Implementing data management

by DANIEL S. APPLETON
Appleton and Associates
Manhattan Beach, California

"If the organization plans to use database as a technique and a discipline for the long-term development of integrated applications, then there is far more to think about than if database is to be used simply as a sophisticated access method for the development of single, complex applications."

A database administrator
Infosystems 10-10-79

Most businesses go through three distinct phases as they implement computer technology. In the first phase, they automate specific, reasonably well defined processes such as payables, receivables, invoicing and sales. In phase two, old applications are integrated and new applications are added either as integrated processes or as stand alone applications. In the third phase, a company database is established and users are given access to the information stored there.

According to most experts, today, only a few businesses are in phase three, though most are headed there. The vast majority of businesses are in the latter stages of phase one or in the initial stages of phase two. In short, the momentum toward database technology and data management strategies is rapidly mounting.

One of the basic reasons for this movement toward phase three is a fundamental change in how business sees the computer. Phase one companies generally perceive the computer as a device to control certain business processes. Phase three businesses treat the computer as a source of information which can be used to both control business processes and to help make decisions. In phase three companies, computer hardware, i.e., the mainframe or the mini-computer, becomes a means to an end—not an end in itself. Its purpose is to support the company database(s).

The general impression is that the movement from an applications-oriented DP management approach (phase one) to a data-oriented DP management approach (phase three) is more evolutionary than it is revolutionary. This is undoubtedly why very few companies achieve phase three. By treating the migration toward data management as a casual process, companies tend to get bogged down as they enter phase two. They end up only partially implementing data management. This they achieve by token changes in the tools they use (e.g., a database management system, a data dictionary or some "user friendly" programming language) or the structure they employ (e.g., database administration, programmer teams). Gradual implementation of changes in tools and structures tends to result in poor implementation of data management. This can be read as phase one management using phase three tools and structures. It should not be misinterpreted as phase three.

The movement to data management should be treated as revolutionary rather than evolutionary. This does not mean it should be done overnight, in great haste or with massive personnel changes or infusions of money. It does mean that data management should be understood to mean changes in the total approach to automation. Not only will tools change—requiring significant investments, retraining and software conversions—but structures will change as will methods of doing things. Structured changes, as we shall see, will affect more than just DP—the user will also be affected. And, as far as methods, they should be the first to change. Among them, probably the most important method to change is the approach used for designing and implementing software. This method is often called software engineering. The revolution which will result in phase three must be felt in all three of the major dimensions of the problem: DP tools, structures and methods, especially in software engineering.

AT THE HEART OF DATA MANAGEMENT IS DATABASE

Before examining each of the three dimensions of the data management problem, we must examine data management itself. The roots of data management are buried deep in the concept we know as database. In the early days, database was just a set of computer software called a Data Base Management System (DBMS). Its purpose was simply to maximize the accessibility of data stored in files on the computer. Subsequently, "database" developed a broader meaning. It came to be an approach to data processing management which rivaled the conventional approach of applications management. It is this concept of database which evolved to become “data management.”

To fully comprehend what data management has come to mean in today’s world of data processing, we must coin a new word: IRRIASSPA, pronounced ear-é-asp’-á. This word is actually an acronym for the nine major attributes of
From the collection of the Computer History Museum (www.computerhistory.org)

First and foremost, DP must overcome the tremendous inertia behind the traditional, deterministic approach to software engineering. This traditional approach will not work in a data management environment. What is worse, it inhibits the whole development of data management. Software engineering is the bedrock of data processing. It is the takeaway. It sets the whole pace. If the methodology is not geared to produce software which is sympatico with data management strategies (IRRIASSPA), the result will be highly dissatisfying.

In addition to software engineering, the whole structure of data processing must be made compatible with data management. This will affect both DP and the user, and it will affect organization structures, job descriptions, organizational policies and operational procedures.

The final piece to the data management puzzle concerns the tools necessary to attain and retain control over IRRIASSPA. These tools involve both hardware and software, and more and more they involve capabilities in the area of communications. These tools, along with the appropriate database software engineering methodology and a well-designed data management structure, will help data processing departments to attain the real objectives of data management: IRRIASSPA.

Failure to recognize and deal with each of the three dimensions of data management is the single most serious impediment to its implementation. DP managers who expect to gain its benefits while only doing half of the job are only fooling themselves; they are also fooling their employees, their users and their management.

DATABASE SOFTWARE ENGINEERING

Marshall McLuhan became famous for writing: “The Medium is the Message.” This phrase is tremendously important to data management. Software engineering is the process of designing and building database applications. It is singularly important in determining the products which will have to be maintained within the data management environment. Thus, there are several critical aspects to this software engineering methodology which we must consider. One of the most important is that the software engineering methodology must fit snugly within the data management structure. As we shall see, this data management structure contains certain basic operational activities which are ongoing and which are important for both flexibility and control.

Another crucial aspect of database software engineering is that it must make maximum use of data management tools. DBMS indexing capabilities, non-procedural programming languages, data dictionaries and directories, interactive query facilities, data definition languages, on-line update facilities, etc., must be integral parts of the development process, not just afterthoughts of the system design.

The primary target of database software engineering is the database itself. The database has meaning beyond the reports which it generates. These reports are its "kinetic"
energy. What conventional development methodologies ignore is the "potential" energy of the database. Database software engineering must maximize the database's potential by designing the database first, not last as is dictated by traditional structured design methodologies. Database software engineering deals directly with data entities, attributes and relations. It does not deal obliquely with issues of database content and structure, looking at them through the haze of arbitrary, temporary output requirements. This is the conventional approach: freeze the user's reporting require-
tical analyses about the database structure, and these can
users so that they can see how the database reacts to various
performance and tuning. This tuning concept must involve
and the relational structure of the database.

and computer software facilities can be used to build statisti­
cal analyses and the computer applications systems are so inefficient, and why
so much time is spent in design and development. Design
changes tend to reek havoc with efficiency.
The ideal database software engineering methodology
probably contains six steps (see Figure 2). The first two steps
are specifically geared to requirements analysis. The first is
an Operational Audit specifically to reveal the “as is” situa­tion and problems. The second step is Conceptual Design.
This step defines the database scope and the “service func­
tions” (applications) which the database will probably sup­
port. This Conceptual Design stage is not in infinite detail,
but it should be fairly specific in terms of the data scope.
The third step of the database software engineering meth­
ology is “Database Definition.” Here the systems ana­
lysts and the users define the database, define the data structures, normalize the database and es­

tablish a prototype, actually installing the database on the
computer. This last step is relatively easily to accomplish,
especially with some of the new tools available to DP today.
The fourth step of the development process is “Heuristic Analysis.” (Heuristic means “trial and error.”) In Heuristic Analysis users should interact with the prototype database
(on-line if possible) in order to finalize its structure and con­
tent. Also, in Heuristic Analysis it is necessary to build the
input structures required to support the database.
The fifth step of the database software engineering meth­
ology is to define both the standard and ad hoc reporting
requirements of the database and to build and implement an
output structure. We might call this step Environmental Test.
The final phase of development must be Performance Re­
view. In this phase, DP must make evaluations necessary
for final database tuning. This step closes the development loop because it can be directly related to the original step of Operational Audit.

THE DATA MANAGEMENT STRUCTURE

The concept of a data management structure may be hard
to grasp. DP managers are used to dealing with using or­
izations (composed of users) and DP departments (of
analysts, programmers and the like). The traditional rela­tion­ship is that between the “users” and the “used.” Little
has been done organizationally to change this relationship,
though major changes are required for data management.
These changes assault traditional DP concepts, and they
affect our perceptions about things as fundamental to data processing as documentation techniques, programming stra­
tegies, systems responsibility structures, and system design content.
Data management is a much more participative environ-
Implementing Data Management

1. New institutions like database administration, input administration, output administration and information resource administration are inevitable. These new institutions are starting to replace the old “adversary” relationships created by conventional data processing management strategies, and they are beginning to fill in some of the gaps between “those that serve” and “those that are served.”

Database administration is already accepted as an indispensable part of data management. However, there is no generally accepted definition of what a database administrator does; indeed, what he does today is to a large extent dependent upon his tools and his politics.

The population of database administrators is very small at this point in time, and the vast majority of those that are available have specialized in specific DBMS software. As a result, they tend to be more technical than not. This technical orientation of database administration is a reflection of its degree of maturity.

A closer look at the DBA function shows that while its description can range from very technical to very abstract, at its heart are four attributes of our data management environment: independence, shareability, relatability, and non-redundancy. If the DBA function does nothing else, it must insure that those four main objectives of IRRIASSPA are in fact achieved. Of course, it would be natural to add “administration” to the functional responsibilities of database administration, thus giving it five major responsibilities with IRRIASSPA. However in doing so, we expand its role beyond that of the technical management of databases into one which has more political overtones. In other words, we are raising it up organizationally. It is of real concern whether any individual could competently handle such a broad spectrum of responsibilities.

Data management is more than just four or five of the aspects of IRRIASSPA. It is all of them. Thus, we must either give our DBA function all of the responsibility for data management, or we must somehow supplement it. In other words, we must expand the data management structure beyond the DBA himself, or we must turn it all over to him. We have already seen that the latter may well be impractical. So, we must explore the former.

An examination of data processing from the functional perspective (i.e., database control, input and output) gives the clues needed for structuring data management. Figure 3 shows the resource requirements of these three critical DP functions. We can see that the demands of the output function are exactly the reverse of those for data storage and processing. Dynamics and responsiveness are low for the database while they are high for the output structure; quality requirements are relatively low for output while, along with technical expertise, they are high for data storage and processing. Input, as we can see, is also unique in its demands on resources. The lesson is that each of the three main DP functions places unique enough requirements to warrant its own administrative structure, with its own policies, procedures, job descriptions, budgets, objectives, technologies, etc.

Figure 4 is one possibility for this DP structure. In accordance with IRRIASSPA, we can allocate I,R,R and S to the DBA. We can give the I of data Integrity to the input administrator. While we’re at it, we might as well give the
The input administrator must also control production sequencing and the scheduling and updating, and he must have some responsibilities for database "S"ecurity as it applies to updating activities. Additionally, the input administrator has responsibility for edits, diagnostics, format checks and all other facilities necessary to maximize the quality of data in the database while minimizing the cost of getting it in there.

Since we have given the database administration functions and the input administration functions to various adminis-
Implementing Data Management

Figure 4—Data management structure.

From the collection of the Computer History Museum (www.computerhistory.org)
DP/USER structure undergoes a metamorphosis. Users take on responsibilities they have never had in areas such as programming, documentation, maintenance, etc., and the data processors, while giving up some of their traditional prerogatives, undertake some new and more important ones.

There are other dimensions to the data management structure which, while less elemental, are still unique to data management. These affect planning, requirements analysis, database maintenance, etc. All of these functions look and act different in a data management environment. To leave them untouched is to leave the roots of future problems buried deep within the organization.

DATA MANAGEMENT TOOLS

We come now to the third and, in many ways, the most familiar of the three dimensions of data management: data management tools. As little as five years ago, these tools were few and far between. Limited to so called Data Base Management Systems (DBMS), they addressed very few of the IRRIASSPA attributes. Today, however, DBMS software is extremely elaborate; in fact, it is so elaborate that it is misleading to view it as just DBMS software. To be clear about its capabilities, we must look at its component parts.

For the sake of simplicity and understanding, we can develop six fundamental types of data management software tools. These include:

- Data Dictionaries/Directories (DD/D’s)
- Database Definition Languages (DDL’s)
- Database Manipulation Languages (DML’s)
- Database Query Languages (DQL’s)
- Database Control.
- Data Communications (DC) Facilities

Before briefly describing each type of software tool, I will try to erase any confusion which might exist between these facilities and what we conventionally refer to as a Data Base Management System (DBMS). Depending on the DBMS vendor, any given commercial DBMS may be composed of all six of these facilities, or only one or two of them. Usually, the more elaborate DBMS’s have all of these facilities, but they are individually priced or priced in groups. Less elaborate DBMS’s (such as those traditionally found on minicomputers) have only some of these facilities. Further, if compared facility by facility, some DBMS’s are better in some than are their competitors and vice versa. I think it is a fair statement to say that no DBMS surpasses all of its competitors in all areas.

To effectively deal with IRRIASSPA, all of these facilities are mandatory. Thus, to properly evaluate DBMS software tools, the DP department and the users should perform a careful analysis of their IRRIASSPA needs. Only from this vantage point can they intelligently evaluate the various software offerings and select its software tools.

At first blush, DP management may feel that one vendor should provide all of these database software needs; however, if they assume that this is necessary, they are mistaken. Further, they should not necessarily limit themselves to only one tool for each area. Multiple software tools, including multiple DDL’s, DML’s, DQL’s, DD/D’s, etc., are advisable under some conditions, depending, of course, upon data management’s objectives, IRRIASSPA strategies and company concerns.

Let’s look more closely at each class of software tool.

Data dictionaries/directories (DD/D’s)

There are many approaches to DD/D’s. Data dictionaries are most simply a list of all of the elements contained in the database and the relationships which are established among these elements. Usually a data dictionary describes each element within the database, its synonyms, the organization responsible for updating, any specific edits which are performed with regard to that data element, its security requirements and a description of what data element means. Data dictionaries also include the data element format as well as its character composition.

Usually supplementing the data dictionary is the data directory. The main job of the data directory is to describe how each individual data element is used, and where it is used. The data directory might point to various computer programs which use a data element; it might point to various job streams; or it might point to various input documents or computer reports. Comprehensive data directories might cover all of these areas in attempt to give a complete description of the utilization of every data element within the data dictionary.

In selecting a DD/D, it is important to determine whether it is integrated or not integrated with the DDL and the DML. It is also important whether the host DBMS DDL was used in constructing the DD/D. In the final analysis, however, the best DD/D’s are those which the DP department feels it can maintain. DD/D’s are not only difficult to establish; they are difficult to keep current—especially the directory features. An elaborate automated DD/D which can’t be maintained is just as useless as is a manual one.

The DD/D is extremely critical in the data management environment. It has a direct affect on every aspect of IRRIASSPA. In fact, in selecting the proper DD/D, data processing management must evaluate each of its IRRIASSPA requirements and assure alignment between the capabilities of the DD/D and the requirements of each IRRIASSPA component.

Data definition languages (DDL’s)

The term Data Definition Language was established by the Committee On Data Systems and Languages (CODASYL). It is used in conjunction with the concepts of database “schema” and “subschema.” Basically, a schema is a description of a complete logical database, and a subschema is the description of a subset of that database which is utilized by an individual computer program. According to CODASYL, the main function of the DDL is to describe the content and structure of both the schema and the subschema.
All DBMS's have DDL's. Some of these DDL's are oriented toward describing data elements within hierarchies, and some of them are oriented toward describing networks. Some DDL's are much more elaborate than others, providing facilities for describing very complex relationships among data elements. Obviously, DDL's affect many of the aspects of IRRIASSPA. For example, the quality of a DDL is in large measure derived by evaluating its ability to produce data independence. Some DDL's are less efficient in establishing data independence than others. Some DDL's are more geared to reducing data redundancy in describing certain types of database structures than are other DDL's. Another important feature of DDL's is whether or not they establish relationships among data elements with pointers embedded in the database or with pointers which are stored in indexes to the database. No DDL is best for all conditions. Some DDL's are easier to use than others, and some are more flexible. Selection of the proper DDL is of critical importance to data management. (No DDL is capable of describing a "structureless" database. This is the ultimate objective of the "relational" DDL. However, relational DDL's are not practical given the current state of hardware and software technology. Perhaps they will be in the future.)

Data manipulation languages (DML's)

Data Manipulation Languages (another CODASYL term) are used to bind together normal procedural languages such as COBOL and FORTRAN with the capabilities of the Data Definition Language. There are many different methods for binding the DDL to procedural languages, and therefore there are many different techniques that are used within Data Manipulation Languages. Most often, DML's are used by procedural language programmers as if they were calls to the DDL. These calls treat the DDL as if it were a set of sub-routines.

Facilities of the DML directly affect programmer productivity. Where the DDL is usually the province of the Database Administrator, the DML is usually the province of the procedural language programmer. The DDL is used by the DBA to provide subschema of the database for manipulation by the procedural language programmer with his COBOL or FORTRAN program. Thus, DDL's and DML's are often closely related. The functionality reserved for each usually describes the technical responsibilities which must be assumed by either the Database Administrator or the procedural language programmer. Some DML's require very little effort on the part of procedural language programmers. Other DML's force the programmer to write complex programs required to navigate through the database. Where a DML can require extensive navigation on the part of the procedural language programmer, a good DDL can act as his automatic pilot, thereby improving his productivity.

Database query languages (DQL's)

Database Query Languages come in many forms with many different capabilities. Most often, they are interactive, on-line in nature; however, there are some very powerful "batch" DQL's. Database Definition Languages which store pointers in indexes as opposed to in the data file (often called "inverted list processors") often provide extremely powerful DQL facilities. These facilities allow direct interaction with the total database schema, and they allow parametric and Boolean search strategies to be used for either data retrieval or update.

DQL's are often referred to as "end user facilities." They are the attributes of the DBMS software which are said to make it "user friendly." Ultimately, DQL capabilities will turn over to users functions which traditionally have belonged to data processors.

While the DDL structure will remain under the control of the data processing department, the DQL structure will move under the control of using departments. The DML activities will, in all probability, remain twilight zone in nature, being shared by both data processors and users who are technically capable. In referring back to our data management structure, DBA's will use the DDL, Input Administrators and Output Administrators will primarily use the DML's, and the Information Resource Administrators will primarily use the DQL's.

Database utilities

Every data processing organization requires specific software utilities to improve its capability for handling certain aspects of IRRIASSPA. These utilities, when coupled with the capabilities of the four prior software tools, will serve to fill in the gaps and equalize the balance among the various components of IRRIASSPA.

Following is a list of the various types of Database Utility software which might be obtained by a company.

- Backup/recovery
- Password security
- Incription/description software
- Image management software (text, graphics)
- Audit trail utilities
- Database tuning utilities
- Database development aids
- Database reordering and reorganizing aids
- Database sizing and responsiveness aids

All organizations need all of these facilities. However, it is important that these tools be part of any data management environment. Their interrelationships with the various components of IRRIASSPA are self-explanatory.

Data communications (DC) facilities

It is not uncommon to hear data management referred to with the phrase DB/DC (Data Base/Data Communications). This terminology was introduced by IBM in the early 1970's to describe their thrust toward on-line database capabilities. In the early days of database, IBM was the only organization
to foresee that database effectiveness was a function of the on-line accessibility of the database to the user. Therefore when IBM introduced its concept of database, they did it using additional software facilities specifically intended to provide on-line accessibility to the database. It is IBM, therefore, which actually introduced the concept of data communications (DC) as an integral part of data management technology.

What originally started out as DC software intended to provide on-line accessibility to databases has now evolved to the point of providing the capability for distributing databases. Ultimately, these same software facilities provide the capability for establishing network DBMS facilities.

Data Communications facilities come in all shapes and sizes. They are used to provide accessibility to DDL's, DML's and DQL's as well as to some of the Database Utility tools. Data Communications software can be provided by either a DBMS vendor or a mainframe vendor, and it can be used for many things beyond just accessing databases. DC strategies are critical in determining the ultimate effectiveness of the data management environment. They affect all areas of IRRIASSPA. As a result, data processing management must be careful in defining its DC architecture, making sure that the facilities adopted for Data Communications augment their strategic data management plans. When building the overall data management architecture, Data Communications facilities can be one of the most critical aspects in determining future capabilities, constraints, costs and productivity.

As I stated at the outset of this section, my topology of data management software tools is relatively arbitrary. I do not even claim that it is complete. However, I do think that it is useful in describing the types of facilities which one must consider in establishing a data management environment. To ignore any of these areas is to create downstream surprises and problems. I am also trying to get across the point that there is no single one best configuration for data management software and that no DP department should depend totally on one vendor to provide all of its resources.

CONCLUSION

I have tried to describe the world of data management in such a way that those who desire to enter it can see its overall implications. I admit to over-simplification, but hopefully, I have raised just as many questions as I have answered. That was my objective.

In this effort, I introduced several brain teasers. First, I introduced the concept of IRRIASSPA. This was to focus the discussion and to define data management in terms meaningful to its implementation. Second, I defined three dimensions of data management, all of which affect IRRIASSPA. In so doing, I redefined two well-known dimensions (data management structures and data management tools) so that they are now more meaningful. And finally, I introduced the crucial third dimension: Database Software Engineering, which, in my opinion, is of equal importance to the other two dimensions within the overall concept of data management. In fact, the Database Software Engineering methodology is of so much importance that it probably should be the first step in any effort to convert from a conventional data processing environment to a modern data management environment. Establishing an appropriate Database Software Engineering methodology will establish the foundation for customizing the other two dimensions of data management: data management tools and data management structure. I based this recommendation on my experiences and observations which say that if the inertia in an organization is so strong that the software engineering methodology cannot be changed, that organization will probably never get into the world of Data Management. Instead, what will happen, is that the organization will implement new structures and new tools, but it will end up using them in old ways.
Data Base Design

In the past, data base design has been treated in literature and technology as a "universally" definable endeavor susceptible to some common or singular approach or model. It is now being recognized, especially by data base designers and users, that data base design is not nearly as tractable as it once appeared since its design is much more user- and environment-dependent than previously recognized.

The two sessions in this technical area present a broad range of views and approaches to designing data bases. They cover user experiences with specific data base management systems, and academic and industry research addressing data base design techniques and tools.

One session, Data Base Practicum, co-chaired by Prof. Jeffrey Hoffer and Donna Shepard Rund, surveys user experiences with various commercially available data base management systems and with custom data base design efforts.

The other session, co-chaired by Dr. David Jefferson and Nan Shu, will discuss data base design focusing on new techniques. These include new specification methodologies; representations of "properties" and their relationships; and other languages and tools for data base design. We find that there are key aspects we must examine when we are designing data base systems so that we can optimize the operational usage and realize the cost benefits for the following:

- the industry/company operational and organizational environment;
- the technical environment;
- the user's current needs, wants, uses, and expectations of the system;
- the future, unplanned uses of the system.

The trend in data base design is toward a more formal, structured design approach. Success in this endeavor should result in more flexible and dynamically expandable data base systems.