SAFEGUARD Data-Processing System:

Software Project Management

By J. D. MUSA and F. N. WOOMER, JR.

(Manuscript received January 3, 1975)

A broad-based study of software project management for the SAFEGUARD project is presented. SAFEGUARD posed unprecedented project management challenges because of its size and complexity, yet the project was successful in attaining its objectives. This account of what some of the challenges were and which approaches were most effective in meeting them will hopefully suggest guidelines for the management of other software projects. Subjects include planning, methods for gathering status information, control actions, requirements, and programming methodology; also, differences between managing a software and a hardware project are explored. This study is based on intensive semistructured interviews with 26 SAFEGUARD software managers at all levels concerning their experiences on the project and opinions derived from them. The views expressed are limited to those of the individual managers interviewed and do not represent a consensus of SAFEGUARD management.

I. INTRODUCTION

For the purpose of this paper, "project management" is defined as *work planning and scheduling; gathering and reviewing status information*; and *controlling and allocating* human, computer, financial, and time *resources* so as to meet project objectives. Control consists of a continuing series of corrective actions resulting from status reviews. Project management differs from management in general in that technical questions or individual personnel matters are not considered, except where they might affect the areas within the definition of project management.

II. METHOD OF STUDY

Most accounts of management experience on a project are relatively personal ones. In this case, it was decided to obtain a broad perspective by interviewing a cross section of 26 software managers about their experiences. The interviews included subcontractor managers and in-

volved every level of management from first to fourth; and the participants held a wide range of jobs. There were varying degrees of background in software and management, and widely differing management philosophies.

A discussion lasting approximately two hours was held with each manager. An interview guide was sent to each participant one week before the interview. One-third of this guide contained background material; two-thirds, questions. The guide prepared the managers for the interviews, giving them time to consider the topics. Each interview was semistructured in the sense that the open-ended questions of the guide were generally followed; however, digressions into related topics were also encouraged.

The authors looked for patterns in the responses and held follow-up discussions with the managers based on the initial draft of this paper.

III. RESOURCES DEVOTED TO PROJECT MANAGEMENT

To gauge the importance placed on project management activities, each manager was asked what percentage of his organization's resources he was willing to allocate to this function. Estimates ranged from 5 to 25 percent, with the average about 12 percent. To get a job running smoothly, it was generally believed that more resources (up to 50 or even 100 percent) should be devoted to project management in the early stages of a job. On SAFEGUARD, the rapidly changing environment necessitated a high level of planning effort extending into the late phases of the job. Several people believed that the proportion of effort required for project management is larger on large projects; this was especially so on such a complex project as SAFEGUARD.

IV. PLANNING

System requirements were first defined in 1969 in a system concept paper; by the third quarter of 1969 they were specified at a detailed system engineering level in the data-processing system performance requirements.[1] During the same time period, a complete software development plan and schedule was prepared, along with the rationale on which it was based. This overall plan was followed by extensive planning in more detail at lower levels, much of it stimulated by the requirements of the Management Reporting System (MRS). The MRS is described in Section 5.2.

One of the most difficult challenges was establishing the proper time relationships between different parts of the job. When this was not done realistically, simultaneous design and coding often resulted; this was inefficient because then interfaces were not ironed out and the feasibility of algorithms was not investigated (and they finally had to

be). It was found that the timing of test planning activities was fairly critical. The optimum time appears to be approximately when program design is complete but coding has not started, since a good test plan should have both requirement- and implementation-oriented components. The point in time at which users had enough knowledge about their data reduction requirements to specify them in detail and the point in time at which the requirements were needed by the data-reduction system designers were usually incompatible. It was very hard to schedule system evaluation, since the design had to be far enough along to provide definitive information but the evaluation had to be completed early enough so that problems could be identified and corrections implemented before the design was frozen.

Several managers would have placed more emphasis on centralized planning. They suggest that a group, reporting directly to the highest-level manager, should be responsible for overall planning and schedule control. A formal development plan for the entire SAFEGUARD project had been prepared; however, almost everyone felt that this development plan was useful mainly for introductory orientation.

Despite a general feeling that planning is important, a slight majority of managers did not think it necessary to prepare written, explicit schedule and resource plans for their own areas. Most of them do not enjoy such planning; however, this distaste does not extend to technical planning, such as the generation of requirements and test plans. Those who feel that a development plan is necessary generally suggest a written plan (with format left to the author) plus supplementary bar charts. Most managers believe the PERT networks are not worth the effort required.

Accurate estimation of time, manpower, and computer time required for a job was found difficult by most managers. Estimation accuracy was not significantly affected by the size of a job. Several managers commented that they found it difficult to evaluate schedule performance because the work often changed after the original plan had been created. One noted that although computer time usage was the hardest resource to estimate, it was the least critical of all the resources to manage (except when "turnaround" time deteriorated).

When errors in estimating occurred, the prime sources of error were found to be neglecting learning time and underestimating the lengths of test intervals and the amount of maintenance support required. One manager felt that from 25 percent to 50 percent of the peak manpower of a project must be "kept" for the maintenance phase. The lower figure represents the manpower required to correct "bugs" but not to implement any changes, and the higher figure represents the manpower required to deal with a fairly high change rate.

Several participants felt that a better basis for estimating a job was previous experience on similar jobs rather than the use of numerical planning algorithms. Hence, their approach was to let their subordinates use their own direct experience to do the estimating and then question them on it.

V. GATHERING STATUS INFORMATION

Gathering status information is one of the most important project management activities. There was a fair amount of diversity in establishing the criteria that were considered to be most important in evaluating the worth of a particular report or information-gathering technique; however, timeliness and accuracy stood out. Definitiveness (providing enough information to identify the accomplished work unambiguously) ranked next. Several managers indicated that, above all, a report or technique should make problem areas visible. Conciseness was considered useful, but completeness ranked fairly low. Some minor considerations were flexibility and understandability. It was felt that asking the question, "What actions can I take as a result of this report?" was a good approach to measuring its value.

A very strong pattern of preference for informal methods emerged, with primary reliance on *oral* rather than *written* communications. This preference was found to be independent of management level. One argument advanced for oral reporting was that the useful life of status information is short. Another advantage of oral communication is that it permits questioning and probing and rapid interaction with feedback. Also, many managers believe that they can evaluate the reliability and accuracy of oral communication better than that of written communication because they can observe the way in which a respondent answers questions. Lower-level managers prefer making oral reports because it takes less time and involves more personal contact with their superiors. The main weakness of oral communication is that for higher-level managers it may be either indirect (i.e., received second-hand) or time-consuming.

Some managers found conventional written progress reports moderately useful as reminders for low-priority items or for keeping up to date on activities in peripherally related organizations, but most believe that written reports have a low density of useful information. (They may be inaccurate, heavily filtered or censored, or out of date.) Written reports usually only formalize communication that has already occurred. Finally, the formality of writing does cost time and money. This cost could be such that the need for any written report should be periodically challenged.

The types of information-gathering techniques used by different levels of management were generally not very different. There was,

of course, less need for detail at the higher levels. Higher-level managers were more interested in tracking the status of capabilities; lower-level managers more interested in components. Lower-level managers were very well satisfied with the information available to them; however, higher-level managers occasionally felt that some of their needs were not met. Most of the attempts made to provide better information to higher-level managers involved written reports. They apparently were only partially satisfactory in meeting the needs, since higher-level managers still primarily relied on oral communication.

The contractor's low-level managers had mixed attitudes about technical involvement of high-level managers. They appreciated the understanding that resulted, but believed that frequently there was too much concern with detail. They felt that this concern indicated a lack of trust and that it restricted their freedom to do the job in the way they thought best. It was suggested that higher managers sometimes dipped into detail simply because they enjoyed keeping technically involved.

High-level managers, in general, recognize some of the dangers that their subordinates cite and realize that there may be disadvantages in their concern with detail. They also know that keeping informed is expensive in terms of demands on the time of their subordinates. However, one of them pointed out that perhaps the disadvantages must be accepted as concomitants to the drive for technical understanding and that the advantages on the whole won out. One advantage, for example, was that technical understanding permitted rapid evaluation of situations and made for more immediate decisions.

Reporting techniques did not change much as a result of changes in the phase of a job, excluding reports obviously tailored for a particular phase, except that there was some tendency for more detailed information to be needed in the later stages. The program-design, code, and unit-test phases of a job, characterized by a large number of parallel efforts, were the most difficult to track. Attempts at numerical characterization of status were not always completely successful because available measuring units are generally nonuniform and are not sufficiently descriptive of problems. The system analysis and system design phases can be tracked by observing the status of the requirements and functional specification documents, and the system integration test phase by the completion of tests of various capabilities. There is a clear need to find a way of defining more intermediate milestones of a specific nature for the program-design, code, and unit-test phases.

It was noted that many persons receiving reports would like to have had direct control of the level of detail and format. In some cases, reports satisfied the needs of several people, but these "communities of interest" were generally small in size and on the same managerial level.

The persons making reports were sometimes unhappy about the duplication of data between reports, resulting from lack of standardization. This situation needs to be recognized as being generally unresolvable; good compromises may not always be possible.

Most participants felt that they needed information on a weekly basis in their area of responsibility. Information more than one week old has limited usefulness for solving problems. A monthly interval is considered sufficient for overall project status information external to one's area of responsibility and for computer usage and manpower and financial reports. (It was noted that the reason for the less frequent reporting of financial status was that dollars usually cannot be controlled very rapidly on a project.) Several participants believe that one needs daily information on a few crucial items, particularly if a problem exists, or during the final stages of a project.

5.1 Discussion and meetings

Informal individual discussions were by far the most commonly used data-gathering method. Managers found that individual discussions were generally more effective than meetings. However, there is often not enough time to talk about a problem with all the individuals concerned.

Meetings, usually held on a weekly basis, were the second most common data-gathering technique. Most managers found that meetings were more successful if they lasted no more than $1\frac{1}{2}$ to 2 hours, with an agenda prepared beforehand. A majority agreed that someone should be assigned to record and publish action items generated during a meeting. It was also agreed that specific problems should usually be "delegated" for later solution.

It was found that there are several common problems that occur in meetings that a manager must learn to deal with. Occasionally, most of a meeting may be taken up with "educating" high-level managers. Some participants may make contributions merely to make an impression. In other cases, meetings can become forums in which one manager tries to shift the responsibility for a problem to another manager. People may make unnecessary efforts to dig up problems just because they believe their managers expect them to bring them up. It was agreed that discipline should be exercised as to the frequency, length, and size of meetings; the principle of representation, rather than that of full attendance of all parties, should also be followed.

5.2 Written reports

Trouble Reports (TRs) were generally considered to be the most valuable written source of information on the project. Fairly early

during the SAFEGUARD project, the concept of writing a TR for a program malfunction was generalized to permit such reports to be written on documentation and requirements as well. This idea was apparently very well received, and TRs were widely used to report and record all sorts of discrepancies. After a program had been turned over for integration, the tracking of these reports was the method most commonly used by first-line supervisors for keeping track of program status. Several managers initially preferred local TR accounting systems to a centralized system, since speed in gathering information and meeting the differing needs for detail of different groups were their most important objectives. However, later in the project a centralized system was established which met these goals. Program-review boards were set up in several areas to evaluate TRs and approve or disapprove suggested program changes; they were found to be very effective.

A computerized Management Reporting System (MRS) was developed for use on the SAFEGUARD project. The system incorporated data bases for schedule, manpower, and computer usage information. The schedule data comprised some 3000 items, with information on the scheduled and estimated start and completion dates of various significant activities and events. Status information was provided, as well as indications of dependencies between the various items. Error-checking and data-interrelating capabilities were incorporated. On a monthly basis, information was updated and a standard report published. In addition, sorting and retrieval capabilities were provided so that a wide variety of special reports could be produced, on an as-requested basis.

The success of this system was mixed. Several managers, particularly at the higher levels, felt that MRS was of significant help in structuring and planning the project, in that it forced both long-range planning and the coordination of plans between different areas. On the other hand, the three-week time lag between gathering information and publishing it was considered too great. Experience indicated, however, that gathering, publishing, and distributing this much information (approximately 850 pages every month to 70 managers), with high standards of accuracy and good coordination of dependencies and dates across interfaces, can probably not be speeded up to any significant degree. Several managers did not like the discipline forced upon them or the background information lost in a fixed, computerized reporting system. They believe that managers should be allowed to choose the reporting method best suited for describing the status of their work.

A Principal Events Reporting System was developed which identified and carefully defined a number of important milestones on the SAFEGUARD project. Many of the events listed were the completion

of tests of various functional capabilities. Status reports on these events were made by TWX within 48 hours of their scheduled completion dates. Estimated dates were given for completion of late items, and follow-up TWXs were sent on the rescheduled dates. A principal events report, which described and gave the status of all items, including previously completed events, was issued at quarterly intervals. This system was primarily valuable to the customer and to higher levels of management; the TWX reports were particularly useful.

Program unit reports (reports indicating the status of the smallest program units and data sets in terms such as "design complete," "coding complete," "unit test complete," etc.) were considered to be of little value by most managers.

The weekly TWX status reports sent between the test sites and the contractor's home location were considered to be useful because of their timeliness and conciseness. The almost universal recognition given to the need for writing and handling a TWX expeditiously ensures that the information is timely. Furthermore, a TWX progress report must be concise; this ensures that only the most important items get reported. This seems to indicate that if a system of written progress reports is to be of any value, it should be severely constrained in both preparation time and length. (It might actually save money to *require* all written communication to be sent by TWX.)

Several managers used wall schedule charts but all eventually abandoned them as not being very useful, except possibly for initial planning. The majority felt that their wall charts did not shed any particular light on critical issues.

Managers primarily employed computer usage reports to spot trends or to make budget projections rather than for control. Manpower usage reports were not employed in manpower allocation decisions because so many other factors were more important.

Financial reports were used for reporting on expenditures. Several managers felt that more detailed information should have been provided as to the sources of the data and the date on which it was valid.

VI. MANAGEMENT REPORT ANALYSTS

One innovation that was introduced on the SAFEGUARD project was the assignment of a staff assistant, called a Management Report Analyst (MRA), to each second-level manager. The MRAs were, in general, college graduates with backgrounds in planning, scheduling, and budgeting. Besides acting as general "right-hand assistants," they aided the managers with budget preparation and control, planning and scheduling, interfacing with overall project management report-

ing systems and following up on action items. Almost all of the managers who were assigned MRAs were enthusiastic about their usefulness. A high-level manager noted that MRAs made it possible to accelerate schedule-information flow. Some of the managers felt that the MRAs saved them time by buffering them from duplicate requests for information.

There seemed to be no need to assign MRAs to first-line supervision on a full-time basis; sharing the MRA assigned to the second-level manager was satisfactory. One high-level manager said that he felt that the usefulness of MRAs was such that they also should have been assigned to the third-level managers. It might be noted that the fourth-level managers were already assigned staff assistants.

The comment was made that MRAs were most beneficial when they were assigned to report directly to a second-level manager rather than to a central group supervisor. However, some managers thought it was best to centralize the physical location of the MRAs so that there could be interchanges concerning common problems, solutions, interfaces, and action items. Also, centralized training was felt desirable.

VII. CONTROL

Control relates to the corrective actions that result from the process of comparing status to plan. The size and complexity of the SAFEGUARD project made prediction of the likely consequences of various actions difficult at times, complicating the process of selecting from among the alternatives.

Good organization, adapted continuously to changing job requirements, was found essential to the successful implementation of management actions. As a by-product, it was observed that the amount of status reporting and communication required was substantially reduced when the organization was well fitted to the tasks to be accomplished and responsibility was not divided. Interface and system-coordination departments had to be in the mainstream of activity and authority to function effectively. Some managers would have created a more clearly hierarchical organization. This might obviate the problem (common among managers with strong technical orientation) of multiple levels of management working on the same problem at the same level of detail. Others felt that *conflicting* needs in a complex project require *conflicting* organizations; consequently, informal organizations and informal channels of communication should be encouraged.

Most of the managers believe that interfaces on the project were handled well or very well, but many agreed that specific improvements could have been made. Some of the interface areas that were initially

significant sources of problems were the contractor-subcontractor interface; the interface between support software users and designers; the interface between system engineering and development; and the interface between users and support service activities, such as the computation center.

Most managers found that personal contact and meetings are the best ways to coordinate interfaces; letters of agreement were considered to be relatively ineffective (partly because of reorganizations), although they were of some value when used with subcontractors. Even where organizations are geographically remote, personal contact is preferred. (For example, coordination with even our Pacific site was found to work much better by phone than by TWX or letter.) It is recommended that documentation should be used only to confirm and record agreements after the fact.

7.1 Control of time

Most managers noted that although only slight deviations from any scheduled end dates were acceptable to their superiors (the average of estimates of acceptable deviation is five percent), they had (and should have) relative freedom to change intermediate schedules.

When tasks could not be finished according to plan, the most commonly employed strategies for recovery were to work personnel overtime or use extra computer time. Increasing manpower on the job or decreasing the scope of the task were secondary choices. Slipping schedules, decreasing or deferring documentation (surprisingly), and decreasing the quality of testing, in that order, were considered to be increasingly undesirable. (One manager observed that decreasing testing is a very bad option because people have a way of forgetting they agreed to reduce testing when a program doesn't work.)

Several managers found that overtime was ineffective except in urgent situations, because people tended to get stale. This is felt to be particularly true in creative jobs.

Many of the managers concluded that adding manpower to a job is usually not a useful technique. Even if budgetary constraints do not exist, suitable people usually are not available at the critical point of the job. Adding people generally hurts the effort because of the training required at such a late stage in the project. (A few areas did add people effectively late in the project but they had special skills as trouble shooters.)

Managers generally determined that, when schedule changes are necessary, it is best to consider all the inputs at one time and do a complete revision. Complete revision permits the changes to be carefully thought out and documented. (One high-level manager said

he felt that some of his principal contributions to the project were vetoes on changes.)

It was generally agreed that the MRS approach to controlling schedule dates was a good one: high-level managers controlled the schedule dates for events, while the supervisor responsible for an event provided his own estimated date. Many managers believe that a subordinate should only be held to "end" dates (i.e., the dates at which deliveries to another organization must be made). A subordinate should be required to make his intermediate dates visible, but should be permitted to modify them as he desires.

7.2 *Control of human and computer resources*

Allocating resources efficiently was found to be most difficult at the higher management levels. It was hard to gain detailed insight into how various activities contributed to the real goals of the project. Activities, particularly in the support area, tended to go on "forever" unless their value was questioned. Most lower-level managers considered that they had the knowledge and the freedom to allocate manpower and computer resources productively within their own areas.

There is general agreement that selecting good people was the factor of greatest importance in the success of SAFEGUARD. Selection must be considered not only as an initial choice but also as the continuing process of assigning people to jobs and problems. The generally flexible, informal management style that prevailed on SAFEGUARD aided management greatly in this process, in that there was a lot of "self-selection." Large numbers of nonmanagers exercised initiative in digging out and solving problems. The strong technical capability of project members occasionally led to excessive technical discussion, design polishing, and uneven work coverage due to a concentration on technically interesting problems (to the detriment of important but less rewarding ones). These difficulties were small, however, compared to the advantages.

Although many of the programmers were inexperienced, there was a cadre of people with three or four years experience (at the start of the project) who became the lead programmers and in some cases first-line supervisors. Previous background of the contractor, background gained with similar systems, was also valuable. Inexperience occasionally resulted in some errors of judgment. However, one of the surprises of this study was that inexperience had a relatively minor overall effect on the project.

The shortage of experienced software managers on the project posed a more serious challenge than the shortage of experienced programmers. It was found that if there was not enough supervisory attention given

to programming, both efficiency in attaining objectives and quality of output sometimes suffered.

A "software mystique" can discourage managers who have no software background from applying some of their general managerial skills; hence, many skills may be lost.

Turnaround time was the key parameter that was monitored in the control of support computer resources, since it had maximum impact on schedules, overtime costs, and programmer satisfaction. Allocation of support computer time was found to be worthwhile only when turnaround time deteriorated, i.e., it did not pay on a regular basis.

VIII. REQUIREMENTS

Overall success in various areas of the project was considered to be strongly correlated with the degree to which project system requirements were clearly and completely defined, written, and stabilized. It was found to be necessary to focus on the feasibility of implementation of requirements and routine details such as interface coordination as much as on the requirements themselves. (Simulations of algorithms on a support computer were determined to be very beneficial.) It proved to be very challenging to pick the right level of specification of detail without unduly restricting the designer's freedom to apply his special competence. It was suggested that it might be useful to have a high-level requirements document for the customer and a more detailed one for the developer. One pitfall to avoid is the incomplete requirements specification, particularly with inexperienced personnel, who often will not recognize the deficiency early enough to request timely corrective action.

Some of the areas that the development managers believe should have received more attention in the system requirements are:

  (*i*) Error control.
  (*ii*) Interface specifications.
  (*iii*) Requirements for equipment tests.
  (*iv*) Maintainability, reliability, and availability.

Coordination of software requirements with hardware changes was found to be very important.

One suggested way of achieving greater focus of system engineering on implementation is to place senior designers in the systems groups during the first part of the project, so that they can make the systems engineering people more aware of their detailed needs. As a system is defined and as more detailed development starts, designers may return to their development groups.

IX. PROGRAMMING METHODOLOGY

Another surprise in this study was that the lack of well-developed methodology in the programming art did not turn out to be a serious problem. Managers did comment that lack of an established technology meant that extra time had to be spent in experimentation, which impacted on cost and schedules.

There are a number of things that managers believe that they learned in the area of programming methodology. Several managers noted the need to keep debugging and testing in mind during design. For example, one must consider recording requirements and clarity of code. The idea of holding design reviews with flowcharts was a technique that many managers tried and found very beneficial. A number of managers became strongly convinced of the virtues of structured programming, considering it to be extremely important in the maintenance phase.

Another attitude that changed during the course of development was the attitude toward patches. It was originally planned to make all changes by altering source code, so that program listing documentation could be kept under good control. It was found, however, that the amount of recompilation and relinking necessary for a large program made this approach impractical for priority changes or for fixing bugs during the testing process. It was more practical to place "alters" in the source code and recompile and relink at less frequent intervals. Consequently, it became very important to provide good patch facilities and good procedures for documenting patches and keeping them under control.

Many managers found that the best documentation for programs was a well-commented listing. This represents a change of opinion on the part of a number of managers who had first seen some value in flowcharts and narratives, but who later found that few people used them in the maintenance of programs. Flowcharts appear to be better design tools than program-maintenance tools. Narratives appear to be of value primarily to system evaluators.

Several managers believe that more attention should be given to developing good unit test tools early in the development cycle. This philosophy of independent testing (i.e., test cases generated and tests conducted by groups other than the original design organizations) was widely used on the project and in general was quite successful.

There is general agreement that using a local "duplicate" computer facility for checking out programs prior to shipment to site was not only useful, but was in fact necessary to the success of the project. Using a support computer for simulations of algorithms, system performance measurements, etc., was very effective.

There were some comments that standards for programming practices should be specified to a uniform level of detail. Several managers believe that standards should be *function*-oriented rather than *format*-oriented; i.e., different formats should be permitted provided they satisfy the objectives of the standards.

X. DIFFERENCE BETWEEN MANAGEMENT OF SOFTWARE AND HARDWARE PROJECTS

All participants were asked how managing a software project differs from managing a hardware project. In general, about half the managers believe that there is nothing essentially different, and the other half see differences ranging from minor ones to everything being completely different.

The largest number of noted differences occurred in the area of development methodology. It was observed that there are more variables in software than in hardware development. It was felt that good engineering tools plus the constraints of physics will ensure a reasonable hardware product, but good programming tools will not ensure a good software product (it appears to be difficult to set up enough worthwhile constraints). Programming is more of an art than a science at present, and software design is often influenced by a personal approach to the problem. There seems to be a much tighter coupling between software development and the entire system-integration process than between hardware development and the system-integration process. This has been caused not only because logic and control is mostly implemented in software, but also because hardware lead-time constraints force system trade-offs to be made with software to a greater extent than with hardware. Software design was considered to require more time because software is usually more complex. However, in hardware design, each element is usually more thoroughly designed because it will be mass-produced.

The management of changes was another large area of noted difference. It was considered that software is generally more volatile than hardware and is more vulnerable to external influences. Several managers believe that for software there is less understanding of the impact of change on schedules and costs on the part of customer, manager, and programmer. The lead time required for changes in hardware is well recognized, but it is not in software.

In the area of personnel, it was noted that technical competence is both much more important and harder to evaluate for software because the design technology is not well developed. There is a shortage of people with both the programming and hardware backgrounds necessary to a good system perspective.

Quality control is at present more difficult for software; the criteria for success (what is a good program?) are less well defined. The volume of a worker's output is much greater in software and, thus, a supervisor cannot examine his work very extensively. (For example, a typical hardware group might produce 12 hardware circuits in a year in contrast to 30,000 instructions from a software group.)

Other comments were that software subcontracting is more difficult than hardware subcontracting because the requirements tend to be more variable, management of software development is more of an art, and estimation of the duration and size of hardware jobs is more accurate.

XI. CONCLUSIONS

What major lessons may be drawn from the SAFEGUARD software experience by prospective managers of other software projects?

(*i*) There appears to be great virtue in maintaining a flexible, informal, participative style of management.

(*ii*) Selecting good people and matching them to the right functions and problems are probably the most important management functions.

(*iii*) Informal, oral approaches to reporting status seem to work the best; written reports should be kept to a minimum. When used, they should be strictly constrained by length, time deadlines, and very hard-headed analysis of their purposes.

(*iv*) Defining and tracking concrete events based on functional capabilities, as exemplified in the Principal Events Reporting System, was found particularly useful by higher-level managers.

(*v*) Informal status reporting should be balanced with carefully prepared and written requirements and test plans and good configuration control of both requirements and code.

(*vi*) Using project management specialists as staff assistants for managers was found to be a very productive technique.

XII. REFERENCE

1. D. W. Meseke, "SAFEGUARD Data-Processing System: The Data-Processing System Performance Requirements in Retrospect," B.S.T.J., this issue, pp. S29–S37.