the library system. From this test, they obtained some idea of the type of operation they could plan on.

Macy's California also went through the exercise of developing "integrated" data definitions, for a few of the more widely used data elements. For instance, such terms as "store," "department," and "class" are widely used within a multiple branch department store. The experiment found a number of variations among the existing definitions, and then attempted to obtain standard definitions that were acceptable to the user departments, system analysts, and programmers. This test pointed out to them just what would be needed in order to develop naming conventions and standards.

There were a number of features of DATAMANAGER that Macy's liked particularly. Since this dictionary interfaces with a number of the leading DBMS, they liked the freedom of choice it provides them in this selection. They like the fact that it allows for defining process entities as well as data entities. They like the dummy entity capability, which allows setting up the definitions for a new entity as the need becomes apparent but before very much is known about those definitions.

There were also some features that Macy's found they wanted in their data dictionary. One was the facility to handle "clerk" entities in much the same manner as "program" entities. For each "clerk" entity, the definitions would indicate the inputs used, files accessed, logic of processing, and the outputs generated. Other users of DATAMANAGER were asking for this same feature, so MSP has implemented it by allowing other than standard member type names. For instance, instead of using the normal system-program-item hierarchy for entities, one can use a company-department-clerk hierarchy, or some other such hierarchy.

Macy's believes that such facilities make the dictionary easier to use for defining *all* data, process, and usage entities.

By the end of the trial use period, the people at Macy's California had decided that they liked DATAMANAGER and that it would serve their needs well. So the dictionary was purchased. But at the same time, they had found a number of activities that should be done in preparation for installing the dictionary. One of these activities was to develop the division's standard data definitions. When we visited them, they were in the midst of this activity. When all of these necessary preparatory activities have been accomplished, the dictionary will be installed.

And after the dictionary has been installed, the people at Macy's California expect to consider the question of what DBMS they want to use.

THE ROLE OF THE DATA DICTIONARY

There are two major objectives for installing a data dictionary system. These are:

- Getting all data and process definitions under control
- Getting just mechanized data definitions under control

We will discuss each of these briefly.

Getting all data definitions under control

Some organizations see the data dictionary as a tool with which the data administrator function can get all of the data, process, and usage definitions under control.

Just what does this objective really imply? Ehrensberger (Reference 3) says that a data dictionary should be able to contain information about the following: data bases, files, fields, transactions, source documents, reports, systems, programs, users, departments, projects, standards, security levels, and personnel.

Another interpretation of this objective is that the dictionary should be able to store and handle the definitions for the organization's data (in filing cabinets, on forms, in conventional mechanized files, in data bases, etc.), the organization's processes (used by clerks, managers, programs, etc.), the organization's systems, (data processing, word processing, data communications, etc.), and the organization's users of these entities (people, departments, divisions, etc.).

At first, such objectives might appear to be almost unattainable. But, in fact, they are not at all impractical. As a matter of fact, there is one data dictionary on the market—PRIDE-Logik—that is specifically designed to help an organization get started in this direction. To illustrate, consider the experience of Marathon Oil Company.

Marathon Oil Company

Marathon Oil Company, with headquarters in Findlay, Ohio, is a fully integrated oil company. It has annual sales in excess of $3 billion and employs almost 12,000 people worldwide. For its data processing, Marathon uses an IBM 370/168 and the TOTAL data base management system.

1974 was a difficult year for the data processing department at Marathon Oil. It was a period of rapid growth in the system development staff and, to compound the problem, it was a period of significant staff turnover. The result was that there were many new people added to handle the growing system development workload. The new people were using a variety of techniques and practices. Projects were slipping behind schedule and data processing management was finding it hard to know the status of each project.

In looking for a solution to these problems, data processing management at Marathon came across PRIDE, developed and marketed by M. Bryce & Associates, Inc., Cincinnati, Ohio. We discussed PRIDE in our December 1974 issue. PRIDE is a structured system development methodology that uses nine well-defined phases. All of the necessary work products are defined for each phase. Development projects that are conducted under PRIDE can thus progress in a standard, controlled manner. Marathon liked the ideas they found in PRIDE and purchased it in late 1974.

From its inception, PRIDE has incorporated a complete data management philosophy. That is, *all* data definitions and process definitions are captured that fall within the scope of the application system being built. Initially, this function was performed manually. In 1974, it was mechanized under the name PRIDE-Logik. PRIDE-Logik thus provides the data dictionary function for PRIDE, operating in a batch mode. However, it is not necessary to use PRIDE in order to use PRIDE-Logik.

During 1975 and 1976, Marathon was well satisfied with their use of PRIDE. So in early 1976,

they decided to obtain and install PRIDE-Logik, so as to upgrade the data management function for system development and maintenance. They also set up the data administration function (which they call the "data manager" function) under the systems manager.

(In describing their use of PRIDE-Logik, the people at Marathon pointed out that their efforts have been most successful when they were dealing with new application systems that did not share data with existing application systems. The discussion that follows is based on such situations. However, where an application system is being developed that enmeshes with one or more existing systems whose data definitions are not under PRIDE-Logik, non-trivial problems do arise. "It is not all peaches and cream in such cases," we were told.)

Use of PRIDE-*Logik.* As indicated earlier, PRIDE itself has nine phases. Phases 1 to 3 cover the system study, system design, and sub-system design. Phases 4 to 6 cover the design of both administrative (manual) and computer procedures, program design, and program test. Phases 7 to 9 cover system testing, system operation, and system audit.

At the end of the first phase, the system analysts at Marathon enter skeleton definitions of the new system into PRIDE-Logik. These definitions are the first pass at the files, inputs, and outputs that will be needed. By the end of the second phase (system design), the analysts are able to enter narrative descriptions of the entities, plus rough formats of the system outputs to show to users. Also, additional definition information is entered for inputs and files.

Further definition information is entered during the third phase (sub-system design). By this point, the sequence in which the sub-systems and their files are to be processed can be determined. And during the design of the administrative procedures, the analysts can enter the definitions of the clerical documents and the data that will be on them. Also, at the end of phases 1, 2, and 3, PRIDE-Logik has the facility to perform system diagnostics, such as checking all entities for completeness and logical flow.

Up to this point, PRIDE-Logik has been used mainly as a repository of information. If there are questions and queries about the data definitions, these can be answered from the output documentation of the dictionary. But it is during the design of the computer procedures that Marathon begins to use the powerful analytical capabilities of PRIDE-Logik.

As computer procedure design starts, the programmer(s) ask for printouts of all documentation applying to the programs they will be working on. This documentation covers record definitions, output definitions and formats, and the like. But in addition, they ask PRIDE-Logik to track all data elements, from their source to their ultimate use. The dictionary system flags all data fields that have been defined but not used, records specified to be used in a program but for which no input has been provided, and so on. In short, PRIDE-Logik looks for inconsistencies in the definitions. Note that this analysis covers not only the mechanized part of the new application but also all of the manual procedures that provide input for it and use the outputs from it.

If mistakes are found at this point, the analysts are called back in to make the corrections. The new definitions are entered at the sub-system design point and all of the subsequent processing is repeated, to see if anything else comes to light. When the corrections pass all of the tests, the programmers proceed with computer procedure design.

At the end of the seventh phase (system test), and after the users have accepted the new system, all of the definitions are put under the control of the data administrator function. From that point on, changes to the definitions can be made only by submitting them through the data administrator function. In addition, all proposed changes are assessed for their total impact on the application system by using PRIDE-Logik.

While Marathon Oil is pleased with their use of PRIDE-Logik, they also point out that they are still in the process of learning to use it. In general, the system analysts provide input to the dictionary, for defining the new system, and the programmers use the outputs from the dictionary for designing and coding programs. Analysts find that they are entering information and retrieving some definitions but as yet are making little use of the analytical powers. The programmers were using the analytical capabilities of the package but find that they have lost some of their previous data definition perogatives. And the new function of data administration has taken away functions formerly

performed by both the analysts and programmers.

So, said data processing management at Marathon Oil, a tool as powerful as this data dictionary has many benefits but it also has some non-trivial side effects. It brings out into the open many differences of opinion between analysts, programmers, and data administration. This is not due to characteristics of the package itself but rather is due to the fact that a new discipline is being imposed that changes the traditional ways of doing things.

When we visited Marathon Oil, they were using PRIDE-Logik simply to support the application system development process. Sometime in the future, they expect to extend the use of the dictionary into daily operations, by means of interfaces with their PANVALET program library system and with their TOTAL data base management system. Further, they hope to use PRIDE-Logik as a repository of test cases flowing from users, programmers, analysts, internal auditors, and data administration.

It is clear, we think, that as more application systems are developed using PRIDE-Logik, and as major enhancements occur that use the same process, Marathon Oil will gradually get *all* of its data and process definitions into the data dictionary. Further, this goal will be realized as a by-product of the company's system development and maintenance process.

Just in case the use of PRIDE-Logik may seem like a "special case" to you, consider the remarks made by Chris Gane at the DATAMANAGER Users Group conference held in New Orleans, Louisiana, in May 1977.

Remarks of Chris Gane

Chris Gane, of Improved System Technologies, Inc., New York City, was an invited speaker at the above-mentioned users group meeting. The subject of his talk was, "DATAMANAGER in a structured analysis environment." The paper is included in the proceedings of the meeting, Reference 4.

In his talk, Gane went through an example of the system building process, using an easily understood application—that of a company that advertises books for sale, receives mail orders from customers, and enters orders to publishers to replenish its inventory. Using a top-down approach, Gane gradually adds more detail into the application. As he does so, he identifies and builds up data flows, definitions of data stores, and definitions of the data. Also, some of the logical processes that will be used are defined. Then, says Gane, a logical data dictionary should be able to store and handle the definitions of data flows, data stores, data elements, and possibly some of the process logic.

These are essentially the same entities that Marathon Oil is storing and handling in its data dictionary.

Since Gane was addressing the DATAMANAGER Users Group, he assessed the capabilities of DATAMANAGER for performing these functions. Note that DATAMANAGER was originally developed to support a DBMS and conventional files in daily computer operations, and Gane was assessing its ability to support the system development process.

DATAMANAGER clearly could handle the data element definitions and the data group definitions, said Gane, since that is what it was designed to do. If additional definitional information is desired to support the development process than is normally carried in the dictionary, it can be handled under the NOTE capability.

Data flow definitions can also be handled, although not as conveniently, said Gane. These definitions would include source, destination, description, data groups that are used, and so on. There seems to be no reason why current capabilities cannot be adapted by the user to this purpose, he said.

Three other types of entities—process definitions, definitions of external entities (such as other systems, departments, etc. that are external to the system being developed), and glossary items—also can be handled by the dictionary.

With such information in the dictionary, there are a number of outputs that the users would desire, said Gane. One is ordered listings of the entities, and another is cross-reference listings that show relationships between entities. Both of these capabilities are already available in DATAMANAGER, he said. Two other capabilities are desired but not yet available in the package. One is the ability to search the dictionary based on a keyword or character pattern. The second is consistency and completeness checking to see, for instance, if there are data flows without sources or destinations, or data elements in data stores that

have not been entered via input.

Is it practical to consider getting all of an organization's data and process definitions under control via the use of a data dictionary? Well, consider the user experiences that we have discussed. DATAMANAGER and PRIDE-Logik are two quite different data dictionary systems, originally designed with quite different goals in mind—the former mainly to support production and the latter mainly to support development. But they seem to be tending in the same direction. PRIDE-Logik interfaces with several DBMS,, for production use, and DATAMANAGER can support many development functions, as discussed. Yes, it is practical to consider such a goal.

Getting mechanized data definitions under control

The objective of getting only the mechanized data definitions, as opposed to *all* of the organization's data definitions, under control is the more typical situation, from what we have observed. This goal has two subsidiary sub-goals: (a) "cleaning up the mess" and (b) aiding end users.

"Cleaning up the mess"

In most organizations that we have talked to, the data definitions and process definitions are "not in good shape." Typically, organizations have many conventional tape and disk files supporting numerous applications. Among these many files, the same basic data item might occur in two or more files, have two or more names, use different codes, have different field lengths, and so on. Even data bases are not immune to this disease. Some organizations use many small data bases, and treat the DBMS as just another access method. In such situations, a data dictionary may be installed to help clean up and standardize the definitions.

Lefkovits (Reference 1) identifies three kinds of "bridges" offered by some data dictionaries to help in this clean up function. One type of bridge is used to collect existing data definitions from existing files, program data divisions, and DBMS directories, and then load the dictionary with them. A second kind of bridge supplies data definitions to conventional files and their programs. And a third kind of bridge supplies data definitions to the data directory of a DBMS. Lefkovits points out that there are significant differences among the dictionaries on the market as to the bridges they provide.

These bridges can be used to help clean up the existing data definitions and to insure that the same practices are not continued in the future.

Identification-of-problem mechanism. Using the bridge mechanism to load the dictionary with existing data definitions can assist in identifying the problems. All of the definitions are brought together at one point. Standard formats help in making comparisons. When undesired redundancies and inconsistencies are uncovered, the data administration function can work with users, analysts, programmers, and others to select standard definitions that will be used in the future. These standard definitions can be used in all new application systems and can be incorporated in existing applications when other changes or enhancements are made to them.

As we pointed out earlier, this process probably should be done with only a portion of the total mechanized data definitions at a time. It might be relatively easy to load all current definitions into the dictionary" via the bridge. But the magnitude of the "mess" might be so great that people are discouraged from even starting the clean up.

Enforcement mechanism. As more and more standard data definitions are entered into the dictionary, the other two types of bridges can be used as a part of an enforcement mechanism. At some logical point in time, the policy can be adopted that henceforth all program data divisions, all input-output area definitions, and all DBMS directory inputs will be obtained via the dictionary. No direct inputs will be allowed. The dictionary can thus help insure that only the approved data definitions are used.

Lefkovits points out that, in the future, dictionaries might well be integrated with operating systems and DBMS. All accesses to data would first flow through the dictionary, to pick up the data definitions. When a change has been made to a data definition, all programs using that data definition would be prevented from running until any necessary changes had been made to them and they were once again "released" for production.

Aid to end users

A dictionary can assist end users in several ways. Users can look up the approved data defini-

tions, in order to interpret data on reports. Users can also find out if particular types of data items in which they are interested are already in a data base, and if so, where. Users can find out what relationships have been defined among data items.

"Users," in these examples, can be interpreted broadly. The term can apply to managers and other members of departments of the organization, company executives, staff members, auditors (both internal and external), as well as system analysts and programmers.

Field experience with dictionaries has shown that the on-line query capability is almost mandatory for performing the above services. The problem is, the definitions change. It can be quite expensive to order a printout of a large portion of the dictionary just to find a recently changed definition. And if such a printout is not ordered, but instead some existing printout is consulted, there is no assurance that the information found is up to date.

This access to the dictionary raises the problems of security and integrity. Authorized access to both entries in the dictionary and to the data itself in the data base must be carefully defined. Some people will be allowed a read-only access, while others will have update privileges. Unauthorized accesses must be inhibited. We only mention these problems here; a discussion of them is beyond the scope of this report.

SELECTING A DATA DICTIONARY

The discussion so far hopefully has indicated some of the major features of data dictionaries that many users and potential users desire. These include the three types of bridges, effective security, integrity, and enforcement mechanisms, ability to support system development as well as daily operations, and so on.

The point to be made here is that *no* current data dictionary has all of the desired features that we have discussed. In fact, it is not at all unusual for a company to begin investigating data dictionaries and find that all existing packages fall far short of its desires. The people making the study might then recommend to management that they be allowed to develop a much more desirable dictionary in-house.

Our advice can be expressed in three words: *don't do it.*

To illustrate the problems that such a course of action can encounter, consider the experience of one company we talked to. This company had a staff member investigate dictionaries on the market; no "suitable" package was found, and a project to build a dictionary in-house was authorized.

The initial version of the in-house dictionary was built. It did work and it did provide services that were not available on commercial packages. In fact, some of those services still are not available on any commercial package. To help recoup some of the investment in the package, the company offered it for sale to others, and some sales were made. And then calamity struck. The two key people who had developed the dictionary left the company.

The company at that time was undergoing a big expansion in its data processing staff, to handle a much increased volume of system development. No one else was available with the interest and background to handle the dictionary project. The sales of the package were cancelled because there was no one available to help install the package. Some of the promised enhancements to the package, upon which the company was depending, could not be accomplished. So the project was officially stopped and the package was withdrawn from the market.

This company is still using a very limited portion of the dictionary, but is now in the process of selecting a commercial package. Staff members still like the concepts upon which the in-house package was designed. But it turned out just not to be feasible to build, maintain, enhance, sell, and support the in-house package in the face of staff turnover and the other demands on the data processing department.

We might go even further with our recommendation: *do not even try to modify a commercial dictionary to better meet your needs.* Much the same types of problems as those just discussed can arise when you try to maintain and enhance a modified package. It is much better to work through the users' group for the package you select, to find others interested in the same enhancements and to encourage the supplier to make those enhancements.

What if you have other than IBM equipment? Some of the dictionaries on the market have been designed to run only on IBM equipment. Some others, while they might run on other brands of

equipment, so far have been implemented only for IBM equipment. With non-IBM equipment, the freedom of choice is very limited. There are at least two dictionaries on the market that have been written in ANS COBOL (DATA CATALOGUE and PRIDE-Logik) so as to run on other makes of equipment. In any case, it would be more desirable to try to get the supplier to adapt the package for your type of equipment (almost certainly at extra cost) than to try to do the job in-house.

A preferred approach

Early in this report, we gave a number of factors involved in the decision to install a data dictionary. We also indicated that it was not unusual for companies to decide on some of these factors in a quick, "practical" manner. These "practical" decisions might be: install a data dictionary to support existing DBMS applications only; purchase the dictionary rather than build it; and it is immaterial whether it depends on the DBMS or not.

It seems to us that, if you are considering installing a data dictionary, it should be recognized that sooner or later you will want your dictionary to cover *all* data and process definitions, not just for the mechanized portions of systems. This may or may not be a near-term goal for you. But if you can at least make progress toward this goal as a byproduct of your installation of a dictionary, that is all to the good.

So we suggest that you lay out a longer term goal, for the use of a data dictionary, and then lay out a program for reaching that goal by a series of stand-alone projects. Someday data dictionaries will be just as important as data base management systems. It is going to take a good number of years for any organization to use a dictionary really effectively.

For instance, your first project might be to get data base data definitions cleaned up and under control. Once that has been done, your next project might be to get conventional file data definitions similarly cleaned up and under control. If the same data items appear both in the data bases and in the conventional files, this may mean some further clean up for those items. Your next project may be to use the data dictionary to support system development, in a manner similar to what Marathon Oil is doing or to what Gane describes in his paper. These, of course, are only examples of a series of projects for getting data definitions cleaned up and under control.

With this series of projects in mind, you would be in a position to select your data dictionary. Before making this selection, we suggest that you read all of the references listed at the end of this report. By all means, obtain and read Lefkovits' book (Reference 1). He discusses the six leading data dictionaries currently on the U.S. market, with one chapter for each. These dictionaries are: Arthur Andersen's LEXICON, Cincom Systems' Data Dictionary, IBM's DB/DC Data Dictionary, MSP's DATAMANAGER, Synergetic's DATA CATALOGUE, and UCC 10. After describing these in some detail, in a standard format to make comparison easier, he analyzes and evaluates the six packages. There are substantial differences among these packages, but as Lefkovits says, there is no one best package. The "best" of the six for any particular company depends on that company's needs.

The two evaluation checklists (Reference 7) should also prove very helpful in making the selection of a data dictionary. Each of these seems to pose questions that emphasize the strong points of its respective dictionary. But still, each raises a good number of valid points that ought to be considered. And if the two were used together, that would tend to remove some of the product bias that exists.

But the selection should be made, we think, with both the current and the future needs in mind, not just the current needs.

Conclusion

Based on the growing sales of the dictionary packages on the market, it would appear that a trend toward the use of data dictionaries is underway. The task of effectively installing and using a dictionary is not an easy one. The decisions of what to do and what not to do should not be made casually.

As we pointed out in this report, the data administration function and the data dictionary system are both overhead items. Further, they can entail a good amount of expense and effort. So both are quite vulnerable to budget cuts, company politics, and arguments to bypass them for the sake of expediency.

But the data administration function and the data dictionary system together *can* provide an effective way to clean up a company's data and

process definitions and to keep them clean in the future.

We have listed, earlier in this report, a number of the benefits that are offered by a data dictionary, as well as some problems it can cause. The major benefit is hard to evaluate in monetary terms. A dictionary provides a means for greatly reducing undesired redundancies and inconsistencies in the data definitions. With these undesirables reduced, both the operating people and the management of the organization should have fewer misunderstandings, fewer poor decisions that are based on misinterpreted data, and so on.

In addition, we listed a number of more tangible benefits. System development times and costs should be reduced, because of the centrally maintained data definitions. Communication between the development staff, users, management, auditors, etc. is enhanced, leading to fewer errors and false starts in system development. Inputs for program data divisions and DBMS directories can be automatically generated. And there are other benefits which need not be repeated here.

It is worth repeating as a final thought, though, the three conditions that Lefkovits feels must be met in order for an organization to achieve the benefits from a data dictionary system. These are: *a good degree of commitment* to the data dictionary system, on the part of management, users, and data processing personnel; an *effective data administration function* that has the responsibility for and the custody of all data; and an *effective method for planning and introducing change* into information systems.

We think that Lefkovits has summed up the situation pretty well.

REFERENCES

1. Lefkovits, Henry C., *Data Dictionary Systems,* Q.E.D. Information Sciences, Inc. (141 Linden Street, Wellesley, Mass. 02181), 1977, 480 pages, price $85.
2. Leong-Hong, B. and B. Marron, "Technical profile of seven data element dictionary/directory systems," U.S. National Bureau of Standards special publication 500-3, February 1977; order from Superintendent of Documents, Washington, D.C. 20402, SD Cat. No. C13. 10:500-3; price $1.05.
3. Ehrensberger, M., "Data dictionary—More on the impossible dream," *Proceedings of the 1977 National Computer Conference,* AFIPS Press (210 Summit Avenue, Montvale, N.J. 07645), p. 9-11; price $60; microfiche $15.
4. Gane, Chris, "DATAMANAGER in a structured analysis environment," *Proceedings of DUG,* May 1977; order from MSP, Inc. (594 Marrett Road, Lexington, Mass. 02173), price $20.
5. Schussel, George, "The role of the data dictionary," *Datamation* (1801 S. LaCienega Blvd., Los Angeles, Calif. 90035), June 1977, p. 129ff.
6. Adams, D. L., "System & audit aspects of the data dictionary," EDPACS (11250 Roger Bacon Drive, Reston, Virginia 22090), May 1976, p. 1-14; price $5.
7. The following two checklists and evaluation criteria can be obtained from the suppliers indicated:
   a) "Evaluation criteria for data dictionary products," prepared by MSP, Inc. (address above), suppliers of DATAMANAGER.
   b) "Data dictionary/directory evaluation criteria," prepared by M. Bryce & Associates, Inc. (1248 Springfield Pike, Cincinnati, Ohio 45215), suppliers of PRIDE-Logik.
8. Three articles from *Database Journal* (322 St. John Street, London ECIV 4QH, U.K.). price £4.50 each ($15.50 U.S.A.):
   a) Gradwell, D. J. L., "Why data dictionaries?" Vol. 6, No. 2; p. 15-18.
   b) Thomas, D. R., "DATAMANAGER—a free standing data dictionary system," Vol. 6, No. 5; p. 14-17.
   c) Maskell, R., "LEXICON—an established data dictionary system," Vol. 6, No. 7; p. 15-21.
9. Report by Data Dictionary Systems Working Party, British Computer Society, 1977. For more information, write BCS (29 Portland Place, London W1N 4AP, U.K.) or ACM (1133 Avenue of the Americas, New York, N.Y. 10036).

You may have been hearing and seeing the term "software engineering" lately, and perhaps have been wondering just what it means. Yes, the intended meaning is just what the words say—the engineering of software systems. Some programmers tend to resist the concept on the basis that programming is an art and is not properly the subject of an engineering discipline. But other very capable people in the computer field disagree, and are going ahead with the development of an engineering discipline for software. In our next two issues, we will discuss where the field stands today in this emerging new discipline.

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

(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_____