Richard L. Nolan

Computer data bases: the future is now

In the natural course of events, data bases that allow companywide programming are emerging from mere theory into reality

There has been considerable debate among specialists about what a data base is, if such a thing exists at all; about the elements of information that ought to go into it; about the administrative complexities created by the concept; and about the real value of the concept as the central organizing principle for a company's EDP operations. To the extent that top management has overheard this debate, it has probably become confused and highly skeptical about the whole idea. This article shows that the concept is real, viable, and beneficial. It also shows that EDP evolution is leading companies with significant EDP operations in the direction of a data-base form of information organization. The author's purpose here is to clarify for top management the essential facts about the nature of a data base, its construction and administration, in such a way that top managers can grapple with the notion and guide lower levels of managers, especially in the EDP department, to a standard of operations that is adequate to the company's need for management information. As documentation, the author uses the results of a survey designed to find out what the data-base concept means in strategic and operational terms today.

Mr. Nolan is Associate Professor of Business Administration at the Harvard Business School, where he has taught and done research in management control systems for computer-based activities.

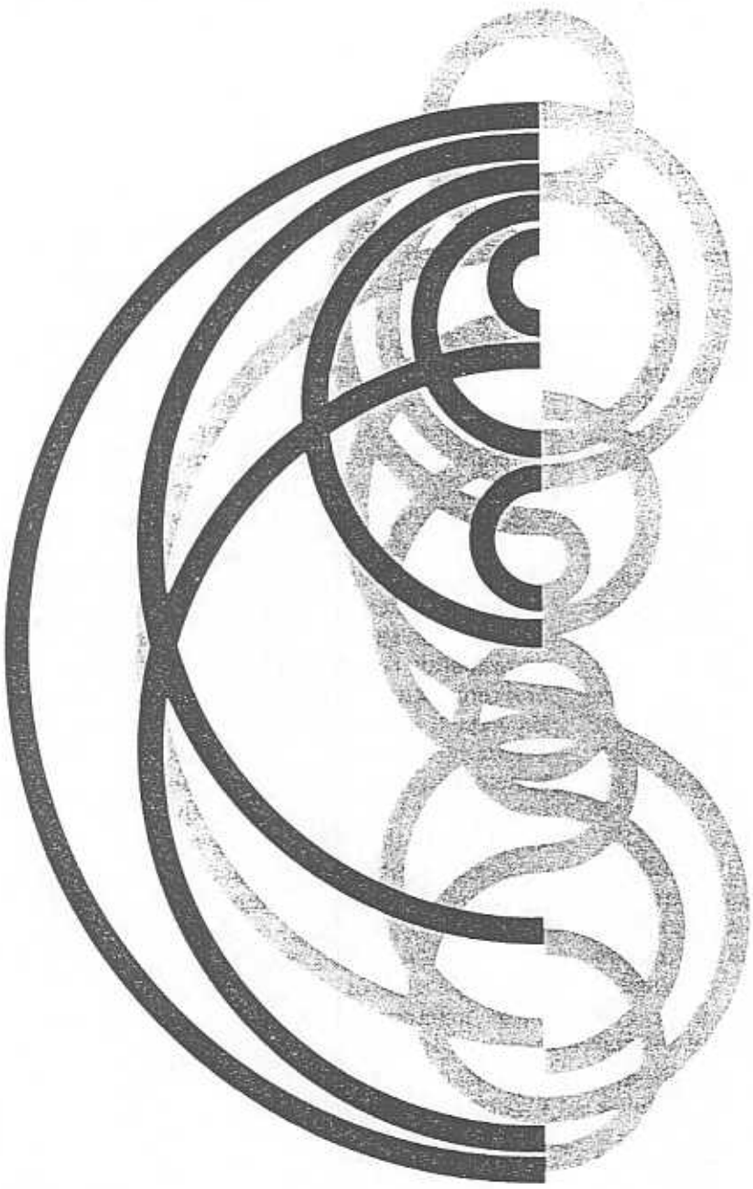
On July 1, two sets of forecasts from the planning department arrived on the desk of the marketing vice president. The first set forecasted sales for the company's existing line of industrial products for the year beginning the following January. The second set forecasted sales over the same period for a new line of industrial products, similar to the company's established line, to be introduced in January.

The company expected that the two lines would complement each other, permitting an in-depth coverage of its markets which would place the company in an extremely strong competitive position. This was particularly important at this juncture; competition had been making inroads into the company's traditional turf, and the higher level of sales that could be expected would give the company some much-needed profits.

With pleasure, the vice president of marketing noted that the forecasts were more promising than he had dared to hope. The forecasting staff had an excellent track record—there was no reason not to take the high projections seriously. So some thought would have to be given to increasing production for both lines over the next months.

Production was really not much of a problem.

Reprinted from HBR September-October 1973, Number 73507



In preliminary talks with manufacturing, the fact had been well established that existing plant and personnel could stretch to produce higher quantities on short notice with relatively little sweat. The forecasts gave adequate indications, too, of just how high a level of production would be needed each month for the succeeding nine months.

Inventory, on the other hand, did present something of a problem. Company sales had distinct regional characteristics; they were made from regional warehouses which the company held on leaseback arrangements. In preparation for the new product line, some provision had already been made to increase the company's regional warehouse facilities, but these high sales forecasts made the marketing vice president wonder whether the expansion had been large enough. It seemed to him that a good many of the regional warehouses might well be severely squeezed—both in sheer physical space and in manpower. With these high projected volumes, he thought, the company could be marketing itself right into a warehousing bottleneck.

Also, the regional variations in sales patterns had not been taken fully into account in making the forecasts, for "technical" reasons—here was

the one "soft" area he had agreed to tolerate in the forecasts. Hence he was not clear on which warehouses would be the worst hit and where there might be some excess capacity in closely adjoining regions.

He thought next of the aggressive marketing and promotion strategies he had just approved, for both product lines, to be carried out in various regions over the next few quarters. If they were as effective as he expected them to be, there *would* be a problem in the warehouses. Perhaps he could cut back on his preferred marketing and sales strategies, but this alternative conflicted with the company's need for increasing sales as quickly as possible.

Definition of a problem

He did some simple arithmetic and then headed for the CEO's office. After outlining the situation in light of the final forecasts from planning, he boiled it down to this: "If things work out the way we expect them to, the Chicago warehouses, at least, will have to operate at four times their capacity for at least three months. That's only one group of warehouses. And we really don't have the cash in hand to contract for additional outside space."

They argued about a number of possibilities, making a few calculations and thinking out various consequences aloud. After an hour's discussion, they concluded that they simply had to have a better picture of the impact of the new marketing strategy on sales of both lines, and they also had to have a better picture of the impact of projected sales on inventory turnover and warehouse crowding. They needed, in short, to pull all three of the threads together.

At this point, the CEO pointed out that parts of the puzzle were already on the computer:

◊ The company had inventory simulation programs which had been developed recently to help adjust inventory policies. Several years' actual data on inventory turnover were also available.

◊ The sales department had a number of forecasting programs designed to provide sales reports and forecast information by region and product.

◊ On behalf of marketing, the forecasting staff had developed a model for market penetration of the new product line, based on the sales of the existing product line, which it was intended to supplement.

So the two men went to see the vice president of computing services and laid the problem out for him. Then they asked, "Can you get us a printout that will tell us what impact marketing and sales are going to have on the warehouses?"

His answer was no. He pointed out that the company had no program for running such a simulation. He also pointed out that while the company had most of the data that such a simulation would require, none of the data were in readily available form. The inventory data had been specially prepared for the inventory simulations and would have to be completely recoded before they could be used for such a radically different purpose. The programs for forecasting sales did provide regional projections, but they had not been adjusted to mesh with the new inventory systems as yet—that development was still some months away. Further, the several years' sales data used by the programs, once again, were specially coded for the sales programs and could not be used in different programs without a massive reorganization of the data.

Finally, he pointed out that their computer system could not handle the sheer volume of data required by a simulation that attempted to combine all the necessary inventory data, sales data, and market-strategy data. At a minimum, more main memory for the computer would be required.

The CEO looked glum: "I'm not worried about the size of the computer. If we need a bigger memory, we'll get our hands on a bigger memory. How long will it take you to clean up the data and write a simulation program that will give us some answers?"

"Nine months, maybe a year," said the EDP manager.

"Because all our data are frozen into these other programs?" the CEO asked.

"That's the main reason," the EDP manager replied.

"That's a hell of a reason," the CEO said and stalked toward the door.

"Of course, we could have done it the other way," the EDP man called after him. "But now what we'd have to do is. . . ."

But the two men were gone.

What's the answer?

The problem in this vignette is not one that has an easy solution. Management asked for a com-

puter simulation that cut across three different departments, and was frustrated primarily because each department's data were locked up into its own applications. Even if time had permitted, the cost of recoding all the data for a cross-departmental simulation would probably have been too high for the company to bear. In other words, the company's own data were a frozen asset—a highly constrained resource, analogous to money which could be used to purchase only one type of asset.

Management requests for such ad hoc processing are increasing. As a consequence, companies are beginning to realize that data are a valuable resource, to be managed like any other basic resource.

What the EDP manager wanted to explain to the CEO, and what the CEO did not wait to hear, is that the company could have been managing its data in such a way that the CEO's request could have been fulfilled.

If the company had maintained all its computer-readable data in a single pool or bank—in a so-called "data base"—and if the company had structured this base of data so that a program for virtually any feasible use could have been run from this data base, then it would have been a matter of sheer expertise and flair for a good, experienced programmer to concoct a program that pulled the desired information together. Further, if the company had been maintaining a data base, its programmers would already have developed the expertise and capability to write such a program, with the aid of a "data-base language," on reasonable notice.

The ability to deal with such ad hoc requests is the special benefit of the data-base approach. It has a more mundane benefit too: in an EDP facility of any size and complexity, it is feasible, and much more efficient over the long haul, to create *any* program—whether big, small, complex, routine, or ad hoc—and to run it from a data base rather than from a lot of separate files of data locked to specific applications.

The truth of this rather strong statement derives from the fact that the data-base approach frees the programmer from the constraint of working over, under, around, and through the structures of separate data files, an expensive fact of life implicit in the traditional approach to EDP operations and planning. With a data base, he need only work with a single structure, that of the base itself.

This one-main-structure feature also makes it easy to decide how data can be obtained and integrated into the base most efficiently and economically—that is, it eases the data-maintenance problem, once the data base has been set up. Considering the extraordinary percentages of EDP budgets that companies allocate to maintenance today—usually over 50%—this benefit is a highly significant one.

Thus the concept of a companywide data base has emerged. It has two key aspects:

1. The data that computer programs use are considered an independent resource in themselves, separate from the computer programs.
2. There is an art and an approach to managing and structuring a company's computer-readable data as a whole, so that they constitute a resource available to the organization for broad-range applications—especially on an ad hoc basis.

Because of its potential benefits, the data-base concept has received much attention recently in the professional press. What I hope to do in this article is explain it in management terms and present some survey results that indicate how the concept is being received and implemented in a small sample of companies.

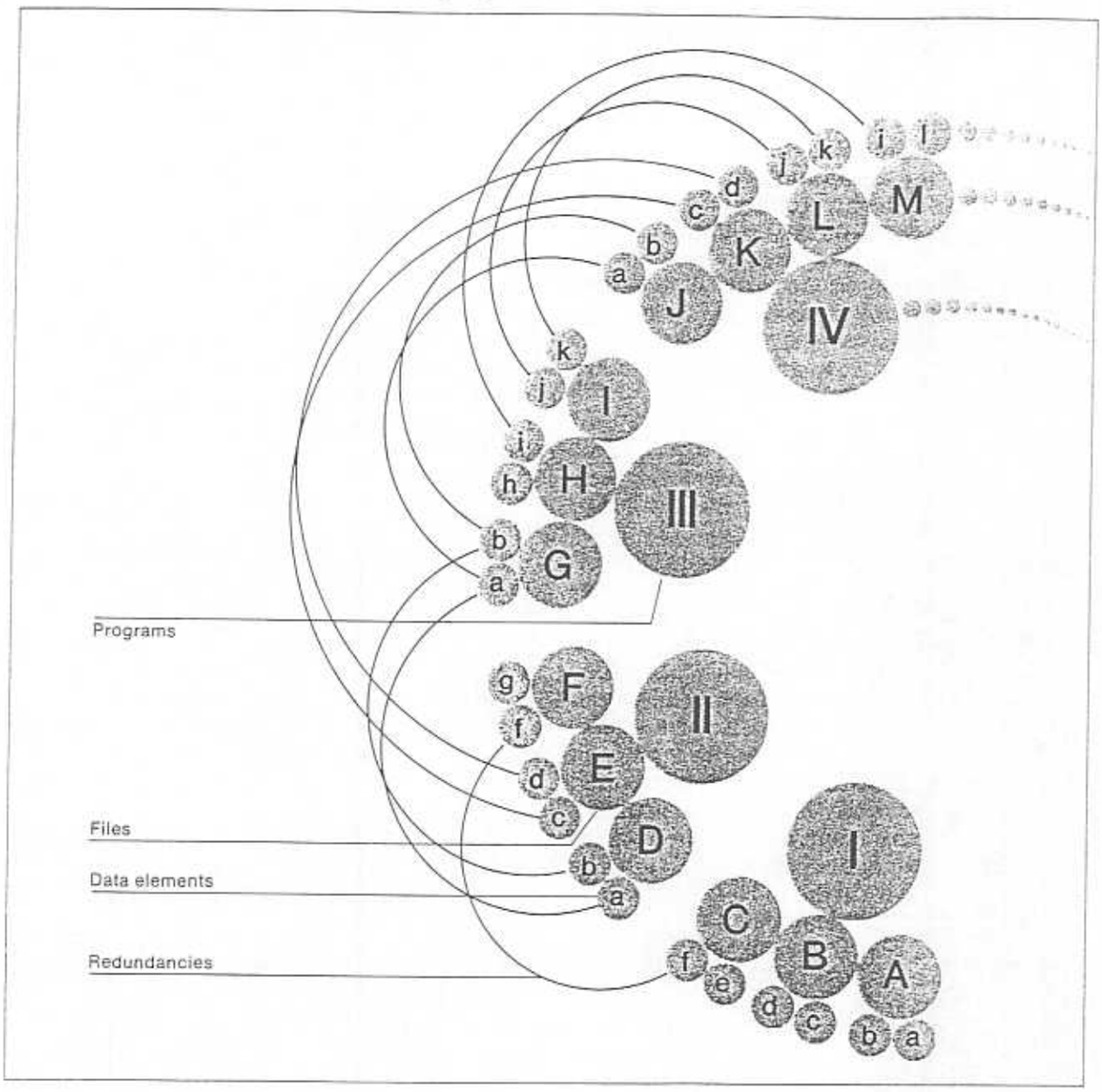
A historical pattern

As the matter now stands, most managements would be hard put to manage—and use—their data to full potential, for reasons that are largely historical. Because of the rapid growth of computer technology, management of data has developed haphazardly and in laggard fashion over the years. A general approach to data management has emerged only very recently, and, consequently, applications have developed discretely from one another in an unintegrated and wasteful fashion. Further, each increase in the complexity and capabilities of computers has brought new generations of applications—but these applications still, for the most part, have been specialized in nature, designed for a specific operational use or for a specialized staff function.

Hence management of data has continued to develop in fragmented fashion and at rather low organizational levels—at a subdepartmental or substaff level.

Today, upper levels of management are seeking information that can be generated only from properly structured, companywide pools that include data from the narrower applications lo-

Exhibit I. The traditional approach to programs and data



cated farther down in the organizational hierarchy. That is, management information today requires that a company have a data base which can be used, in conjunction with broad-range programs, to generate information on a broader and more comprehensive scale than the single, isolated applications of the past could usually do.

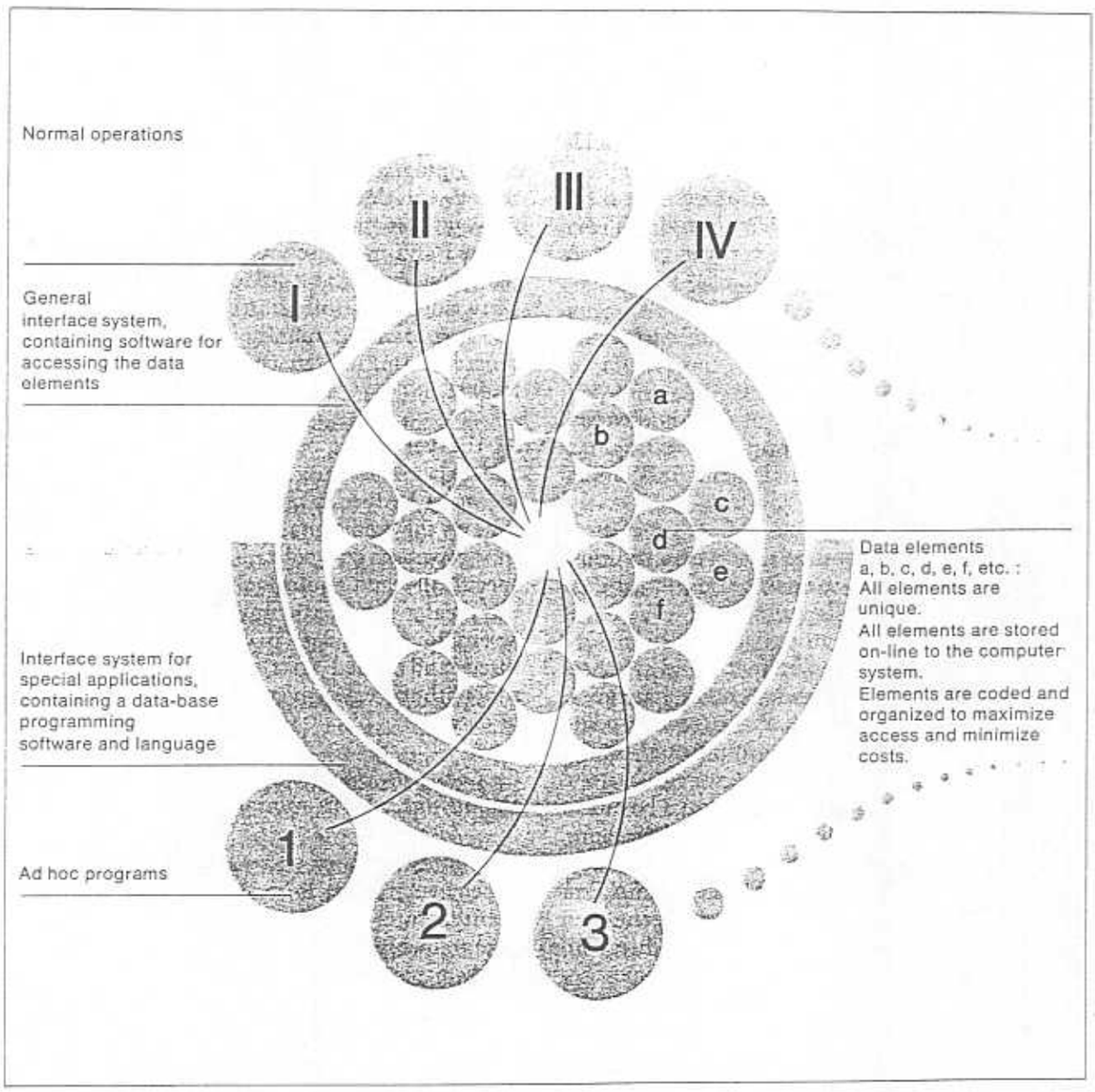
Notwithstanding the new demands, tradition is still strong; indeed, it has barely been challenged. *Exhibit I* represents the traditional way of doing things—collecting and coding data for specific programs and thereby gluing them more or less permanently and exclusively to those programs. In retrospect, this approach has had three significant disadvantages.

1. Files and records have tended to become redundant.

Suppose Company X originally *had* only a single, computer-based system—say, for accounts receivable—which is represented in *Exhibit I* as Program I. (At this point, please pay attention to *Exhibit I* only. I shall explain its companion—*Exhibit II*—subsequently.)

Program I has three data files: A, B, and C.

Exhibit II. The data-base approach to programs and data



File A contains customer records, each consisting of data elements *a* and *b*; *a* might be the customer name and *b* his outstanding balance. Files B and C contain other data elements needed for the accounts receivable program.

Assume that now the company wishes to implement a second program—Program II, as illustrated in *Exhibit I*—with Files D, E, and F comprehending elements *a*, *b*, *c*, *d*, *f*, and *g*. Note that the company already has all these elements, except *g*, on file for Program I. In all probability, however, its programmers coded Files A and B (including all the elements *a*, *b*, *c*, and *d*) expressly for Program I, and hence cannot now use A and B intact for Program II. Thus the programmers have to make a choice:

▽ They can recode A and B so that these files can be used by either Program I or Program II. *But* this would mean rewriting Program I to take account of the recoding.

△ Alternatively, they can build two "new" files, consisting of data from A and B but coded for the special convenience of Program II.

In the past, when faced with this kind of choice, an EDP department has usually just gone ahead

and constructed the two "new" files. Going back over Program I ordinarily seems like too much trouble, so making up the new files seems the easiest way out. It is—*in the short run*.

But in the long run, as the exhibit shows, Company X might easily find itself creating more and more quasi-duplicate files as it adds new programs. For example:

◊ It will need two new versions of File B for Programs II and IV—that is, Files E and K.

◊ It will need three new versions of File A for Programs II, III, and IV—that is, Files D, G, and J.

◊ It will need a new version of File I for Program IV—that is, File L.

And so on. The redundancy of data is obvious. In just this little, highly simplified example, 7 out of 12 (58%) of the data elements in the files are redundant.

Initially, redundancy does not cause a great deal of trouble. As soon as pieces of data must be *updated*, however, it *does* cause a great deal of trouble. In an EDP department of any size, it is virtually impossible to update all the redundant files and reports in systematic and synchronized fashion. Consider what must happen if Company X adds a customer: it must update A, B, D, G, and J, and that would only be the beginning.

Once files, records, and reports have begun to overlap and updating becomes a serious chore, updating procedures begin to sag of their own weight and different parts of the organization begin to receive inconsistent reports generated from files that are in various states of disrepair. In one large company, the inconsistencies between sales reports at the division level and sales reports at the branch level were so extreme that the salesmen began to keep very elaborate *manual* sales records. These two sets of reports were, in fact, generated in large part from redundant files that were updated at different times.

These particular inconsistencies resulted from a mere difference of organizational *level*—that is, the divisional versus the branch level. Severe redundancy problems can arise even more easily when reports from one function must be meshed with reports from another function. For example, there is absolutely no reason to expect that a company's inventory-control report will jibe with its accounting report unless the updating disciplines for the files of both functions are synchronized with each other.

Even slight variations in the data used for the two functional reports can cause glaring inconsistencies:

In a large retail chain whose applications had developed in the traditional fashion, the needs of the business forced management to request the integration of a number of different functional programs and systems. With great effort, the job was done. However, it was done in such a way that many quasi-duplicate files were created and many separate, but essentially similar, programs were patched together. The company suddenly found itself spending 90% of its programming man-hours just keeping the programs running in concert and the files up to date.

At the very least, redundancy spells confusion and expense for any sizable operation. Perhaps its worst feature is that the longer a company follows the traditional pattern and keeps adding new programs and redundant files of data, coded specifically and exclusively for those programs, the greater the task it must face when it finally assembles all its data in a single pool, so structured and coded that new programs can be run without extensive recollection or recoding of data.

2. The traditional approach undercuts or aborts the advances of computer technology.

Computer memory was once a great deal more expensive than it is today. A major computer manufacturer is now predicting that semiconductor technology will reduce the present cost of main memory by many orders of magnitude in the not-too-remote future. Even now the costs of random-access storage have been greatly reduced by the development of extremely large disc devices. Furthermore, new software has introduced new dimensions to computing, dimensions that make possible the more advanced kinds of information systems. For example, virtual-memory techniques allow one to explore, cheaply, relationships between elements in a relatively huge pool of data, not all of which need necessarily be present in literal fact.

Originally, the relatively high cost of on-line storage ("memory," in a rough sense) was a main factor that induced companies to delimit the scope of programming and therewith the amount of data needed during any given run. In effect, this reinforced the practice of creating and maintaining separate files for each appli-

cation in the company's portfolio—companies tended to store no more data than were needed for the run at hand.

Today, however, many companies that have followed the traditional route, but have acquired up-to-date on-line storage systems, find they have the capacity to keep relatively huge amounts of data alive in the system. But their data are still organized and coded along first-generation computer lines—that is, by specific programs. From a rational viewpoint, this is as awkward, expensive, and absurd as keeping modern accounting records wholly in Roman numerals.

3. The traditional approach obstructs upper management's growing demands for applications that require a data base.

The reason for this unfortunate condition is locked into the history of the computer. A brief review of the evolution of computer-based applications runs as follows:

- The computer was first used to replace existing manual functions, primarily within the accounting function.
- Next came the integration of computer-based systems within and between functional areas—this was the cross-functional stage.
- Now cross-functional/interlevel systems are being developed to serve middle and upper-middle management; or, to put it another way, management is now demanding the benefits of computer innovations.

At this third stage, the redundancies and inefficiencies that result from the traditional approach to the management of data become so signal and so extensive that applications can be adequate only if they are developed in such a manner that specific programs are separate from the data. That is, the whole body of a company's data must be structured into a flexible data base.

A modern concept

As *Exhibit II* shows, the data-base concept structures EDP activity in such a way that all of a company's computer-readable data are merged in a single pool, which is used to run both routine programs and programs written in response to ad hoc requests. Note that no files appear in this exhibit—the base of data elements constitutes the general file for the company, and specific files are by and large unnecessary. Note also that two additional software systems are in evidence here which were not in evidence in *Exhibit I:*

The data-base interface system—this enables a specialist data-base programmer to organize and structure the data elements in a manner that minimizes or eliminates redundancy and optimizes the economic costs of data storage and accessibility.

The interface system for special programming—this includes a high-level programming language especially designed for manipulating data elements contained in the data base, solving problems, and producing reports. To write an ad hoc program, the programmer works successively through the interface for special applications and the general interface system to the data base itself.

Comparing *Exhibits I* and *II*, one can see an immense contrast between the traditional concept and the data-base concept, both theoretical and practical. If the company described in the vignette with which I opened this article had had a working data base, the CEO might merely have asked his EDP manager to set a programmer to work on an ad hoc program. No question of data availability would have arisen; the only variable in the case would have been the time required to actually write the program, and this time might only have been a matter of hours.

One can more fully appreciate the contrast if he looks forward to the fourth stage of development in computer applications—applications that senior executives will use in corporate management. This development will most likely emerge from the union of the data-base concept and the corporation-model concept; and while this union is still but a gleam in the specialist's eye, the company that adjusts its EDP policies to the data-base concept now will enjoy a very significant advantage over the company that pursues the traditional patterns until the day of reckoning has actually arrived. (Just *how* a company should pursue this adjustment is a problem I shall consider later.)

Since much of the computer technology necessary to implement the data-base concept exists and the rest of the technology is being developed rapidly, a strong case for adopting the data-base approach can now be made. Yet, in operational terms, the concept is still novel. To what degree is it being used? What are the issues and problems involved in implementing it? By

Exhibit III. Evolutionary stages of the data base

	Company 1—Manufacturing	Company 2—Public Utilities	Company 3—Banking	Company 4—Manufacturing
Evolutionary stage	Low	Low	Low-medium	Medium
Data-base concept	All computer-readable data	All computer-readable data	Tape and disc files	Disc files
Data-base structure	Individual applications	Individual applications	Operational, by products	Operational, by functions
Degree of integration across level	Minimal	Minimal	Low	High
Degree of integration between levels	Minimal	Minimal	Minimal	Minimal
Data-base strategy	Brute force	Brute force	Brute force	Piggyback
Decision maker for data-base contents and designs	Systems analyst	Systems analyst	Steering committee, with an administrative position planned	Systems analyst, with an administrative position planned
Personnel with direct data-base access	Programmers	Programmers	Programmers	Programmers

what strategies can a company work toward a data base? And what benefits can we realistically expect from it?

An interview study

To answer these questions I administered a pattern interview to the data-processing managers of ten companies in six diverse industries. The questions permitted unrestricted responses, and hence the information these managers provided (summarized in *Exhibit III*) is not as clear-cut as one might wish. However, it is informative. It provides some operational perspective from EDP managers on the following topics:

- Current impressions of what a data base should be and do.
- Approaches to structuring and organizing a data base.
- Strategies for building a data-base system.
- The assignment of responsibility for the base's scope and contents—that is, data-base administration.
- The role of the data-base administrator.
- Access and security.
- Organizational and technical problems associated with the data-base concept.

The opinions expressed on these topics varied considerably among the EDP managers I interviewed. By and large, a given manager's opinions reflected the particular stage his company had reached in the evolutionary progression toward full use of the data-base concept.

For the reader's convenience I have organized

Company 5—Manufacturing	Company 6—Insurance	Company 7—Manufacturing	Company 8—Manufacturing	Company 9—Wholesale/Retail Food	Company 10—Manufacturing
Medium	Medium	Medium-advanced	Medium-advanced	Advanced	Advanced
All computer-readable data	Shared random-access files	Shared random-access files	Shared random-access files	All computer-readable data	Shared random-access files
Individual applications	Operational, by products	Operational, by functions	Operational, by functions	By key tasks in accounting and distribution	By key tasks in planning and manufacturing
High	Medium	High	Medium	High	High
Minimal	Minimal	Minimal	Low	Medium	Medium
Key task	Piggyback	Piggyback	Piggyback	Key task	Key task
Systems analyst	Data-processing manager	Data-processing manager, with an administrative position planned	Steering committee, with an existing administrative position	Steering committee, with an existing administrative position	Data-base administrator
Programmers	Programmers	Programmers	Programmers, analysts	Programmers, analysts	Programmers, analysts

the material in *Exhibit III* in evolutionary sequence. One manufacturing company, Company 1, at the far left, has barely begun to understand and use the concept; in Company 10, at the far right, one finds a fairly sophisticated example of a data base in operation.

Let me now discuss the topics listed, one by one, with some attention to the way the data base shapes up at various stages of its maturity.

Nature of the data base

First of all, I found a certain amount of confusion about what "data base" means. My open-ended question, "What is the data base in your company?" usually brought first a puzzled expression to the manager's face, and then a request for clarification. I answered that I wanted a statement on how he views his company's data base, if, indeed, he views it at all.

Responses ranged all over the lot. Some managers included all the computer-readable data in their company. Others defined the base more narrowly—for example, as including only the random-access disc files used for routine reporting and analysis.

The common thread in the responses was "computer-readable." Since all the interviewees were data-processing managers, this common thread is not surprising. But, obviously, the great majority of an organization's data are non-computer-readable; they are maintained in file cabinets as well as in the minds of management.

Although more and more data are being put into computer-readable form, as the technology improves and makes more sophisticated com-

puter-based applications both feasible and economic, much of the literature on data bases falsely assumes that companies have already translated all the data needed for these applications into machine-readable terms. This simply has not yet happened—indeed, most companies have not even begun to collect the data needed for these applications, in machine-readable form or otherwise.

In general, the more advanced a company's use of the data-base approach, the less naïve and more realistic the manager's definition of what the base ought to contain—for example, "shared random-access files used for [periodic] production programs and ad hoc management requests." Such a definition reflects the two key characteristics of the data base: (a) sharing data between programs, and (b) structuring data so that ad hoc management requests can be served. As *Exhibit III* shows, the more advanced companies conceive their data bases in this light.

One data-processing manager articulated the criterion of responsiveness to ad hoc management requests especially well. He said that his company will realize the data-base approach fully when he has incorporated the technology that will permit him to respond to any reasonable request by management for reporting or analysis within one day, and without undue degradation of his continuing data processing. He further described a reasonable request as one that draws on existing computer-readable data.

Structure & integration

Companies 1, 2, and 5 in *Exhibit III* viewed their data base as structured under the single criterion of individual applications, in the fully traditional manner. Companies 1 and 2 had a minimal degree of cross-functional integration—that is, sharing data between such functional applications as manufacturing and accounting.

Company 5, in the middle of the spectrum, had a high degree of cross-functional integration, as *Exhibit III* shows. In my interview with its EDP manager I had been led to believe that cross-functional integration was minimal. However, further discussion with their lead systems analyst pointed to high integration. With more probing, I found out that this man had taken it on himself to design files to accommodate sharing between programs. He was quite active in the EDP professional societies and expressed strong feelings that this was the "right" approach.

Still, all three of these companies had minimal sharing of data between levels of management. In fact, there were very few programs developed for management in any of the three companies.

In addition to Company 5, four other companies (4, 7, 9, and 10) indicated a high degree of cross-functional integration of their data bases, and they had very well-developed computer applications in the operations aspects of their businesses as well. But I should note a significant difference between Companies 4 and 7, and Companies 9 and 10:

▽ Companies 4 and 7 had well-developed applications for general operational activities—accounting, distribution, and inventory control. However, these two companies had not integrated their data bases with middle and upper-middle management applications, such as sales forecasting and production planning. The EDP managers of both companies were rather strong-willed managers, men who would stick to their lasts unless induced to do otherwise; and apparently their upper managements had never pressed the issue of interlevel integration.

△ I found quite a different situation in Companies 9 and 10. The business operations were indeed well supported by computer applications; the operational aspects related to the product flows were supported by highly developed computer applications (for example, the ordering of raw materials, sales distribution, accounting for accounts payable and receivable, and inventory control).

More significantly, however, under strong direction of upper management, the EDP managers in Companies 9 and 10 both used *key-task criteria* to integrate their data bases. Company 9 viewed distribution and efficient accounting for billing, product movement, and pricing as key tasks, while Company 10 viewed manufacturing and planning as the keys to the overall profitability of the company. In addition, both companies had integrated their data bases for managerial reporting and analysis with their operational data bases.

In both companies one can see the beginnings of interlevel integration—both have been ranked as "medium" on this parameter in *Exhibit III*. Interlevel integration must soon appear, after all, where planning is considered a key focus for company and data-base development, as in Company 10, and where pricing decisions are considered a key focus, as in Company 9.

Without this impetus from upper manage-

ment to focus integration around key tasks, Companies 9 and 10 could never have reached the advanced stage of computer usage and data-resource management which they have attained. Management's choice of and insistence on this particular strategy was all the more fortunate, considering the popularity of other, far less viable alternatives.

Three strategies

Thus a main characteristic of the key-task strategy is the capability to respond to management's ad hoc requests for reports and analyses. A company can pursue a couple of other strategies to satisfy such requests without recourse to the data-base concept; but the alternatives are not likely to be very successful.

These are the strategies I can identify: the brute-force strategy, the piggyback strategy, and the data-base/key-task strategy. The strategy of each company interviewed is specified in *Exhibit III*; and in *Exhibit IV* I have attempted to define the three diagrammatically.

Suppose an EDP manager is suddenly given an assignment of the kind I described at the beginning of this article—that is, an ad hoc request for management information that draws across the functions and levels of the company. These are the possible ways he can do the job.

Through brute force: He can start from scratch, collecting all the needed data, coding them, writing special programs, and acquiring hardware capability, if it is required. The effort demanded by this approach is likely to be huge, the expense prohibitive, and the time demand wholly impossible. This approach is, as a consequence, very rarely followed.

Now, *Exhibit III* would indicate that Companies 1, 2, and 3 service ad hoc management requests through just this approach. The manager of one of these companies even stated emphatically that this approach is cheaper than any other—and notably cheaper than the data-base approach, which I shall come to presently.

However, he admitted that he has virtually never received such a request for ad hoc reporting. This is no small wonder; such a request would disrupt his department completely, and I suspect that this fact is known to upper management in the company.

The other two companies in this group, equally, have had virtually no experience and no success at all in servicing such requests. Claims for the virtues of brute force should be taken with grains of salt.

Through piggyback: Using this strategy, the manager attempts to "ride the special project through" on more-or-less existing capabilities. What he does is strip data from existing files, structure them into a special data pool, augment this pool with new data as necessary, expand old programs and write new ones, and increase his hardware capability, if this is required. This approach has two signal disadvantages: it requires the construction of a totally redundant data pool, and, while it consumes less money and time than the brute-force technique, the money and time are still substantial indeed.

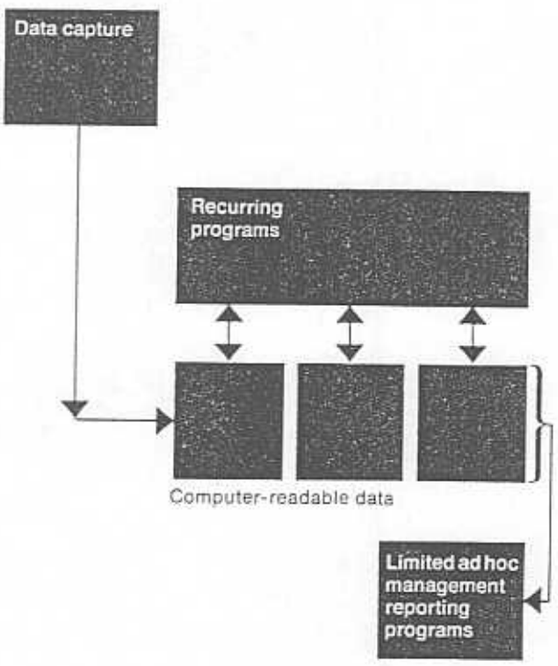
The piggyback technique is somewhat more common than the brute-force technique. Companies 4, 6, 7, and 8 have all used it, as *Exhibit III* shows. But since in every case this approach represented a special effort to obtain a special kind of information, its uses were marked by a certain narrowness and shallowness. Possible projects were limited by the quantity and nature of existing data; and the programming skills developed in these companies were not really adequate to create programs tailored to the companies' specific needs. In fact, commercially available data management systems were generally used to structure the new data pools and to generate the programs that produced the reports for management.

Through a data-base/key-task strategy: I have already described how an EDP manager would attack the problem of a management request for information; the reader may wish to look back at the organization shown in *Exhibit II*. The diagram for data-base/key-task response in *Exhibit IV* is a close parallel to the general data-base organization, but it contains some new terms and options. To explain what these mean, let me return to a discussion of Companies 9 and 10, which have evolved furthest toward a full data-base mode of operation and have used it most successfully.

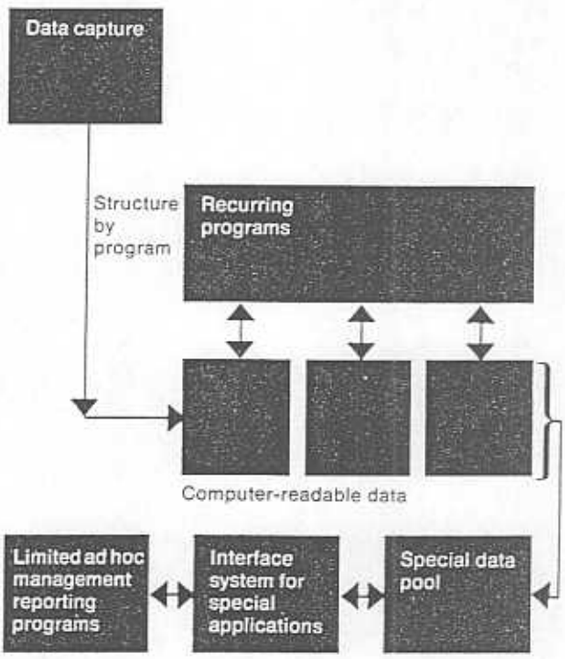
Both these companies, 9 and 10, had highly integrated data bases, as I remarked before. These bases were structured according to the companies' key tasks, and were sufficiently developed so that the companies could use commercial data-base software packages. They used one software package for data organization, defining records and files to support multiple applications

Exhibit IV. Three strategies for responding to ad hoc requests

Brute force



Piggyback

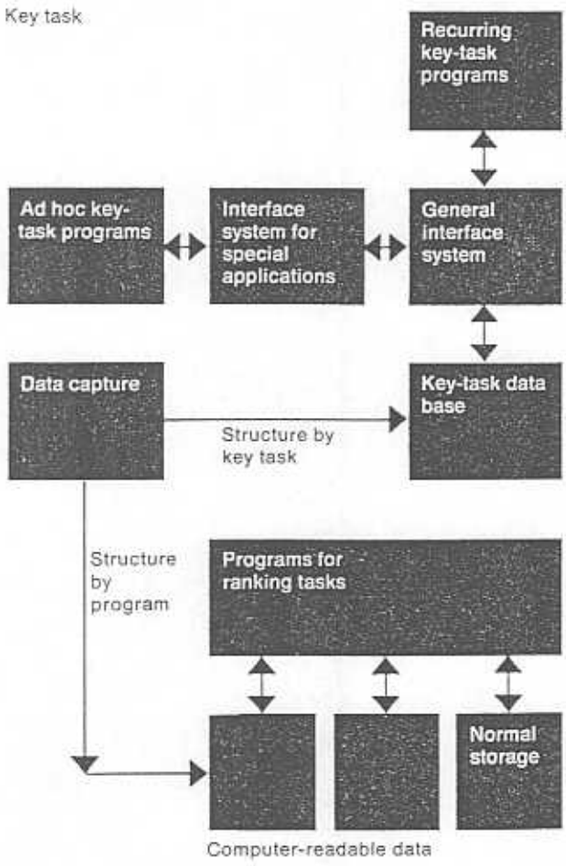


(the General Interface System of *Exhibit II*), and a different software package to produce ad hoc management reports and analyses (the Interface System for Special Applications in *Exhibit II*).

Both data-processing managers were reasonably satisfied with the commercial software they were using. Nevertheless, they both commented that even the most sophisticated data-base software commercially available did not incorporate the more advanced data-structure methods. Such methods coordinate theoretical data structures (for example, things resembling immense decision trees) with the access constraints of physical storage devices, such as rotating magnetic discs. Suffice it to say that data organization is extremely complex and technical.

It is so complex, in fact, that one is virtually forced into using commercial software. One of the data-processing managers stated that structure technology is so complex today that he could not possibly support an in-house effort to develop the software. The other manager had initially hoped to develop his own data-base software, but, after a preliminary investigation of the costs and problems, he decided to acquire a commercially available package.

However, this complexity ultimately derives from the nature of the key tasks for which top



management wants the data base to be used. If upper management focuses on key tasks that embrace all the company's data and require very extensive vertical and horizontal integration of reports and analyses, the job of organizing the data base is tougher than when the key tasks embrace only a part of the data and require less than the complete integration of all functions.

In addition, my interviews led me to a conclusion that may be directly useful to top and senior managers, as they set their sights on the issue of data-base organization: the more closely related the functional uses of the data, the easier it is to design a nonredundant, integrated data base. For example, Company 10, which structured its data base on the key tasks of planning and manufacturing, seemed to be incurring fewer problems in data-base design than Company 9, which structured its data base on the key tasks of accounting/distribution.

The planning/manufacturing structure used by Company 10 focused on vertical integration of the data base. Its manufacturing data were organized at the operations or transaction level; many of the planning data were obtained by summarizing the manufacturing data. The data base was designed to explicitly accommodate the information needs of different levels of management. Also, there was a strong linkage between organizational objectives and use of the data resource through a computer-based planning model. The model provided continuing guidelines for determining important data to be collected at the manufacturing level.

On the other hand, the accounting/distribution structure used by Company 9 required integration of data from two functional areas at the operational level. Although the structure reflected information flow, it did not explicitly recognize middle-management information needs. Thus there was not a continuing source of guidelines and benchmarks for determining which data and which analyses were important, as there were for Company 10.

Generally, then, the data-base/key-task strategy is more effective than brute force and piggyback because it forces cross-functional and interlevel integration in a manner that will suit upper management's needs. For its part, once it has decided that this strategy is the right one, upper management must take a careful look at what it considers its key tasks and make sure that it has settled on the ones that (a) make sense for the company and (b) are clearly understood by the EDP personnel responsible for structuring and maintaining the data base.

Let me return for a moment to the key-task diagram in *Exhibit IV*. This diagram makes provision for the fact that some application, ad hoc or otherwise, may be remote from the key tasks of the business. If such applications are really required, and if they are *really* remote from the organizing principles used to structure the data base, a company can always have recourse to special files and special programs, in the traditional fashion. However, it seems clear enough that a company that wants to follow the data-base/key-task concept and at the same time succumbs to the temptation to create a "special"

or tangential application every time it needs a new program may find that it is doing neither effectively.

Data-base administration

In addition to good decisions on data structure and organization, good decisions on what information will be included in a company's data base are central to the successful use of the concept. The total sum of data must be usefully large—but not ungovernably large, or the system will collapse under its own weight.

In all ten companies I studied, the systems analysts responsible for the various user areas initiated all requests to develop new sets of data or to change old ones. However, the processing of the requests varied considerably:

☐ Companies 1 and 2, still in the early stages of data-base development, permitted the individual systems analysts to decide what files would be developed and how they should be structured for different applications. These companies had not yet begun to consider data independently from the programs for which they were collected and coded.

☐ Companies 4 and 5 were just becoming aware of the need to separate data from applications and to recognize a "data resource function," but the systems analyst was still the primary decision maker.

☐ Companies 3, 6, and 7 assigned decisions on what data should be maintained to a higher and more centralized authority than the systems analysts—namely, to the data-processing manager or to the EDP steering committee.

☐ Companies 8, 9, and 10 had advanced to the stage at which they had formal data-base administrators who played a central role in monitoring the content and standards of the data base. In these companies, a request for data for a particular application was studied from a number of viewpoints: necessity, redundancy, cost/benefit, procurement methods, EDP planning, and so forth. These analyses had become sufficiently specialized to give rise to another formal administrative position—data-base analyst.

The role of the administrator

Exhibit III shows that data-base administrators actually exist in three companies—8, 9, and 10—and that Companies 3, 4, and 7 plan to fill such a position in the foreseeable future (in two to five years).

However, in none of these cases has the company fully delineated the administrator's exact responsibilities or authority. Companies seem to agree that the administrator should concentrate his energies in the areas of planning and designing the data base; and they seem to view the administrator as a useful focal point from which the whole issue of computer data can be viewed as an integrated whole. Too often companies have waked to find a plethora of fractured repositories of data scattered all over the organization.

I might note also that EDP managers with purely technical backgrounds often want to include all possible data—the president's personal files, traffic rates, and so on—in the data base, whereas managers with some executive experience in broader areas tend to take a more realistic view of the elements that ought to be included in a company's working library of data. I have argued elsewhere for the wisdom of injecting the broader managerial viewpoint into EDP department decision making.[1]

The interview discussions on the data-base administrator's responsibilities raised an interesting dichotomy. In the comments on what the responsibilities *should* be, data were talked about as a corporate resource. In the comments on what the responsibilities actually *are*, data were viewed merely as computer-readable items.

However, this dichotomy really should not be surprising. The data resource function is being carved out of the general management function. Specialization in data associated with computers is warranted at this time; the conditions for specialization exist.

But the activities for managing data as a corporate resource have not yet developed to the point where specialization is warranted. Senior management within the EDP department can still carry out these activities better than can a specialist.[2] I have already pointed out upper management's responsibility to give guidance on key tasks. Moreover, since the data-base administrator represents a relatively rare, advanced stage of development in the data resource function, it behooves top management to supervise this area closely.

A note on access: Exhibit III shows that the groups that either directly or indirectly had ac-

1. See my article, "Plight of the EDP Manager," HBR May-June 1973, p. 143.

2. John Dearden, "MIS Is a Mirage," HBR January-February 1972, p. 90.

cess to the data base are programmers and analysts. None of the companies provided either managers or their analysts *direct* access to the data base. Companies 8, 9, and 10 provided an *indirect* access for analysts through a data management system. Here the analyst communicated a request to a programmer who, in turn, used the data management system to obtain a rapid response.

In all the other companies I studied, the analyst and manager were forced to work through the process of satisfying their needs through a programmer. This process will be changed and greatly simplified as the quality of interface software improves.

Organizational & technical issues

The EDP managers expressed concern over these major organizational issues associated with the data base:

- Acquiring personnel that can handle its technical aspects.
- Funding and developing suitable charge-out systems to support it.
- Setting and enforcing companywide standards.
- Using the data resource to best advantage.

The major associated technical issues for which they expressed concern were these:

- Converting data to data-base form.
- Providing appropriate software for the interfaces.
- Designing a data base which will permit ad hoc responsiveness without degrading normal computer processing.
- Building in reliability and the ability to reconstruct lost data.

Both the organizational issues and the technical issues were generally felt to be of such magnitude that aggressive action on implementing the data-base concept fully was not warranted at the time. The consensus was that the concept is sound, but that much more needs to be done administratively before it can be effectively realized in practice.

What should management do?

Use of the data-base concept is the next natural milestone in the evolution of EDP applications. It embraces the specialization of EDP functions; it allows management real flexibility in satisfying its need for information; and it permits companies to view and use their data as a real resource. Yet caution and patience are advised in pursuing the concept. What should managers do to deal with this push-and-pull condition?

1. Take the idea seriously.

Upper management should provide direction to the EDP manager by identifying key tasks of the business and setting priorities for an improved information capability. Perhaps the single most important factor that permitted Companies 9 and 10 to break out of a parochial treatment of data was upper management's guidance and its insistence on exploiting data for the interests of the business.

2. Set up a data administration function.

The issue is when to set up a data administration function, rather than whether to have such a function. Ultimately, an administrator will be needed to implement the data-base concept, anyway. For those companies currently without such a position, an administrative structure is needed for formulating a data-base implementation plan, regulating the pace of implementation, and establishing data-base standards, controls, and access procedures. At a minimum, a data-base specialist should be acquired now to provide decision-making guidance for the EDP manager and steering committee. This person can also provide guidance in evaluating and selecting appropriate software.

3. Incorporate data-base technology into the computer system.

The hardware technology, as well as the software technology, for data bases has matured to the point that the data-base concept can be both feasible and cost-effective for many organizations. While the company will not be noticeably hurt in the short run by ignoring data-base technology, it will in the longer run.

Also, the data-base concept cannot be implemented overnight. If a company begins to plan and act now, it can assimilate even drastic technological improvements into its existing systems in a slow, comfortable, and orderly fashion.

To incorporate the technology that will permit data-base operations, an organization must

identify its key computer-based systems and restructure them (a) to remove redundancy and (b) to facilitate their use by higher levels of management. For the present, companies must probably acquire commercial software for structuring data and responding to management requests for ad hoc analyses and reports.

4. Think of data as a resource.

For the longer term, management should begin to think of data as a basic resource. It should accept this idea as a natural consequence of functional specialization of the general management function. Since the data-resource concept is closely associated with a fast-moving computer technology, management should expect to see the movement toward specialized data-management activities proceed at a faster rate than, say, specializations in the human resource function. My survey of the ten companies indicates that the emergence of the data-base administrator is a key event in the specialization.

5. Estimate your own company's position in the evolutionary sequence.

If you find that your company is still at a rudimentary stage of development, plan for development. If your company is fairly advanced, press forward a little harder. The only thing to lose is redundancy.