EVOLUTION IN COMPUTER SYSTEMS

By Mayford L. Roark

When I was asked to address this group, with its theme, "The Conversion Problem," I felt some puzzlement about the thrust of the meeting. Was there a concern that data base systems might be something of a special burden for the organization faced with a conversion problem? Or rather, was there the hope that the data base system might be something like "Seven League Boots" for making otherwise tough conversions into happy and speedy journeys? Or was there a lingering horror that the data base systems themselves might turn out to be conversion nightmares as improved DBMS resources become available?

As the working outlines began to arrive through the mail, it became clear that all of these questions were concerns. As Jim Fry's statement put it, "The basic problem we are addressing is the need to migrate data and applications due to the introduction of new, or change in the current requirements, hardware and/or software systems." In short, it appears that your intent is to map a jungle.

I won't try to preempt the work of the individual panels by offering any detailed map of my own. Rather, as a frequent traveler through this jungle, my traveler's notes may be of some help to your individual survey parties. My comments will take the form of generalizations and impressions. As you survey a topic in depth, you may well conclude that some of these were inaccurate. This has certainly been the usual result in geographic history, as detailed surveys refine the rough findings of early travelers.

As an initial generalization, let me categorize conversion problems into four families of change: hardware changes, software changes, evolutionary application development and migration to a new DBMS.

I should like to discuss each of these families of conversion problems and to relate my own impressions as to their frequency of occurrence and their significance with respect to data base mangement systems. In an effort to be as specific as possible, I will try to rate the impact of a DBMS in each case against a "BB scale"—that's shorthand for "boon or burden." If a DBMS, in my view, is a major boon or help for a given class of conversion problem, I shall score it as plus 2. If it is a moderate boon, that will be plus 1. If DBMS is more likely to be a burden than a boon, I shall score it as minus 1 or minus 2. Now let me proceed to a discussion of these families of conversion problems.

The keynote speaker is executive director of systems for the Ford Motor Company in Dearborn, Michigan.

One of my chores at Ford is to undertake, with the aid of others, a prior review of every computer hardware acquisition involving a change of processor or an added rental of more than $10,000 a year. I have been doing this for a dozen years now, during which time I have signed off on about 3,000 computer projects. If Ford is typical of the universe, I would guess that 90 percent of all data processing computers are likely to be replaced within three to five years from their acquisition. This would not hold for minicomputers used in dedicated, industrial-control applications; these may remain in operation without significant change for a decade or more. Data processing computers do change fairly rapidly, however.

For most of us, three to five years is long enough to saturate the capacity of whatever computer we have and to move on to something more powerful. We are also faced with a dazzling succession of new products every few years, always with substantial improvements in cost effectiveness. So, for most of us, I think the three- to five-year cycle is likely to continue for a while.

The extent of conversion trauma from a hardware upgrade depends on the nature of the change. If the change involves shifting to another supplier, an event that occurred in something like 5 percent of our hardware changes, the conversion effort can be an extended affair, requiring a major effort over a period of a year or more. In such cases, a DBMS is likely to be something of a burden. In all probability, the new equipment will require a shift to a different DBMS. In any event, the logic requiring conversion will be somewhat more complex and difficult to handle if a DBMS is present.

One of our divisions recently completed a conversion from Honeywell to Burroughs, a change made necessary because of reorganization unrelated to the system function. Some of the application programs to be converted had been developed in Honeywell's IDS environment; they were converted to Burroughs DMS II. IDS is a network system; DMS II is a hierarchical structure, so one might guess that we would have had problems. In fact, I am assured, both by our own people and by the Burroughs conversion team, that this transition went fairly smoothly. On the basis of that once-in-a-lifetime experience, I would have to score the

DBMS in this case as minus 1. Perhaps it would have been minus 2 if the application programs involved had been larger and more complex.

Hardware conversion can also be a serious trauma when migrating to new products of an incumbent supplier. Suppliers do present us with new families of hardware from time to time that require structural changes in our application programs and supporting software. One hopes this kind of change will be less frequent in the future than it has been in the past. Even so, I suspect we are likely to be faced with something of this kind every 10 years or so. Already, one hears strange rumblings about the kinds of changes likely to unfold two or three years hence in a new product line called "The New Grad." So far, DBMS has not figured importantly in any of our conversions within the hardware products of a given supplier. On the basis of guesswork and a skeptical disposition, I would have to assume that the presence of a DBMS for a major hardware family transition would be similar to that where a change in supplier is involved, possible minus 1 or minus 2 on the BB scoring scale.

The remaining hardware conversions, which represent the vast majority of all hardware upgrades, involve moving up within a compatible family of products offered by a single supplier. In these cases, DBMS would not be much of a factor, so we might score its presence as zero on the BB scale.

Despite IBM's assurance that MVS is here to stay, our experience would suggest that we can look for major software upgrades or enhancements from any supplier every five or 10 years. At Ford, we are somewhat preoccupied at the moment with the MVS problem at our large data center in Dearborn. The completion of this conversion will requre a major effort extending over a two-year period. There are three substantial groups of data base systems that will be affected by the conversion. These involve, respectively, IMS, TOTAL and SYSTEM 2000.

I have checked with each of the systems groups involved to see how they assess the impact of DBMS at this point, when we are about midway in the conversion effort. The managers working with IMS agree that its presence has added substantially to the difficulty and complexity of the conversion task—something close to twice as much effort as would have been involved with non-DBMS applications. The IMS conversion involves more than a new operating system, however. It requires the transition to a new DBMS called IMS-VS. One manager sees the new package as offering many attractive new features. Another manager sees no incremental benefits at all for his particular applications, but he has no choice; his present IMS software will not be supported under MVS.

The managers using TOTAL and SYSTEM 2000 see no special conversion problem at all in going to MVS.

This sampling of experience adds up to a very mixed bag. As the warnings say about some drugs, a major software conversion "may be hazardous to your health"—but again, it may not. I think we have to score the presence of a DBMS in such a situation with a range from 0 to minus 2 on the BB scale.

We do have numerous other software changes, of course, on an almost continuous basis—new releases to old operating systems, new software packages to handle new functions and the like. There is nothing in our experience to indicate that DBMS is much of a factor one way or another in these minor changes.

Now we move into the territory of what the systems function is all about and what DBMS also is all about.

Before getting into the particulars about this family of change, we ought to be clear on one central point. "Evolutionary change" is the natural state of computer systems, just as it is for biological systems. Perhaps I should not push this analogy too far because there is one dramatic difference: Evolutionary change works more rapidly in computer systems, where even five years can produce dramatic changes in structure, outputs and even objectives.

During the past five years, the workload at our major computer centers at Ford has been growing at close to 20 percent annually. I might explain here that we attempt to measure workload in BIPS (billions of instructions processed) for purposes of capacity planning. So, when I say that growth has been close to 20 percent a year, I mean that the BIPS have been growing at close to 20 percent annually.

As best as I can judge, about half of this growth results from new applications and about half from new adaptations of existing applications.

The new adaptations, which are often described by the rather condescending term "maintenance," include a lot of things. One is simply the effect of volume. If we sell more cars, other things being equal, we will process more BIPS. Another is changing requirements. In our industry, every model year is, in a sense, a new game. The products may change, terminologies may change and data requirements may change. One of the biggest sources of changing requirements in the automotive industry is government regulation. The work of the regulators never ceases, nor does the growth in requirements for additional data elements in the burgeoning computer records that we must maintain as evidence of compliance with all sorts of directives from Washington.

Some of these changing requirements are massive things like OSHA regulations or like corporate fuel economy standards. Others seem almost trivial until they are examined in the context of required changes in computer files. The industry is currently being asked to adopt as an international standard a 16-character vehicle identification number. Our existing identification numbers, with 11 characters, turn up in more than 50 separate computer files in North America alone. The cost of converting these files and their related application programs to the new international standard is estimated at $3 million.

In all fairness, the government is not the only source of changing requirements. Our engineers and product planners are pretty good changers themselves. Our service parts files include roughly twice as many parts today as 10 years ago, when our basic inventory-control system was developed. Our organization people contribute more than a fair share of changing requirements. Every time there is a corporate reorganization, we find it necessary to go through a restructuring of the hundreds, or even thousands, of computer files and application programs that are affected by the realignment.

And even systems people are contributors to the evolutionary process, only we usually call the changes "efficiency improvements." Almost every systems activity has its own more or less continuous effort aimed at cleaning up programs that run inefficiently, that are hard to maintain or that otherwise need overhaul.

To make another sweeping generalization, I would judge that each of our systems activities annually rewrites somewhere between 5 percent and 25 percent of its accumulated code.

What a fantastic opportunity for data base management systems. I would score the DBMS impact on this area of evolutionary application development as plus 2 and proceed to the next topic except for one problem. The DBMS benefits can be extremely difficult to realize in practice, and the price of the cure is sometimes worse than the pain of the ailment.

Many of our files most subject to change are huge affairs. Some of these files run into the billions of bytes of data. For systems of this size, processing costs may be counted in six or seven figures annually. An added overhead burden in a range of 30 to 40 percent can amount to several hundreds of thousands of dollars in added annual cost. One of our activities last year made an analysis of overhead associated with one of its large data base systems. It found tht 70 percent of the instructions being executed resulted from DBMS overhead. This activity is in the process of restructuring its DBMS with the objective of reducing processing requirements by a full 40 percent.

Several years ago, another of our activities launched a large-scale data base system in which nearly half of the file requirements were for pointers. Processing requirements in such a case can far exceed expectations based on experience. This sort of overhead can mean a loss in processing productivity, something that has to be weighed against any benefits in programming productivity.

The best score I can give DBMS, therefore, for its impact on "evolutionary application development," is plus 1. It sometimes works well, but it can also be awfully expensive.

Somewhere in the deliberation of this "or related groups," there ought to be some consideration of what can be done to simplify DBMS technology. If this is not possible, perhaps there could be some guides or standards to protect the systems designer with a modest problem from unwittingly stumbling into a huge solution out of all proportion to need. Sometime in the future, I would like to go through an exercise like this again and conclude, without any reservation, that DBMS is an unqualified boon for the evolving system regardless of its size and complexity. We are still in the very early stages of DBMS art.

I have saved the family of conversion problems in migration to a new DBMS until last. In a sense, it overlaps all three of the categories discussed earlier. Migration to a new DBMS may be prompted by a change of hardware. It may be forced by a change of software. Or it may be undertaken with a view to realizing the benefits discussed under "evolutionary applications development." Still, the migration to a new DBMS is a major event that deserves consideration in its own right, whether the migration is from a non-DBMS environment or is the rare change from one DBMS to another.

There is a logical inconsistency in assigning any rating at all to this family of conversion problems. It is like asking, "What impact does DBMS have upon itself?" Yet, at the risk of sounding completely illogical, I am going to assign a BB rating of minus 1 to this category of conversion prob-

lems on the ground that a DBMS is a high-risk undertaking entirely apart from whether it is related to changes in hardware, software or applications.

Moving to a new DBMS is not unlike the process of getting married. It takes a lot of desire, commitment, sacrifice and investment. It involves moving to a wholly new lifestyle. Once there, the return to the old lifestyle may be difficult or impossible without wrenching adjustment problems.

I have several times had the experience of encountering a spokesman for some other organization who tells me, "We've made the policy decision that all future development work in our organization will be based on DBMS." This makes me shudder. It is a little like saying, "We've decided that everyone should get married." I must add, however, that in every instance cross-examination has established that, policy or no policy, the organization in question still does a lot of its computer business outside the DBMS environment. I expect this will be true of nearly all of us for many, many years to come, or at least until the DBMS technology can be simplified to the point where it no longer requires total desire, total commitment, total preparation and heavy investment.

Not too long ago, we compiled a catalog of all the computer system applications in use at our North American divisions and affiliates. The listing came to something like 50,000 application programs. Some of these resulted from major projects requiring scores of manyears to develop. Others were relatively simple. The full range of applications represents a wide spectrum of data needs and complexity. DBMS technology today does not really address this whole range of developmental requirements. Perhaps it never should. In any event, migration to a new DBMS system must be taken as one of life's climactic events to most systems people. We all need more guidance about when to undertake such a migration and how to stay out of trouble when we do.

These, then, are the major problems to be found in this jungle of systems conversions. To summarize, my BB scorings have suggested that a DBMS can constitute a moderate-to-heavy burden for major hardware conversions and major software conversions. The presence or availability of DBMS, however, can be a moderate-to-strong boon for evolutionary application development. Finally, the migration to a DBMS system can be a long and tedious journey, especially where existing systems of great size and complexity are involved.

DBMS, in short, still offers attractive visions of a world in which systems might respond quickly to changing requirements, in which new creative applications might be easily spawned from existing data files and where data bases, in the words of Dan Magraw at the earlier conference of two years ago, can "move the DBM's into the area of decision making."

These visions ought not to be taken lightly. In preparing for this meeting, I checked back with our managers who have been in the DBMS mode for five years or more. What benefits did they really get? Their consensus might be summarized as follows:

1. They, indeed, have been able to respond more quickly to changing requirements, although not always as easily as they once hoped.
2. New applications development has been easier. The managers see a productivity improvement factor in a range of 10 to 20 percent.
3. Most important, the managers believe they have capabilities that previously did not exist at all.

Let me expand on these capabilities for a moment. Those 50,000 computer programs I mentioned are a long-time accumulation in response to numerous problems and opportunities that were perceived by our division and staffs over a period of two decades.

Our typical division, which might correspond to a moderate-size company, has its own accumulation, usually a range of 1,500 to 3,000 programs. If the division is not yet in a DBMS environment, each of those programs comes with its own set of files. In the past two years, we have been greatly influenced by the concepts of "top–down design" and "structured programming." For systems of recent vintage, these approaches may have provided a logical structure that would give some accessibility to files. For older systems, with what our people call "spaghetti programs," accessibility is something else—fragmented and scattered files, with all the access keys hidden in "spaghetti code."

As functional entities for the purposes they were first created, these old programs may serve well. Even where we want to launch a new application that will need to draw on the data in these files, the problem is not overwhelming. We can, and do, write programs to get at the needed data files, wherever they exist.

But suppose we do not want a new system. All we want is an in-depth analysis of a difficult problem on which five or 10 different files have something useful to say. This is the sort of problem that gives computer people a bad reputation as being slow-moving and unresponsive. It can be extremely difficult to extract data from five to 10 different sources, all with different maintenance cycles and data-control procedures, and produce anything but a mess.

We do a lot of computer-based analysis at Ford because we have found that the data resources in our computer files can open up all sorts of insights and understanding that would otherwise be lost. I wish we could do even more, but this form of analysis can be very time-consuming and frustrating in the non-DBMS environment. We have been working on one such exercise involving non-DBMS file sources for more than three months, all because of the relative inaccessibility of data. Last week we decided to skip a promising analysis altogether because it would have taken

more than a month to extract and organize the needed data from a variety of source files.

So when our managers talk about new capabilities from DBMS, they are talking about one of the company's most important potentials—the power to carry analysis to entirely new levels of understanding.

The benefits cited by our managers are impressive testimonials. Why, then, has the data processing world been so slow in converting to DBMS? I have seen no studies that would provide an accurate measure as to the proportion of data processing oriented to DBMS. In the absence of any sure data, I am going to offer the opinion that the proportion is no greater than 10 to 20 percent.

I have never expected that all computer systems would, or should, move to DBMS. In the early '70s, however, as we first began to realize the potential of this technology, most of us, even the conservatives I believe, would have expected that something approaching half of all computer files would acquire a DBMS format by the end of 1977. Even at Ford, where I believe the impact of DBMS has probably been greater than average, the progress has seemed slower than I would have expected. This painfully slow progress points up perhaps the greatest conversion problem of all: How can we move to a DBMS environment with existing systems?

Most of our DBMS user divisions made their first moves to data base at least five years ago. A couple of these divisions went through a conversion trauma, eventually recovered and never undertook a subsequent DBMS project. Once was enough, they concluded.

The other divisions have continued to extend their data bases in areas of new systems development. But this leaves a very large accumulation of computer files more or less untouched by the new technology. One manager, possibly our most enthusiastic data base advocate, believes that about 40 percent of his divisional data is now in DBMS format after some five years of development. He guesses that this figure may reach 70 to 80 percent in another five years.

We have still another group of divisions with heavy maintenance workload which has elected not to try the DBMS approach at all.

The problem is not unlike that of our Detroit skyline. We recently completed a magnificent new Renaissance Center along the riverfront, with some of the most beautiful hotel and office structures to be found anywhere. This is exciting, but there is still a long way to go to bring the rest of the city up to Renaissance Center standards. The accumulation of history is still a huge obstacle to those who want the best of things right now. Omar Khayyam, who summed up so many things in language that a sophomore can understand, expressed this frustration perfectly:

". . . Could thou and I with fate conspire
To grasp this Sorry Scheme of things entire,
Would we not shatter it to bits—and then
Re-mould it nearer the Heart's desire!"

All of us, however, have to find ways to work with what we have inherited. We might all wish we could somehow get rid of the old mess and start all over again. Unfortunately, that old mess represents an investment in the hundreds of millions of dollars for my company and in many tens of billions of dollars for all of us collectively.

One of our divisions several years ago tackled this rebuilding problem in what seemed to me an innovative kind of way. This division had identified 15 overlapping files that had evolved over the years, with inventories and other data related to parts. As might be expected, there was much redundancy, and it was difficult to reconcile one file to another. A complete overhaul of the applications programs did not seem feasible, but the division hit on the idea of a DBMS master file to serve as a sort of front end to these application programs. There was a saving of more than $100,000 annually in data preparation and data control. This limited effort brought the division into the DBMS environment and laid the basis for solid evolutionary development, including subsequent application revisions to exploit more fully the potential of the data base environment.

If there is one thing I would particularly want to see come out of this conference, then, it would be some easy-to-use, easy-to-apply and inexpensive approaches for upgrading this great accumulation of computer files we have all been working on for the past two decades or so. We have all had tantalizing but too-brief experiences with data bases as they can be. The question before us now is, what can we do to make those benefits available wherever we need them?

Where does an organization with an accumulation of 1,500 to 3,000 programs begin, without going out of business for a year or two? What tools can it use to map out its data resources? How can it go about restructuring these files without scrapping its investment in application programs? I hear that some of you came up with answers to these questions. Perhaps you can point the way to this new data-rational world we all seek.

The realization of this promise is still a long way off. The real world of tomorrow will come when the DBMS can contribute to the evolutionary process of applications development and to the full analytical use of computer resources without exacting an extortionate price. When the DBMS can aid in the evolutionary process of hardware and software change. And when the decision to go to a DBMS is no longer a high-risk affair requiring an all-out commitment through one of the most difficult conversion problems to be found in systems.

I have no doubt that something very much like this world of tomorrow will appear one of these days, in great part because groups like this one pressed on relentlessly in the definition of problems and the search for creative solutions.