INTRODUCTION
The problems of managing programmers have been growing rapidly in recent years. The specific issues referred to are the increasing costs of programming and the failure to complete projects when scheduled or needed. This paper proposes more effective supervision of programmers together with improved management policies and procedures as steps to reduce the effects of these growing trends in programming projects. These problems have resulted, in part, from the shortage of qualified personnel, the rapidly changing technology, and the management personnel in the computer center. This paper does not deal with the problem of project management, that has been dealt with elsewhere, but rather the supervisory processes involved in the management of computer programmers.

The shortage of qualified personnel
One of the fundamental problems facing the manager of a programming group is the shortage of qualified personnel. The number of installed computers is rising rapidly with a variety of estimates being made for significant future growth. Regardless of whose estimate is accepted the fact remains that an increasing number of computers will be installed and operating in the near future. We cannot expect, in the short run, to find a decreasing demand for programmers. It has been estimated that at the present time there is a shortage of over 50,000 programmers in the United States. A recent major study of the problems of the computer industry estimated that by 1980 there will be a need for 200,000 programmers. These figures have led Brandon to estimate that by 1980 there will be a net shortage of about 90,000 programmers. Part of the staff shortage problem would be relieved if the educational institutions were to provide the proper education to a large number of students. In a study of computers in higher education Hamblen found that in 1964-65 nearly 120,000 undergraduates and 29,000 graduate students received some computer training. In that same time period it was estimated that approximately 4,000 undergraduates and 1,300 graduate students were majoring in computer science. By 1968-69 it was estimated that 81,000 graduate students and 350,000 undergraduates will have training in at least one programming language. At that same time the number of students whose major area of study is computer science will increase to 19,000 undergraduates and over 5,000 graduates, see Table I.

Table I—Estimate of the number of students being trained to use computers

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Graduate</td>
<td>Undergraduate</td>
<td>1964-5</td>
<td>1968-9</td>
<td>1964-5</td>
</tr>
<tr>
<td>Computer Science majors</td>
<td>1,314</td>
<td>5,318</td>
<td>4,338</td>
<td>18,807</td>
<td></td>
</tr>
<tr>
<td>Other majors (with at least some skill in using one programming language)</td>
<td>25,800</td>
<td>80,793</td>
<td>119,092</td>
<td>350,178</td>
<td></td>
</tr>
</tbody>
</table>

Source: Reference 6

The response of the educational institutions to a national problem, as reflected in these figures, is impressive. But before concluding that in a few years the problems of shortages will be resolved by adequate supply it must be realized that there will be intensive
competition, as there now is, for those students who have majored in computer science. In addition it will require substantial incentives to motivate the over 400,000 students who have had training in at least one programming language to leave their own field of major interest and become programmers. Even for those who are lured into computing further intensive training will be needed to bring them up to an acceptable level of competence. Therefore it may be found that in the next five years a labor pool exists of college graduates with some training in programming. The shortage of qualified and experienced programmers will still exist and perhaps be even more severe due to the need for training and supervising these new entries into programming. There also has been a rapid expansion of non-degree granting programming schools both private and public. It is too early to appraise the effect this source of supply will have on the labor market.

It should be noted however that even though we can expect staff shortages to exist in one form or another for some time to come there is something that can be done to relieve the problem. That is to increase the productivity of programmers. If programmer productivity were increased by a modest amount, say 10 percent, it would, based on estimates of 200,000 programmers, be the equivalent of 20,000 additional programmers being made available. This would be 40 percent of the estimated current shortage of 50,000 programmers. The techniques suggested in this paper have been used to increase programmer productivity and it is toward that end that they are presented.

Rapidly changing technology

The past changes in computer technology are well known and need not be repeated here. Of importance, however, are the changes that may take place which affect the technology of programming. Such technological developments include languages, compilers, operating systems, documentation aids, etc. In the past a change in hardware technology has generally brought with it changes in the programming technology. The result is that additional time and money must be invested for training so that the staff can stay abreast of the latest developments. These types of changes are mandatory because the new hardware technology has forced them on the user of the particular computer. A second category of optional, or non-hardware related, changes exists. These are those technological developments which result from attempts to achieve maximum exploitation of the computer. The training of programmers in these techniques is not mandatory but frequently is highly desirable. The net result of these two types of changes in technology is that continuous training for programmers is a way of life for effective program production.

Even now there is some indication that the next generation of computers may not be program compatible with those currently available and in use. An executive of a major computer manufacturer has been quoted in an article as saying that his company would not hesitate to ask the data processing community to “reset-to-zero” and accept program incompatibility “if something comes along we think is a substantially better way to get price/performance into the market.” It therefore appears that still another cycle of programmer retraining and program conversion may be forthcoming.

The computer center manager

Fundamental to the solution of the problem of better management of programming efforts is the manager of the computer activity. By computer manager we mean the highest level person in the organization who devotes all of his time and energy to the problems of computation. This definition avoids having to enumerate the myriad titles associated with the position. In a study of the managers of large computer centers it was found that these managers were relatively young (41 years old), well paid (average salary of over $20,000 per year) and well educated (typically a second degree). Of significance is the fact that only 12 percent of those sampled preferred only technical problems, 40 percent preferred a combination of technical and human problems, and 48 percent indicated that the principal problem they preferred was human. Educational background had a slight effect on these average results, Table II. When asked which problem they face gave them the greatest difficulty 42 percent indicated personnel, Table III.

One of the reasons that can be attributed to the manager's personnel problems is the rapid growth of the industry. As a result of this growth many persons have been forced into managerial positions without adequate preparation. This lack of prior managerial experience is most noticed when it is observed in a number of computer centers that there are inadequate control systems for reporting on the operations of the department or organization. The quintessence of management control is the planning of activities, delegation of authority, staffing the organization and control of plans over a period of time. This has not been generally done in computer organizations.

It is therefore proposed that the problem of managing computer programmers starts with the senior manage-
Table II—Education vs. problem preference

<table>
<thead>
<tr>
<th>TYPE OF PROBLEM PREFERRED</th>
<th>Percent in Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Education</td>
<td>Technical</td>
</tr>
<tr>
<td>Technical degree or degrees</td>
<td>14%</td>
</tr>
<tr>
<td>Technical degree plus M.B.A. Business and Liberal Arts Degrees</td>
<td>0%</td>
</tr>
<tr>
<td>Percent</td>
<td>12%</td>
</tr>
</tbody>
</table>

Source: Reference 8

Table III—Problems facing the computer manager

<table>
<thead>
<tr>
<th>PERCENT</th>
<th>PROBLEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.6%</td>
<td>Lack of Funds</td>
</tr>
<tr>
<td>42.2</td>
<td>Personnel</td>
</tr>
<tr>
<td>18.4</td>
<td>Education of Top Management</td>
</tr>
<tr>
<td>18.4</td>
<td>Keeping up with User Needs</td>
</tr>
<tr>
<td>15.8</td>
<td>Long Range Forecasting of Computational Requirements</td>
</tr>
<tr>
<td>2.6</td>
<td>Cost Control</td>
</tr>
</tbody>
</table>

100.0%

Source: Reference 8

Management of Computer Programmers

The programming manager

Unfortunately little is known about programming managers. A great deal can be conjectured based upon an examination of the individuals performing this job. First, and perhaps most important, is the process by which a person may be appointed to this position. The logical, but unproven, concept that the best manager is the one who has done the job for a reasonably long period of time leads many to look to their most experienced programmers to find a new manager, or supervisor, of programmers. Little or no attention is paid to the individual’s suitability for a position of leadership. The individual, on the other hand, is faced with the dilemma of leaving technical work for the higher pay and status of a management position. Few firms offer the much talked about dual career paths for professionals and managers. Even among those who do there are frequently differences in salary ceilings for the person who chooses technical rather than managerial work. The result is that the first-rate programmers are frequently motivated by financial factors into becoming second-rate managers.

Steps toward change

While it is possible to identify a number of causes of present, and probably future, problems there is no magic formula for an effective solution. It is proposed that through the intelligent and orderly application of management techniques steps can be taken to reduce the impact of these problems and provide a basis for further improvements. The point of view taken is that the manager of a group of programmers is responsible for the quantity and quality of work produced. He is also responsible for meeting schedules. The techniques available for him to use in achieving these goals center about the management process in general and include such topics as supervisory techniques, work assignments, evaluation of output, documentation, work habits, leadership, and evaluation and review in particular.

The management process

Reynolds has pointed out that while the management of programming is similar to the management of other technical activities it differs in that there are few numerical indices that can be used to judge progress in a programming project. There are the following steps in the management process:

1. Establish a plan of action to achieve a goal within a certain time period and investment.
2. Measuring, during the course of the project, the performance against that goal.
3. Making evaluation and decisions each time the plan and the actual performance mismatch.
4. Taking corrective action iteratively until the job is done.

The establishment of a plan requires that there be sufficient understanding of the task so that it can be broken down into sub-tasks and using a procedure such as PERT an overall strategy developed. If this level of knowledge about the project is not known it must then be considered to be a research and development effort and so treated.

The second step, that of measuring during the course of the project actual performance and comparing it with the planned achievement, depends on an effective reporting procedure. The system of reports can be related to, or even a part of, the system documentation as is frequently recommended. Reports are meaningful only if they are related to the benchmarks or steps used in the planning process. For evaluating the effectiveness of the ongoing programming effort the use of PERT/COST has proved of value.

The last two steps of the planning process follow logically after the data is available. What steps are taken should performance not equal that anticipated by the plan is dependent on many factors. There are, of course, good reasons for deviation such as changes in specifications but there are also reasons such as failure to produce at the expected level which represent problems in personnel management or estimating. The corrective steps taken are a function of the problem and its likely cause. In these last two areas the manager’s knowledge and ability come to the foreground. Since the first two steps are somewhat mechanical and have been dealt with elsewhere attention can best be focused on the problems of the manager in his supervision of programmers.

Supervision

The point has been made that the management of programming is another example of the more general problem of the management of technical personnel. But when this general statement is made it is often forgotten that the management of a technical function usually requires two skills. The first is an intimate knowledge of the specialized skills involved and the second is a knowledge of more general managerial skills. In such an environment as that under discussion it is questionable whether an effective job can be done with only the second. Too many have tried to be effective with only the first, technical skills, only to find themselves faced with many difficulties. The programming manager, like any other manager, requires the combination of these two types of skills to perform his job.

Programming is still largely an art rather than a science. The programmer therefore learns his trade from experience either on the job or in a classroom. In either case there is only a small, but hopefully growing, body of theoretical knowledge that is available. The programming manager must therefore have served an apprenticeship to have learned his trade. These specialized technical skills, it will be seen later, are essential for the manager of programming. The general management skills required can be learned in the traditional fashion by attending courses, on the job training, tutorial sessions with experienced managers, and others. While it must be acknowledged that few men become managers as a result of courses and training alone it is also true that there is a body of knowledge about management that can efficiently and effectively be taught. A well instructed individual can then apply this knowledge and start on a management career.

It has been emphasized in this paper that the programming supervisor is responsible for programmer management and productivity. We therefore now examine some of the steps he can take to insure that these goals be achieved.

Work Assignments. The programming manager should define the job to be done by each person under his control. Ideally this work assignment will be a well defined program step which is to be written, tested, and documented. A good work assignment will require minimum interaction between programmers implying that interfaces, fields, data names, and other such matters have been defined prior to any work assignment having been made. Further, the programmer should be expected to confer with his supervisor on any question of technical details that arises.

It is not always easy to divide a large and complex program into modules that can then be assigned to a programmer. Alt contends that it is the inadequacy of natural language that creates problems for the programmer because it is not a good language for communicating and specifying all possible combinations of circumstances that relate to the problem. The formalisms for problem definition and description which are required are not now available. As a result we must be aware of the problem and continue to use the module as the work assignment.

Evaluation of Output. The manager has the responsibility to evaluate both the quantity and quality of work produced by his subordinates. To do this the
supervisor should review the programs that are written. In this way he can be aware of the amount of code written and he can evaluate the quality of the work. In these evaluation sessions the manager can tutor the programmer, as required, in more advanced techniques or alternative approaches. The implication of this is that each programming supervisor have relatively few subordinates whom he can closely supervise.

It has been proposed that productivity standards be used for purposes of evaluation. From a practical point of view this is difficult because all the variables that affect productivity are not known. Martin has pointed out that while there has been found an average figure of 12–15 instructions per man per day the range is from 2 to 50. Even if standards were available Canning questions their use from an emotional point of view. He points out that they would place the programmer in the same category with factory and clerical workers. Undoubtedly this is true but the needs to increase productivity and reduce costs are as great in programming as in the office or factory. When such standards are developed they will be one of the important tools of the programming supervisor. At this time it is difficult to envision how a meaningful and accurate set of such data could be collected.

**Documentation.** Good documentation of a computer program is not an activity to be tagged on to the end of a project. Rather it is an ongoing activity with documents being developed at each stage of the programming process. The manager, therefore, is responsible for establishing documentation procedures and then perhaps more importantly enforcing them. It is interesting that Hill reports that resistance to documentation is encountered by all but the most experienced programmers. Those with one to four years experience being the most resistant. If the programming department has standardized its documentation procedures then the supervisor’s principal function in this area is to make certain that it is being properly carried out.

**Work Habits.** Amongst some programmers it is considered good practice to flaunt any or all work rules. The argument is advanced that programming is a unique creative activity that should not destroy personal values. The actual amount of creativity involved in programming is questionable. Further there is no reason why work rules should destroy initiative and creative urges in programming any more than in other comparable fields of activity. In many architectural and engineering offices reasonable work rules have been established and enforced and some of these people turn out very worthwhile work. The programming manager can be expected to establish work rules which are then enforced.

**Leadership.** Perhaps the most difficult job of the supervisor is to lead his subordinates. Leadership processes can be discussed in classes and written about in books but it is difficult to define. Each supervisor must develop a leadership style that fits his personality and life style. He can be made aware of the need for leadership and be told about how others have done it, but in the final analysis this is one of those intangible factors that each person must develop on his own. Failure to do so should result in a serious question as to whether or not the person is qualified for a higher position or even to continue in his present job.

**Employee Evaluation and Review.** It is common practice in many organizations for supervisors to meet with each subordinate and review the persons performance for a period of time. The review may take place annually or semi-annually but it is required and is then reviewed by managers higher in the organization. Formal employee evaluation is relatively rare in programming departments and should be encouraged. To aid the evaluator a form is used which can serve as a checklist to guide the discussion and provide a permanent record of the interview. One such form has been developed and published.

**Employee reaction**

It has been said that employees welcome controls such as those advocated in this paper. With programmers this is not true, resistance, perhaps even massive resistance, can be expected. Part of this resistance is based on the folklore that has grown up about the field of programming and the behavior patterns that are legitimate for programmers. At one time in the history of computation it might have been true that such employee controls were unreasonable. But today such supervisory techniques when used with intelligence and understanding are not only desirable but essential for controlling activities.

**CONCLUSIONS**

The program of management controls recommended in this paper have as an objective to increase programmer productivity. Collectively they imply firmer controls over the actions and behavior of programmers. Undoubtedly some programmers will be unwilling to accept this condition and will leave. Their effectiveness will be largely a function of the intelligence with which they are used. In all cases it is recommended that all processes discussed be applied. But the emphasis may well change with the experience of the people involved and the type of programmers being supervised. Finally
these recommended procedures are complementary to, and not substitutes for, good project control procedures.

REFERENCES

1 C P Lecht
The management of programming projects
American Management Association Inc New York 1967
2 P B Hill
The control of large-scale software projects
Paper presented at IFIP WG3.2 Workshop on Curriculum
For Systems Design Fribourg Switzerland January 28-31 1969
3 G Bylinsky
Help wanted: 50,000 programmers
Fortune Magazine March 1967
4 The state of the information industry
AFIPS Report 1966
5 D H Brandon
Managing the economics of computer programming, the problem in perspective
Proc A C M Conference 1968
6 J W Hamblen
Computers in higher education
Southern Regional Education Board Atlanta 1967
7 4th generation may be incompatible
Computersworld February 12 1969
8 M H Gottfried
A profile of the computer manager
Proc Fifth Annual Computer Personnel Research Conference 1967
9 C H Reynolds
Managing the economics of computer programming, the problem in perspective
Proc A C M National Conference 1968
10 F L Alt
Computers—Past and future: The costs of computing; and failure in computer programs
Computers and Automation January 1969
11 D H Brandon
Managing the economics of computer programming, the problem in perspective
Proc A C M National Conference 1968
12 J Martin
Programming real-time computer systems
Prentice-Hall Inc 1965
13 Managing the programming effort
EDP Analyzer Canning Publications June 1968
14 P B Hill
The control of large-scale software projects
Paper presented at IFIP WG3.2 Workshop on Curriculum
For Systems Design Fribourg Switzerland January 28-31 1969
15 R A Dickmann
A programmer appraisal instrument
Proc 2nd Annual Conference Computer Personnel Research Group 1964