Section VI

PROJECT CONTROL

THE BELL SYSTEM TECHNICAL JOURNAL
SAFEGUARD SUPPLEMENT
Printed in U.S.A.

SAFEGUARD Data-Processing System:

Management Overview

By E. J. DAVIS and H. M. JACKSON II

(Manuscript received January 3, 1975)

This paper describes the management approach developed to support the SAFEGUARD software design effort. Project organization and some techniques used for planning and control are discussed.

I. INTRODUCTION

The magnitude and scope of the SAFEGUARD system software-design effort presented unique management challenges across a broad front. Solutions to problems involving organizing, planning, activating, and controlling had to be tailored to the specific needs of the project. Successfully achieving the objectives of perhaps the most ambitious software development effort undertaken to date was no easy task. Although no dramatically new techniques or remarkable insights into the management process emerged, several useful lessons were learned. While there was not a wealth of tradition and folklore to draw on with regard to similar software development efforts, we found that the fundamental management approaches and disciplines developed over the years in hardware and systems design and other software development activities at Bell Laboratories were in most cases directly applicable.

II. ORGANIZATION

The organization structure that emerged for managing the SAFEGUARD software project is a case in point. We established an organization designed along the general lines of major deliverable generic systems. This organization is shown in Fig. 1. Note that there were four centers reporting to the project director. One center was charged with total SAFEGUARD systems design responsibility. This meant that this center concerned itself with high-level requirements, with evaluation of the design, and with customer interaction. This center undertook software design in the form of simulation programs and other

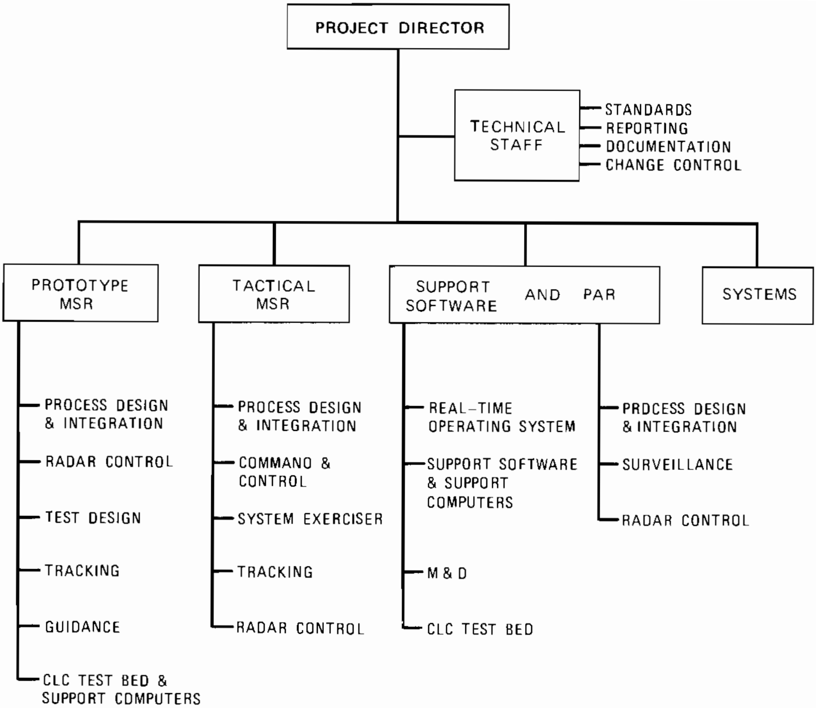

Fig. 1—Organization structure.

analytical tools which were necessary to support evaluation or the development of requirements, but designed no software system deliverable to the customer. Each of the other three centers was charged with design, test, documentation, and delivery of software associated with specific radars, i.e., the prototype Missile Site Radar (MSR) at Meck Island, the tactical MSR at Grand Forks, and the Perimeter Acquisition Radar (PAR) at Grand Forks. The PAR center was also charged with the responsibility for designing support software for the tactical radars.

The departments within these centers were given specific functional design tasks as indicated by their abbreviated titles. The identification of a number of subprojects, derived from the total project work breakdown, permitted a second organizational structure to be superimposed on the line organization structure of Fig. 1. Figure 2 shows one of these subproject organization structures for the MSR weapons subsystem. A project manager was designated for this subproject; in this case, he was the department head (second-level manager) of the Process Design and Integration Department. His responsibilities as project manager included high-level planning for the subproject, detailed design and its implementation, integration and testing at all levels, and monitoring and control of all subproject critical activities. He generally was the person who scheduled and conducted design reviews and periodic project meetings where key engineers, programmers, first-level managers, and support personnel worked together to identify problems and initiate action to solve the problems. The subproject meetings also were used to disseminate information of interest to all those working on that particular subproject. Because the organization remained intact throughout the life cycle of the project, the project manager frequently was called on to preside simultaneously over control of a released system, a system in the planning and design stage, and one in the integration and test phase. The project manager was given a great deal of latitude as to how he managed his subproject. As is evident from Ref. 1, a variety of management approaches were used concurrently, and many contributed to the overall project success. Emphasis was on results rather than technique.

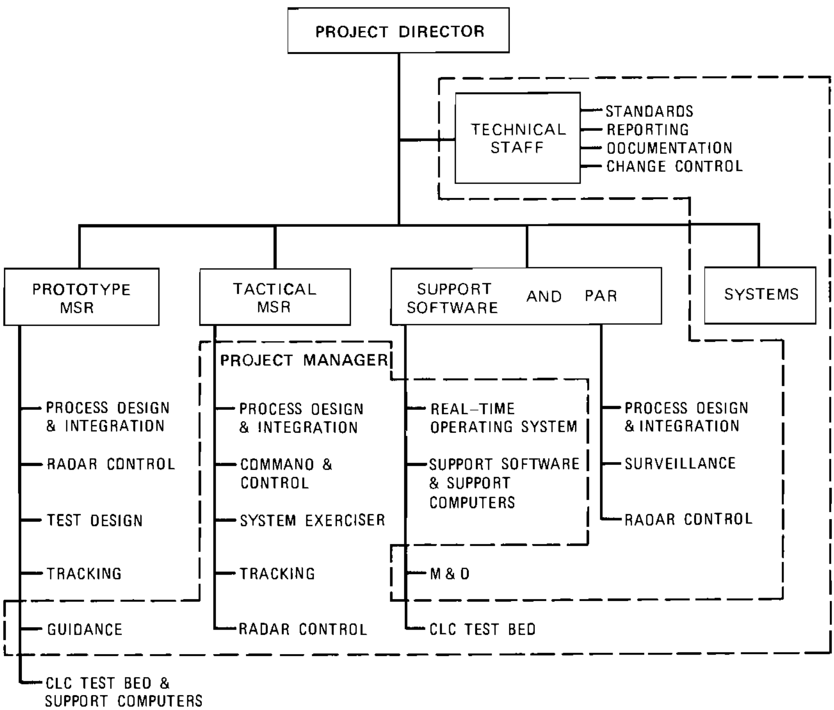

Fig. 2—Subproject organization structure.

Figure 2 shows that the MSR weapons subsystem manager considered people in other centers—for instance, the systems engineers, the CLC test bed operation, the guidance designers, the real-time operating-system designers, and the support-software and support-computers people—as part of his subproject. Note the horizontal spread of this project as it reaches across center boundaries for the people to provide its component parts. Conceptually, it illustrates the coordinated system of relationships among essential functions typical of a matrix type of organization.

All together, there were 17 subprojects—some of them nested within major subprojects like the one mentioned above—with project managers at the second level of management. Experience proved that there was a great deal of commitment to subproject goals on the part of all personnel involved. Clearly, this structure had the potential for conflict—particularly relative to critical resources like the CLC test bed, where goals for two or more subprojects were in competition. However, overall project goals were pretty well understood at all management levels so that conflicts rarely had to be referred up the line-management chain for solution. While the potential conflict situation was recognized, the benefits of cross-fertilization were also a consideration. Good ideas and design approaches were frequently passed rapidly from one subproject to another because of subproject ties that spanned the line organization.

In Fig. 1, note that there was a technical staff organization that had the charter to attack certain projectwide problems, such as training, project standards, documentation, change control, and management reporting. In some areas it provided services to the various project managers, such as training new people. In other areas, it acted as a catalyst to cause project standards to be created. It was not an enforcement agency. For instance, this group sponsored studies and development of structured programming and promoted the development of critically needed macros, but it did not have the authority to impose structured programming as a standard on any subproject. That type of decision was in the province of the project managers.

The project management approach as implemented on the SAFEGUARD software project proved to be a stable organization capable of eliciting strong project commitment at the working level and close technical control in the appropriate line organizations.

III. DETAILED PLANNING

Once overall project and subproject goals were defined and an organization was designed to accomplish them, a detailed development plan was constructed. This development plan, which was prepared in parallel with the Data Processing System Performance Requirements (DPSPRs),[2] forecast the needs of the entire project and spelled out the development approach.

Estimating algorithms, derived in part from a study of previous Bell Laboratories work in electronic switching systems and software development for earlier military systems, were used to help plan the allocation of resources. These algorithms were applied to the estimation of resource demands for each major activity. Schedules were then built up within the constraints of budget, time, and manpower. Trade-offs among these primary resources allowed the coordinated scheduling of critical activities. This anticipation of requisite predecessor/successor relationships between various parts of the job was designed to minimize delays, bottlenecks, and interruptions. Obviously, the initial plan was changed many times during the course of the project. However, it eventually led to very detailed plans which were extensively used throughout the project.

The planned addition of large numbers of people to the project, coupled with an increasing reliance on subcontractor performance, presented a significant management challenge. For example, the accomplishment of in-house training required establishment of a corps of instructors and preparation of text materials. The overall plan had to provide for this substantial investment in student and instructor time. In some cases, where traditional mechanisms were not feasible, novel techniques for evaluating and controlling subcontractor performance were adopted. One such method, the Cost-Plus-Award-Fee contract,[3] was considered one of the major project successes.

In order that forecasts of manpower buildup and total project cost be realistic, it was important that the development plan be implemented and kept current. To this end, a management reporting structure was set up by the technical staff organization to update the development plan and schedules and to provide monitoring information to project managers.

The significance of planning was that it existed across the entire project and that it used reasonably consistent definitions. The subproject managers were not required to use the algorithms that had been put together in the original development plan in working out their more detailed plans.

A conscious effort was made throughout the planning process to require the active involvement of those people who were to be charged with the responsibility of implementation. Participation in the formulation of goals, plans, and schedules conduced to a personal commitment to carry them out. In addition, the unconstrained format of the plan encouraged teamwork and emphasized the use of creativity.

IV. STANDARDS AND CONTROL

Development of appropriate standard operating procedures for designing, testing, documenting, and delivering software was a difficult and tortuous process. Since comprehensive standards did not exist at the start of the project, to a certain extent it was necessary to develop them in parallel with initial development of the software itself.

Rather than create a large, specialized bureaucracy, a small group was organized to act as a catalyst for generation of necessary standards. This group identified the need for specific standards either independently or through requests from design or test groups. A sponsor, usually from one of the design groups, was appointed for each required standard. The sponsor, in concert with designers from other subprojects, prepared a draft that was circulated to the management of affected organizations. Eventually, through a process of iterative feedback, each standard was approved at the highest level for projectwide implementation. In practice, this procedure proved very time-consuming, frequently requiring reliance on preliminary drafts when no approved standard existed. As might be expected, one of the first standards that was provided consisted of a procedure for changing standards.

The standards were divided into a number of different areas, the major ones being change management, documentation, and management reporting. In the area of change management,[4] for example, standards for "freezing" a software unit were developed. As a minimum, to be considered for freezing, a software unit must have been properly documented, successfully assembled or compiled, and successfully unit-tested. While freezing did not stop changes to software units, it did require the application of configuration control procedures, which made all proposed changes clearly visible to interested managers.

Also included in change management were standards and procedures for reporting program malfunctions. The primary mechanism was a standardized trouble report/correction report form that kept all information about a problem and its solution on a single sheet of paper. This report was eventually adapted for describing any discrepancy between observed status and requirements and, as such, became very widely used to track current program status.

Documentation standards attempted to identify and describe every type of document that was needed. Since documenting any large system is a costly and time-consuming process, each requirement was subject to the criteria of reasonableness, usefulness, and timeliness. First, it is not reasonable to expend a great deal of effort to produce a formal document when the information it contains can be made available less expensively in other ways. Second, there is no point in preparing a document that is not going to serve a useful purpose. Finally, a document's utility is greatly diminished if it is not available when, where, and in the form that it is needed. Certainly, schedule constraints did not always allow the criteria to be met, and quite a bit of learning as to just what was useful took place only after the documents were put to the test of use.

Management reporting standards were keyed to a computerized management reporting system that was developed for use on the project. The system incorporated data bases for schedule, manpower, and computer usage information, and was designed to produce a wide variety of special-purpose reports.

V. DISCUSSION

Although, as stated before, no major new management techniques emerged during SAFEGUARD development, the project's success can be attributed at least in part to the close attention that was paid to the content and control of requirements documents and to the early and detailed planning of testing. Most important, highly skilled technical people were selected for key management positions. They were relieved of most tasks peripheral to their jobs, and, subject only to the constraints of *necessary* standards and control, they were allowed to use their own style.

The papers that follow deal with some lessons learned in establishing software change control systems and subcontract administration systems. A critical appraisal of SAFEGUARD project management—as seen by the managers—is also included.

REFERENCES

1. J. D. Musa and F. N. Woomer, Jr., "SAFEGUARD Data-Processing System: Software Project Management," B.S.T.J., this issue, pp. S245–S259.
2. D. W. Meseke, "SAFEGUARD Data-Processing System: The Data-Processing System Performance Requirements in Retrospect," B.S.T.J., this issue, pp. S29–S37.
3. W. H. Mac Williams and J. E. Petersen, "SAFEGUARD Data-Processing System: A Cost-Plus-Award-Fee Contract for a Large Software Development Program," B.S.T.J., this issue, pp. S237–S244.
4. D. Van Haften, "SAFEGUARD Data-Processing System: Software Change Control," B.S.T.J., this issue, pp. S231–S236.