SOME COST CONTRIBUTORS TO LARGE-SCALE PROGRAMS

Burt Nanus and Leonard Farr
System Development Corporation
Santa Monica, California

INTRODUCTION

In the early days of computer technology, only a small handful of highly competent, scientifically-oriented researchers were familiar with the programming arts. In those days, the management of programming effort was only slightly different from the management of other types of research activities. Each project was unique and its probability of success uncertain; experience was severely limited; tools and techniques were custom-built for each job. In the computer field today, despite many thousands of man years of experience in program development, we still tend to plan and to manage as if each program were a unique research project. This is partly due to the immaturity of the field, and partly because we have not yet fully recognized that the similarities between computer programs and their development are far more extensive than the differences between them. We have seen the development of many new tools to make the programmer more effective in his work; we must now ask whether we can develop new tools to make the program manager more effective in planning and organizing his scarce resources of talented manpower and expensive computer time.

One of the most important requirements for management planning is an accurate estimate of the resources required for the completion of the project. In programming management, the two principal resources to be estimated, scheduled and controlled are man months and computer hours. Together, these resources may be considered the cost of producing the program. Historically, these costs have been very poorly estimated; there are abundant examples of actual costs that exceeded estimated costs by 100 per cent or more.

Because better cost estimation is an important step toward more effective programming management, because the costs of programs may be a significant portion of the total costs of large management or command information systems and because the estimates have been little better than guesswork to date, the Advanced Research Projects Agency of the Office of the Director of Defense Research & Engineering sponsored some research in this area at the System Development Corporation (SDC). Early efforts were aimed at data collection and analysis of several large-scale command and control programming efforts representing a total of more than two million instructions and 1500 man years of work.2, 3, 6, 7 This paper is a summary of a subsequent effort to (a) identify the common factors that influenced the cost of developing programs and (b) perform a preliminary analysis on some of the data.* These are viewed as necessary first steps toward the development of a more accurate cost esti-

* The full research report upon which this paper is based will be published as TM-1447, Factors that Affect the Cost of Computer Programming, System Development Corporation.
mating procedure. Further progress depends upon more systematic data collection, and one of the purposes of this paper is to recommend the types of data that should be collected by programming managers for estimation purposes.

PROBLEMS ENCOUNTERED IN DETERMINING COST FACTORS

In the process of identifying and analyzing cost factors, one soon encounters a number of limitations in the programming field that may not exist in other, more mature disciplines. Some of these problems include the following:

1. **Lack of Agreement on Terminology**—There are no universally acknowledged definitions of many of the terms used in the computer programming process. For example, the words “debugging,” “parameter test” and “program validation” may all describe the same process; a “programmer” in one organization may be called a “coder” in another and a “system analyst” in a third. Although we attempted to keep within the more limited context of command and control programs in our research, we found, even in this narrower field of programming, a widespread lack of agreement on terminology.

2. **Poor Definition of Product Quality**—Apparently there has been little success in defining those attributes that characterize the nature or the quality of a computer program. For example, one hears programmers talking in terms of flexibility, economy of memory, and maintainability, but there seems to be no generally agreed upon criteria for comparing programs on the basis of these attributes.

3. **Poor Quality of Cost Data**—Present cost collection methods seem to be designed primarily for accounting purposes and not for planning or control. For example, the collections of costs are usually grouped by organizational units rather than by product or function to be performed.

4. **Dynamic Nature of the Field**—Although computer programming is maturing as a discipline, there is still a wide diversity of techniques and approaches being developed and used. As a result, any study of cost factors must consider the history and likely future trends of programming technology.

5. **Nonquantitative Nature of Some Factors**—Experience has shown that many of the factors that affect the cost of computer programs are qualitative in nature. In some cases, it is possible to predict at least the direction that cost will be affected by an increase in a given factor. For example, one would expect that the more experience one had with the particular type of program or computer involved in a given task, the less it would cost to perform that task. In other cases, qualitative factors appear to have a nonmonotonic effect as when an increase in a given factor (e.g., management planning) first decreases and then increases total cost. Of course, determination of the magnitude of the effect on cost of qualitative factors is even more difficult than determining the direction of the effect.

Although these problems combine to make an analysis of computer programming cost factors somewhat difficult, a start must be made if program development efforts are to be more effectively planned and managed.

COST FACTORS

It is possible to identify hundreds of factors that contribute to the cost of computer programs, if such a level of detail is desired. In this paper, we will present a list of approximately 50 such factors, consolidated from a much larger list. Obviously, some classification scheme is necessary for discussion or analysis purposes. Factors might be grouped by work phase, such as program design or test; by management activity, such as planning or evaluation; by general categories such as resources, requirements, or environment; by units of cost measurement, such as man months or dollars; or by the classic accounting method of direct and indirect costs. However, these schemes seemed to cause difficulties because of ambiguities and overlap; as a result, a new classifi-
A cost classification scheme was developed and is illustrated in Table I, in which factors are divided into these categories: The Job to be Done, The Resources that are Available, and The Nature of the Working Environment.

Since a full discussion of all of the factors is not possible here, only the most important one or two in each category will be briefly presented; the complete list is included as an appendix.

**TABLE I—COST FACTOR CLASSIFICATION SCHEME**

<table>
<thead>
<tr>
<th>Logical Grouping</th>
<th>Category Name</th>
<th>Category Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>THE JOB TO BE DONE</td>
<td>1. Operational Require-</td>
<td>Includes cost factors associated with the operating characteristics of the system for which the program is being written.</td>
</tr>
<tr>
<td></td>
<td>ments and Design</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2. Program Design and</td>
<td>Includes cost factors associated with both support and operational programs as determined by the constraints imposed by personnel, hardware and operational requirements.</td>
</tr>
<tr>
<td></td>
<td>Production</td>
<td></td>
</tr>
<tr>
<td>THE RESOURCES THAT ARE AVAILABLE</td>
<td>3. Data Processing Equ-</td>
<td>Includes cost factors associated with the data processing equipment required to produce and test a program, including all input, output and peripheral equipment.</td>
</tr>
<tr>
<td></td>
<td>ipment</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. Programming Person-</td>
<td>Includes cost factors resulting from the direct labor needed to completely develop a program.</td>
</tr>
<tr>
<td></td>
<td>nel</td>
<td></td>
</tr>
<tr>
<td>THE NATURE OF THE WORKING ENVIRONMENT</td>
<td>5. Management Proce-</td>
<td>Includes cost factors associated with the plans, policies, practices and review techniques used in the administration of all phases of program development.</td>
</tr>
<tr>
<td></td>
<td>dures</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6. Development Environ-</td>
<td>Includes cost factors resulting from relationships of the programming staff with other organizations, such as customers and other contractors.</td>
</tr>
<tr>
<td></td>
<td>ment</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7. Facilities, Services</td>
<td>Includes cost factors related to supplies, physical plant, indirect labor and overhead.</td>
</tr>
<tr>
<td></td>
<td>and Supplies</td>
<td></td>
</tr>
</tbody>
</table>

**THE JOB TO BE DONE**

1. *Operational Requirements and Design*

The factors in this category tend to center around the question, “How well are the operational requirements of the information system known?” or, “How well is the problem defined?” Unfortunately, it is virtually impossible at the current state-of-the-art to evaluate the cost-contributing effect of many of the factors which relate to this question.
The primary cost factor in this category would appear to be the extent of innovation in the system, in its components, and especially in the automatic data-processing function. The extent of similarity of the new system to older systems may be a clue to estimating how clearly or easily the requirements of the new system can be stated. If the requirements are well known to the program designers, the programming job is more straightforward and less costly. Somewhat related to this factor is the extent to which the programming organization will participate in the formulation of requirements; the less active its role in determining requirements, the more likely misunderstandings will develop, resulting in costly errors, omissions and ambiguities.

2. Program Design and Production

In a large-scale program system, program design involves the determination of the broad logical subdivisions of the computer system, the design of an executive program to control the sequencing of programs, the design of the data base structure, the allocation of computer storage, and specifications for utility and support programs. As in the first category, these factors center around the question, "How clearly understood are the program requirements?" The factors also are concerned with the size and complexity of the job; the resources and tools available; and the plans for documenting, verifying and testing the product.

Undoubtedly, the most important cost factor in this category is the number of computer program instructions and the types of programs that must be produced. In current techniques for estimating cost, the size of the program is often used as an intermediate measure to estimate the number of men who will be assigned and the number of computer hours required. Despite this reliance on size as the key to cost estimation, it appears that little research has been done to develop systematic and reliable ways to predict the number of instructions.*1

So far, the experience of many program managers is that the number of instructions is often grossly underestimated, except when very similar programs can be used for comparison.**

The conversion of the estimate of number of instructions to programming man months for large-scale systems is frequently done by allowing one man month for each 200 instructions. To test this rule of thumb, empirical data on the number of machine language instructions in eleven large command and control systems were compared with the 200-instruction-per-man-month guideline (Figure 1). The programs represent a variety of command and control systems using several different computers and languages. The number of man months includes program design, testing and coding. A further analysis of the data revealed that the production rate for operational programs averaged 225 instructions per man month while the rate for utility programs was 311 instructions per man month. The explanation for the higher rate of utility program production was that the

** For example, a report of the Controller's Institute stated, "... almost all EDP groups have at one time or another seriously underestimated the number of steps required. . . . Every company we visited added a substantial safety factor varying from 20% for a company which claimed, due to experience, a reasonable accuracy in its estimating procedures, to 400% for a company which had found itself that far out on a previous estimate." (Business Experience with Electronic Computers, New York Controller's Institute Research Foundation). "Measuring the Profitability of a Computer System." See also J. D. W. James.

* R. Bleier of SDC has reported on our attempt to relate total program length to the frequency of certain decision-class instructions, TM-1603.
program developer is his own customer for the program system and, therefore, can write his own requirements with little external coordination. For smaller programs not shown on the figure (i.e., less than 10,000 instructions), rates of as much as 400 to 1000 instructions per man month for individual programs were reported.

The conversion of the estimate of number of instructions to computer hours is also subject to various rules of thumb. Figure 2 is based on experience with eight large programs. The three points falling below the line represent efforts in which a procedure-oriented language was used, indicating that such use may reduce the amount of machine time required for program development. As shown in the figure, one computer hour is required for approximately every 53 instructions.

Another hypothesis resulting from the same data is illustrated in Figure 3. This shows a near-linear relationship between the two resources of manpower and computer time. It is likely that better information for predicting the number of computer hours required for the production of programs of various sizes and complexities is available in most programming organizations although it is often buried in accounting data.

A second important factor in this category is the extent of support program availability, reliability and documentation, including utility programs, debugging programs and library routines. Clearly, the more support programs that have to be produced "from scratch," the more manpower and computer time will be required for the total program development.

One other factor that may have a considerable influence on cost is the number and types of documentation produced for various types of programs. The graph in Figure 4 is based upon an analysis of five large-scale programming efforts. It suggests that there is a linear relationship between the number of pages of documentation actually produced to satisfy contract requirements and the number of instructions in the program. These data represent
documents delivered to customers. There may be many times this number of pages actually written in a large programming effort.

Some other guidelines for estimating the costs of documentation have been suggested. A drafting rate of 3 to 5 pages per day (750-1250 words) is a good rule of thumb for various types of programming documentation. A technical review rate of 20 pages per day seems average, but reviews that do not require extensive rewrites may run to 50 or 100 pages per day. Estimates on typing rates, illustrating rates and duplicating rates are usually fairly easy to obtain in most organizations.

RESOURCES THAT ARE AVAILABLE

3. Data Processing Equipment

The development of computer hardware is proceeding at such a rapid rate that it is difficult to make long-range estimates (beyond a few years) concerning its effect upon programming. Such improvements as faster add times, greatly increased memory capacities and speeds, multi-processors and new input/output devices may profoundly affect the accuracy of programming-cost estimation.

With respect to equipment, a critical factor is the number of hours per day that the computer is available to the programming staff. A commonly held intuitive notion is that the more hours per day the computer is available to programmers, the lower the over-all cost of the programming effort. Among the considerations in determining the number of hours per day of availability are the number of shifts per day, the time required for preventive maintenance and the number of other computer users sharing the equipment. Of course, another strong influence is computer capability; for example, large memory capacities seem to make programming easier. Further considerations are the power of the order code, the speed of access to primary and secondary memories, operating time and speed of input/output gear.

4. Programming Personnel

The most important factor in this category relates to the experience of the programmers assigned to the job. There are three particularly important types of experience:

(a) Experience with the particular computer—Clearly the more experience a man has with the particular machine for which the program system must be designed, the more apt he is to be familiar with its capabilities and therefore the less time it should take him to do the programming and testing.

(b) Experience with the particular language—Programming languages differ in their suitability for various types of programming efforts. Familiarity with an appropriate special language, such as NELIAC or JOVIAL, certainly makes the programmer more efficient and requires a smaller number of man months.

(c) Experience with the particular application—If the programmers have experience with the particular type of system application and/or design being programmed, then less time should be needed for the early analysis and definition phases of the job.

In addition to experience, cost is affected by the number of man months of programmer training required for the project. A possible hypothesis is that the cost of programmer training may take the form of a U-shaped curve. That is, for any given task there seems to be an optimum amount of training that the programmer should have. More training presumably would not produce a commensurate return, while less would lead to errors and confusion in the programming.

The number of people to assign to a given function or task also has an important impact on cost. There is probably an optimum number, for each type of programming effort, although research has not been conducted to determine what this is. In the field of research and development, which has some similarities with programming, the optimum number in a work group seems to be between 4 to 7.5

There are also costs associated with obtaining personnel—either in hiring or in transferring from other contracts and in relocating them, if necessary. This is a function of both
employee turnover and the size and type of the project. A study is currently being sponsored by the Navy at the University of Southern California to analyze the job of the computer programmer, develop criterion measures of performance and determine optimal personnel selection and classification procedures.9

NATURE OF THE WORKING ENVIRONMENT

5. Management Procedures

The design and institution of clear-cut management procedures may seem costly at the outset, but their true value must be determined by comparing the cost of the plan with the cost of not having the plan. Both these costs are extremely difficult to determine but experience in many programming efforts indicates that well planned projects enjoy higher productivity rates.

The most important effect upon total cost in this category is the use, maintenance, and monitoring of a management plan that includes at least communications and decision-making procedures, mechanisms for handling changes in the program, delineations of responsibility and authority, and schedules for major milestones and products. The plans should also specify standards for flow charts and documentation and quality control procedures. There is probably some optimum per cent of time that should be spent in planning for program development.

Another important cost factor is the number of computer runs permitted to each programmer each day. One of the reasons for the current activity in the development of time-sharing procedures is the belief that a larger number of computer runs per day for each programmer will shorten lead times and decrease costs. An equally reasonable hypothesis is that a cost trade-off exists between desk checking and computer testing such that some optimum number of computer runs will minimize total cost, the optimum depending upon the relative cost of computer time and programmer time.

6. Development Environment

A particularly important problem is the number of agencies with which the programming staff must coordinate. In addition to the user or customer, there may be separate agencies responsible for the contracting and for the other aspects of the system, such as hardware development. Problems of coordination and concurrency multiply as a function of the number of groups with which the programming staff must deal. Often, mutual education between these various staffs becomes necessary and the cost of briefings and meetings for such purposes may make the cost per instruction higher for large-scale program systems than for smaller ones.

7. Facilities, Services, and Supplies

In most accounting systems, normal overhead and miscellaneous supplies are covered by a percentage addition to the estimated direct labor and materials. However, unusual expenditures associated with large-scale programming efforts may not be adequately covered by the average overhead burden rate. For example, various types of technical and administrative support are needed to assist the programmer. Computer operators and EAM personnel save the programmer considerable time during testing. Effective and experienced management and administrative personnel assure that the work is efficiently organized and free of programmer to concentrate on technical matters. Technical editors help to ensure that documentation is adequate and understandable. There is some optimum mix of such support personnel that will ensure minimum cost.

Another important factor in this category is travel and communication cost. Trips may be required for briefings and conferences with user organizations and associated developmental agencies; for data gathering; for training and familiarization; and for concurrence on requirements or design. There is a strong tendency to underestimate these costs or to curtail them as an economic or political measure; this often results in delays in getting data or concurring on programming details, and these delays may be very costly in terms of time and manpower.

In some cases an important consideration is the cost of special facilities, e.g., simulation devices, special office equipment, or computer...
SUMMARY AND CONCLUSION

In this paper we have listed some cost factors that represent a consolidation and skimming of a larger list of factors that contribute to the cost of computer programming efforts. In many cases, the factors are very difficult to measure or quantify; further, their effect upon other factors and upon the total costs of the programming effort is often difficult to determine. For many of the factors, some data exist in current accounting records, but these data have not been collected, compiled and analyzed.

The mere listing of cost factors in programming is only a first step, albeit an important one, toward the development of a more scientific and, hopefully, a more precise method of estimating the cost of programming efforts. Ultimately, one would hope to discover by analysis of data some predictors that would enable a more accurate estimate of the number of instructions, man months and machine hours. Among the cost data—both estimated and real—that we would hope programming managers would begin to accumulate for the various types of products and activities in their projects are the following:

1. The number of machine instructions and the programming language used. Also, the percentage of the finished program consisting of library routines and subprograms from previous programs.
2. The number of man months of programmer effort, including the first level of supervision, and the experience level of the programmers.
3. The number of hours of machine time required for testing and debugging purposes and the types of machines used. Also, the pattern of machine usage in terms of runs per day and hours per run.
4. The number, types, and timing of important program changes and, in at least a qualitative sense, the effects of these changes on the final product.
5. The types and number of pages of documentation required, including a notation as to whether they are single or double spaced.

In addition, it would be useful if a log could be kept by a project “historian” describing certain qualitative attributes such as those described earlier in this paper. This section should describe the data-processing functions of the program system and its relationship to other systems. It should also identify all interim and end products, such as types, listings and descriptive documents.

A series of experiments to study those factors that can be analyzed only in controlled environments ideally should be conducted simultaneously with the collection of new and existing data on program costs. For example, well designed, statistical experiments would be useful for determining the effects of different types of programming languages upon total cost, the effects of greater or lesser machine availability upon costs, the optimum size of programming staff for different types of programs, and the best mix of programming talent (experienced versus inexperienced) for given types of jobs.

Obviously, it may be quite some time before a valid predictive set of equations can be developed for program cost estimation. Nevertheless, it is to be hoped that each small step in this direction will represent a useful experience in itself in terms of increased insight into the programming process. Certainly, research into techniques for improved programming management should be encouraged if the industry is to keep pace with the increasing demands for its services.

APPENDIX I LIST OF COMPUTER PROGRAMMING COST FACTORS

Summarized below for the convenience of the reader is the complete list of cost factors discussed in this paper.

Operational Requirements and Design

1. Extent of innovation in the system, its components, and especially the automatic data-processing function.
2. Extent to which the programming contractor will participate in a determination of the information processing needs (i.e., the system and operations analysis, and the system and operational design).

3. Number, size, frequency, and timing of system design changes.

4. Extent of command and control system decentralization and number of interfaces.

5. Number of other components and subsystems being developed concurrently as part of the command and control system (e.g., intelligence, sensor, etc.).

Program Design and Production

1. Number of computer program instructions and the types of programs that must be produced.

2. Number, types, and frequency of inputs and outputs to the computer(s).

3. Extent of innovation required in the program system; that is, the degree to which programs are similar in nature to those previously written.

4. Number, types, and quality of publications and documentation for both customer and internal use.

5. Extent of complexity of the data-processing functions.

6. Degree to which the following program design characteristics are recognized and must be incorporated.
   (a) Maintainability—the ease with which program errors can be detected and corrected.
   (b) Changeability—the ease with which new functions can be incorporated in the program.
   (c) Usability—the ease with which personnel other than designers can use the program.
   (d) Flexibility—the ease with which the program can be used for other purposes with only slight modification (e.g., SAGE programs for air traffic control).
   (e) Generality—the ease with which a program can accept a wide range of inputs.

7. Extent of the constraints on program design (e.g., real-time requirements).

8. Number, size, frequency, and timing of program design changes.

9. Extent to which data for data base is available, or data collection is required.

10. Number of entries (total size) for the data base, the number of different types of data needed for it, and the extent to which each can serve many programs or subprograms.

11. Efficiency of the programming language and the compiler or assembler.

12. Extent of the completeness and clarity of the system test and acceptance test requirements.

Data Processing Equipment

1. Number of hours per day of computer availability.

2. Extent of capability of the computer and its suitability for the job to be done.

3. Extent to which the operation of the computer and peripheral equipment is reliable, well tested, and well documented.

4. Number of equipment components being developed concurrently with the program.

5. Number of different computers for which programs are being prepared.

6. Number and types of displays used.

7. Extent to which adequate EAM support will be available.

8. Extent to which routine preventive and emergency maintenance will be available.

Programming Personnel

1. Types and quality of programmers.

2. Number of man months of programmer training required.

3. Number of programmers to be assigned to a given function or task.

4. Policy of obtaining and phasing of personnel to staff a new contract.

5. Rate of turnover.

Management Procedures

1. Extent of use, maintenance, and monitoring of effective management plans within
both the customer's and program developer's organizations.

2. Extent of formalized procedures to use the computer facility.

3. Extent to which there is a well defined and controlled system change procedure.

4. Extent of an error-reporting and -correcting procedure.

5. Extent of contingency plans in the event the computer is overloaded or otherwise unavailable.

6. Extent of quality control that is exercised during testing (e.g., reliability requirements).

Development Environment

1. Number of agencies the programmer contractor must deal with and their level of experience with system development.

2. Average number of days and effort required for concurrence.

3. Travel requirements.

4. Extent to which delivery dates for required programming tools are reliable, and correspondingly, the amount of pressure caused by a tight schedule.

5. Extent to which the computer is operated by another agency.

Facilities, Services, and Supplies

1. Number of computer operators and EAM personnel required.

2. Number and experience of technical management personnel, administrative personnel, and technical editors.

3. Cost of special simulation facilities, computer room facilities or special office equipment.

4. Number of square feet of new office space or building required.

5. Exceptional costs of graphic arts and reproduction.

6. Cost of punched cards, magnetic tape and other special supplies or equipment.

7. Cost of special security requirements (e.g., Top Secret vault).

BIBLIOGRAPHY


