Organizational issues of effective maintenance management

by GARY L. RICHARDSON
Texaco Inc.
Houston, Texas
and
CHARLES W. BUTLER
University of Arkansas
Fayetteville, Arkansas

ABSTRACT

It is a continuing challenge to today’s data processing (DP) organization to evolve a management structure that matches the technological advances of the system for which it is responsible. The goal of this paper is to synthesize the emerging role of a DP organization within its corporate environment, focusing particular attention on the issue of software maintenance.

Attention is primarily on three dimensions of the required organization: the user view, organized by functional area; the technical view, organized by area of expertise; and the organizational view, arranged by planning horizon. The conclusion is drawn that a single group should have responsibility for integration and enhancement of all installed applications. The tasks of this group comprise software configuration control, operational integrity, performance tuning, and requirements analysis and planning support for installed systems.
INTRODUCTION

Today's data processing (DP) technology is headed in many different directions at once. This explosion in technology, together with twenty years of systems development, has suddenly brought us to the exciting threshold of new organizational strategies and direction. The goal of this paper is to present a generalized organization structure that facilitates development and enhancement of software, focusing particular attention on the software maintenance function. The ideas raised here are meant to be independent of any single organization and are designed to help the reader sort through the myriad of factors that affect a particular organization. However, it should be emphasized that the approach presented has been applied within a large DP organization.

THE EMERGING DP ENVIRONMENT

A practical framework for analyzing the DP function can be formulated by reviewing critical operational components and observing their unique pattern of evolution or growth. In the final analysis, organizational structure depends on three distinctive factors: data resources, technological components, and the application environment. Because of the integration of various levels of technology, the organization generally evolves according to the pattern outlined in Table I. The driving force in these evolutionary stages is hardware technology, which supports increasing capability, as demonstrated by today's complex network configurations and operating systems. This fact is illustrated in Table I by the diagonal arrows, which indicate the pushing effect of technology on the application environment and organizational framework. When one compares it to these technological advances, it is apparent that the evolution of application software and organizational framework has lagged behind technical sophistication.

Organizational growth is continuous, therefore difficult to describe in discrete intervals, but recognition of four points along the continuum provides critical insight into the emerging organization. The four points are as follows:

1. Narrow.—Low personnel requirements, data maintained as a separate file for each application; focus is on input and output (I/O) and on replacing existing manual procedures.
2. Expanded.—Extension of application functions or processes (I/I/O) and an accompanying growth of personnel resource requirements. Processing becomes more complex.
3. Functional specialization.—Integrated systems composed of major functions and processes (I/I/O); personnel with specialized knowledge within critical functional areas.
4. Multidimensional support.—Automated applications cover the firm's entire operating sphere; information centers emerge that focus on query and decision-oriented areas.

In the past, applications such as payroll or accounts payable were designed with limited scope. Today, each of these is merely a subsystem, a single element of multisystem integration. The firm is now operating these virtual applications. They are critical in that they are embedded in the firm's functional operating environment. However, even with this level of sophistication these systems are never quite finished, owing to the dynamic nature of business operations.

Along with the existing virtual system complexity, most large organizations have also been expanding their services in multiple directions. A sample of these is as follows:

1. Providing on-line access to users
2. Installing nonprocedural software for users
3. Directionally moving toward the information center concept
4. Decentralizing installation of personal computers
5. Installing text processing and/or electronic mail systems
6. Connecting multiple processing and user nodes into distributed networks

It takes only a casual glance at this list to see that the scope of work undertaken has broadened considerably. It is not so obvious what pressure this brings on the traditional DP organizational structure and on its procedures for maintaining existing software systems. The breadth of hardware and utility
software used is too much for one group to handle. In other words, the new environment demands a different organizational approach to meet the breadth and complexity of demands being placed on it.

DESIGN FACTORS

Operating Factors

The fundamental issues affecting the structure of organizational units should be examined in such a way that the following five questions will be addressed:

1. What is the perceived goal of the information systems department?
2. Should software systems be carefully planned before coding, or is prototyping the best technique?
3. Should development work units be organized primarily around skills or around user functions?
4. Could your organization operate adequately if the computer system were lost through a catastrophe?
5. What are the typical problems encountered by users?

Obviously, these questions can be answered in a wide variety of ways. Collectively they indicate the continued need for an application development unit.

First, the goal of the information systems department is more and more recognized to be the maximization of the value of corporate information (including data, graphics, video, and voice, as well as text). The chore of writing new traditional systems code is declining relative to the new support services such as telecommunications (networks), training users, decentralizing hardware, and managing the orderly growth of computer technology within the total organization. This suggests that user involvement with the organization is broadening in scope beyond the functional systems specification activity.

Second, if we can prove that the use of good software engineering practices does not adversely affect delivery times for new systems or enhancements, there will be increased justification for splitting system specification and analysis away from code construction. However, a large segment of DP professionals believes that prototyping is the only way to find out what users really want. Quite probably there will always have to be some prototyping in the process of collecting specifications; possibly skill in prototyping is needed in application development.

Third, development work needs to be controlled by professionals who understand both the user requirements and enough about code construction to make intelligent design tradeoffs. We propose that development groups be supplied with carefully described specifications, be predominantly skill oriented, and work from formal design blueprints, just as carpenters would in building a house.

Fourth, if you feel that the computer is not indispensable, then your organization has not reached the evolutionary stage we have been describing.

Fifth, from the list of user services developed from questions 1 you should observe a wide assortment of requests and problems, embracing such items as data availability, purchased software, micros, remote terminals, local training, and so on. The list should definitely include more than just simple application development system software concerns. The truth of this premise would mean that we really cannot simply organize around user requirements. They would need the entire organization! Rather, we must consider techniques to provide a simplified organizational view to users while hiding the true organizational complexity away—in other words, a multidimensional structure.

Human Factors

Human factors are also important in the design of a DP organization. It is important to consider, for example, whether computer professionals naturally relate to users or to technology. Research evidence indicates that they are primarily technology oriented and that well-trained individuals can learn how to translate a reasonably well-defined user problem into a computerized system. This suggests that staffing is easier when it is based on skill groups, rather than on functional applications. If requisite skills within the organization were homogeneous this would be a moot point. That is not the case today, and the cost of broadly training individuals is becoming quite noticeable. In addition, there is a limit to the range of divergent technology an individual can be technically proficient in. A natural conclusion is that skill-centered units are the most natural organizations for the application development and technical-support functions.

Organizations can hinder or help human productivity. In order for a professional to understand his or her role, the organization must provide a coherent structure for logically perceived work processes based on a division of labor. One more decade of developing and operating information systems with current techniques and philosophy will bring even greater chaos. Too many options are being generated by technological development, and management is increasingly looking at computer technology as a tool, rather than a status symbol. The organization cannot tinker with technology and human resources; it must apply good business decisionmaking principles.

THE IMPORTANCE OF SOFTWARE MAINTENANCE

Is maintenance important? The answer to this question is unequivocally yes. We are now entering an era in which the operational nature of most software systems is so intermeshed with functional systems that large segments of the organization cannot perform their duties when the computer system is not operational. Changes to these systems are increasingly complex because of this and because of the integrated nature of the systems. Finally the investment that firms have in installed systems is becoming widely recognized, as system retrofit activities highlight their replacement cost. In order to extend system life, the role of maintenance is critical; it is clearly significant for the system life cycle. As illustrated in Figure 1, the system life cycle can be viewed as a series of

From the collection of the Computer History Museum (www.computerhistory.org)
broad subprocesses: requirements definition, construction, and maintenance/enhancement.

Another critical factor that helps to explain the importance of maintenance is the scope or impact of maintenance changes. Research by Swanson and Boehm aids in sorting out the importance of change scope. They divide maintenance work into four levels of change scope: (1) corrective, (2) adaptive, (3) perfective, and (4) expansive. Comprehensive understanding of existing software is required for successful maintenance changes. Corrective and adaptive maintenance requires an understanding of existing software and is generally restricted to small segments of an existing system. As the scope of a change progresses into the perfective and expansive categories, system performance and functional enhancement become the primary goal. In these instances, more knowledge of component interaction is needed, even though relatively small amounts of code are used for successful changes. In essence, reevaluation of system construction is performed.

Three conclusions can be drawn regarding the nature of software maintenance. First, the demand on the maintenance function depends on the quality of products that are generated during the development phase of the life cycle. Second, regardless of the quality of the software product, maintenance success depends on a level of understanding of the existing system. Finally, even the smallest software change is significant because of the value of software as an asset.

GENERALIZED ORGANIZATIONAL STRUCTURE

A multidimensional organizational structure is required to assimilate technological innovation properly, address broad user needs, and manage the software life cycle properly. Each of these requirements is addressed in the organizational structure illustrated in Figure 2. First, the horizontal axis represents the technical scope of the DP organization. By organizing along technical skill lines, specific skill requirements can be developed. Next, the vertical axis represents an appropriate packaging of the key functions of the DP organization.

The various work units do not extend continually up the organization, because at higher levels less technical and more managerial skill is required. Finally, the third dimension represents the extension of data processing into the firm’s operation and is a functional user view of the organization.

Given the wide range of technological alternatives and diversity of user requirements, at least five major organizational units are required for multidimensional support. As shown in Figure 2, they are as follows:

1. Planning—Development of strategic and tactical plans, including (a) the primary planning interface to the management of operational departments and (b) evaluation of technical advances.
2. Application development and support—Software development and support activities that are taken within the department.
3. Technical support—Technical support functions required for the department to perform its responsibilities (systems programming, DBA, telecommunications, etc.).
4. Administration.—Administrative functions involving the personnel and financial activities within the departments’ operation.
5. Operational support.—Operation of the computer centers and remote I/O stations.

Salient to the issue of effective life cycle and maintenance management is the applications development and support unit. Our premise, which is supported by research conducted by Mooney and Canning, is that applications development and support will be composed of two distinct responsibilities: development and installed applications. Figure 3 is an example of application development with four skill environments (e.g., COBOL, PL/I, APL, and micros). The skill requirements shown are intended to be development-oriented skill groupings. For instance, a COBOL group would be distinct from a PL/I group. The point is that skills are rarely intermingled within the same development work units.

Note that the traditional maintenance function is retitled
"Installed Applications" and made responsible for the proper functioning of all installed systems, as well as for minor enhancements. It focuses on software configuration control, operational integrity, performance tuning, and requirements analysis and planning support. It should be staffed with an appropriate mixture of professionals and senior personnel who can handle the broad range of tasks implicit in this function. The installed applications will require managerial and technical skills critical to the activities of software maintenance.

As a responsibility of the application development and support unit, maintenance activity typically consumes more than half of the development function, and the growth of virtual applications will bring additional support requirements into the maintenance programming sector. In this environment, it is necessary to learn how to efficiently maintain installed systems to provide optimal resource allocations in new development areas and a stable data resource base for other applications to use in a decision support mode. The installed applications group concentrates on operational systems, which are a basis for shared data in the firm. The proposed structure offers the operational benefits of a wide range of technical skills within which maintenance is a significant component.

MAINTENANCE MANAGEMENT BENEFITS

As stated earlier, an organizational structure can only facilitate productivity. It is up to management to optimize human and technological resources. Operating within the general organizational structure, the installed applications unit presents a number of operational and managerial features for maintenance management:

1. Quality assurance.—Experience reveals that the requirements definition (analysis) is heavily user dependent, while the construction (design, coding, and implementation) is more heavily slanted toward the hardware/software environment. After implementation, the operation/maintenance phase requires both user knowledge and technical skills at various times. In all circumstances, maintenance is performed on products originally developed through requirements definition and construction. A final system acceptance point is established between the two subprocesses; inferior products can be rejected at that point.

2. Low prestige factor.—Separation of functions can enhance image through top-level management recognition and participation. When a development project is completed it is transferred into the installed applications unit along with a skeleton support staff.

3. Resource level determination.—Resource allocations to a recognized function should result in equitable human resource allocation for maintaining the valuable software asset.

4. Unique skill requirements.—Educational and training programs can be established for the function, not the individual. Special tools can be developed or acquired to complement the function.

5. Professional development.—Job rotation is handled at the planning level, aggregating individual needs and training objectives. Thus, a professional can perceive that he or she is not buried in the installed applications area.

6. Change integrity.—Through orderly job rotation a constant base of knowledge is maintained. By organizing maintenance as a single function, better internal control can be implemented to ensure that changes are authorized.

CONCLUSIONS

From a practical point of view, installed applications are an unavoidable fact of life. It has been stated that "the main problem in the maintenance business is that you cannot just do maintenance on a system which wasn't designed for maintenance. Unless we design for maintenance, we'll always be in a lot of trouble after a system goes into production." Recognition of this has legitimized many of the methodology developments during the past decade. Unfortunately, it glosses over the major problems and fails to address yesterday's legacy of a voluminous software inventory that we are maintaining today.

The primary objective of this paper was to focus on organizational factors that facilitate maintenance management. However, the complexity of the function and the related sensitivity of the organization to the subject required that this discussion go beyond the single issue of maintenance. In particular, there are three major elements that must be integrated to effectively provide support for installed applications; they have been identified as follows:

1. Organizational structure.—A formal organization architecture within which resources are distinctively responsible for maintaining the firm's applications.

2. Clearly defined work process.—An identified life cycle within which the natural growth of installed applications can be managed and controlled.

3. Special technology.—Unique manual and automated aids which serve as diagnostic, investigative, and corrective mechanisms for today's complex applications.

Within each of these areas, managerial expertise is required to overcome inherent problems of morale, professional development and training, coordination between work stages,
assimilation of technical tools, and integration of multidimensional support. It is hoped that the reader can identify pertinent factors that apply to a local organization and use them in constructing a path for his own organization.

Many well-read writers such as Martin and McCracken declare that the coming of fourth generation software is going to allow users “simply” to query mystical databases and answer whatever question is on their minds. This sounds neat, but our experience with approximately 1,000 such users suggests that user training, consulting, database design, and data migration are all left behind with the computer professional. While we hope that this approach will slow the growth of expensive maintenance for traditional software and cut user backlogs, we have recognized that maintaining corporate software assets is a critical DP function.

ACKNOWLEDGMENTS

The authors wish to acknowledge the contributions made by L. J. Crockett and D. W. Hutzelman of Texaco’s Computer and Information Systems Department in Houston. Much of the structuring concepts presented here resulted from discussions with them; their input helped shape many of the ideas expressed here.

REFERENCES
