SOFTWARE ESTIMATION TECHNOLOGY

Wilfred E. Lehder, Jr., D. Paul Smith, and Weider D. Yu

Wilfred E. Lehder, Jr., *is supervisor of the Project Management Technology Group at AT&T Bell Laboratories, Liberty Corner, New Jersey.* ***D. Paul Smith*** *is head of the Switching Systems Quality Department at Bell Laboratories at Indian Hill Park, Naperville, Illinois.* ***Weider D. Yu*** *is a member of technical staff in the Switching Systems Quality Department at Bell Laboratories at Indian Hill Park. Mr. Lehder is responsible for formulating and implementing measurement plans for software projects in the R&D community. He received B.S.E.E. and M.S.E.E. degrees from Tulane University. He joined AT&T in 1967. Mr. Smith's work has included design, development, and project management of digital switching systems and, most recently, quality management* (continued on page 18)

Setting unrealistically short schedules and low staff levels in the early stages of software projects frequently leads to lower product quality and reduced customer satisfaction. Effective cost- and schedule-estimating models of the software development process, however, can suggest appropriate staffing and reasonable schedules. Commercial models for this purpose have been available since the early 1980s, but differ widely in their algorithms and in the factors they use to adjust estimates to local environments. This paper reviews efforts in AT&T Research and Development to develop and use such models on AT&T projects. R&D models have been formulated and incorporated into software tools. The tools are being integrated into the planning process, and early results have been promising. When the tools were tested, the estimates were generally within 20 percent of actual experience.

Quality and Productivity

AT&T has established new, ambitious goals for quality and productivity on software projects. The goals reflect the increasing expectations of our customers for quality and the growing need to reduce project and software development costs in a competitive environment.

Solutions to the challenge of meeting the goals are multifaceted, as evidenced by the other articles in this issue of the *AT&T Technical Journal.* One promising approach is to refine the way we staff and schedule projects. Improper staffing and schedules, often short of real needs, may lead to delays for customers, compromised methodology, volatile priorities, inadequate testing, poor product quality, and low project productivity. Projects that are correctly staffed and scheduled generally satisfy commitments to customers and favor

increased productivity.

The software industry has proposed various models of the development process for estimating schedule and staffing needs. Barry Boehm of TRW Inc., an early contributor to the field, suggested in 1981 a model called COCOMO.[1] Other models include those of Jensen,[2] Putnam,[3] Rubens,[4] and Jones.[5] Many of these models have been incorporated into software tools that can be either purchased or leased through time-sharing services. The models are being refined continuously and, while still at an early stage of maturity, are becoming more accurate; this, in turn, has led to increased use. The federal government, for example, requires that bidders on new military projects use such tools to justify their proposed software effort.

Opportunity in AT&T R&D. Achieving a better estimation process is made easier in the AT&T R&D community because of similarities among projects. The community widely uses the UNIX® operating system, the C programming language, and the AT&T 3B processor line, for example. In addition, although a standard development methodology has not yet been set up, most managers have received similar training, such as that provided at the Software Project Management Workshop.

On the other hand, the estimation process to date has been carried out largely by local managers with only sporadic use of commercial tools. Only recently, through the work reported here, have we started to take full advantage of the wide range of experience of managers throughout the company, of expertise available through commercial products, and of recently completed projects that can provide valuable reference information. Using commercial references as a general guide and a database of historical information, we have developed two types of estimation models: one for use on a broad range of products ("standard projects") and one designed expressly for the 5ESS® switch, an extremely large project with special needs. The 5ESS switch model is distinguished by its use of well established hardware and software architecture; the size of a development effort is viewed largely in the context of its impact on the architecture. The model is further distinguished by its ability to separate effort on a feature-by-feature basis.

Estimation Scope in the R&D Environment

Software estimation is based on a model of software development in which a project life cycle is broken down into a series of events in the lifetime of a software product. An IEEE standard[6] defines the following life-cycle phases:

- Planning:
  - —Concept and definition
- Development:
  - —Requirement analysis
  - —Design
  - —Implementation
  - —Integration and test
- Maintenance:
  - —Installation
  - —Operations and maintenance
  - —Retirement.

Each phase is a semi-independent process that consists of a sequence of activities leading to completion of a major project milestone. The software estimating technology described in this paper pertains to the development subcycle only. The model for estimating development schedule and staff should resemble internally the development subcycle, which comprises the major separable segments of work.

Industry Models. Historically, models used in the software industry assume a "waterfall" life cycle described above— that is, one that passes through tiers of major activities—and consist of two parts. One part provides a base estimate as a function of high-level parameters such as project size and type (for example, military or commercial). Project size is generally measured in lines of source code. A second part modifies this base estimate to account for the influence of environmental factors such as the experience of the staff. As an illustration, Barry Boehm's COCOMO uses lines of code, representing size, raised to a power between 1.05 and 1.20, to derive a base effort estimate.[1] The specific exponent depends on whether the project is simple, average, or complex. The model identifies 15 environmental factors, covering computer, personnel, and project attributes, and treats them as multiplicative and independent in the effort estimation

Panel 1. Basic Equations

$$\text{Effort} = \frac{\text{size}}{\text{PR}} \prod_{I=1}^{n} 1/A_I$$

$$\text{Staff} = \frac{\text{size}}{\text{AS}} \prod_{I=1}^{n} 1/B_I$$

$$\text{Schedule} = \frac{\text{effort}}{\text{staff}}$$

where size = lines of code
PR = productivity (size of product normally generated by an individual per unit time)
AS = assignment scope (size of product normally assigned to one individual)
A_I = effort adjustment value for each model factor
B_I = staff adjustment value for each model factor

algorithm. It derives schedule and staffing from the effort estimate and other previously used project characteristics.

The Jensen[2] and the Putnam[3] models are more complex formulations than the COCOMO model. They are based on a Rayleigh curve view of effort during the life cycle. They include a so-called "diseconomy of scale," related to schedule, that is not necessarily reflected in other models. In Putnam's SLIM model, staff-months of effort for the same project increase as the schedule is compressed. Boehm, Jensen, and Putnam provide a single model of the aggregate life-cycle phases. They make phase estimates by dividing the overall effort and schedule estimates on the basis of experience.

The Jones[5] SPQR/20 approach is somewhat different. It models each life-cycle function separately—planning, design, implementation, test, user documentation/training, and project management. For each of these activities, staffing estimates are derived separately from effort estimates. The mathematical formulations, although not in the public domain, appear to be similar to the Boehm approach. Because it uses 12 separate algorithms, the Jones approach provides more flexibility when users prefer to fit the model precisely to the local environment.

SPQR/20 and the ESTIMACS model of Rubens[4] include an alternative to lines of code as a measure of software size. Because lines of code is frequently difficult to estimate early in a project, these models give users the option of choosing *function points* to identify size. Function points are an empirical measure of functionality that includes counts of inputs, outputs, inquiries, interfaces, and files. The concept was introduced in 1981 by Alan Albrecht of IBM.[7] Function points have been used effectively in the Information Management organization at AT&T, where most projects deal with management information systems and multiple languages are used. However, for applications with a large amount of real-time processing, Albrecht's function points are of limited value because program size generally is not related linearly to the number of inputs and outputs of the system and depends on other project conditions.

R&D Estimation Technology

AT&T R&D models are constructed like commercial models and are based on estimates of the size of the software product in lines of source code, adjusted by an appropriate set of project factors. Major factors affecting productivity and quality span the software development *project/process*, including personnel and support issues and the characteristics of the software *product*. Models in our estimation technology currently include between 10 and 20 factor inputs.

R&D estimation technology can be separated into five major components:

1. *Historical project database*, a collection of completed projects with known effort, staffing, and schedule results as well as environmental factor values.
2. *Estimation model*, a set of algorithms that produces the estimate values by modeling the software development process, the development environment, and the characteristics of the software product.
3. *Estimation process*, a set of procedures used by the estimation support staff and client project personnel to

generate the estimation results.

4. *Estimation tools*, user-friendly means of accessing the historical database and models and generating the estimates.
5. *Estimation results*, a report containing the estimate, including details such as development effort, cost, and schedule.

R&D Models. We developed our models by following these steps:

1. Review estimation needs with project personnel.
2. Establish a fundamental understanding of software products and the development environment.
3. Identify commercial models to be used as references in building models for AT&T R&D processes.
4. Identify the factors that influence software productivity and quality, using developer input and commercial references.
5. Establish data collection standards.
6. Collect historical project data and analyze results.
7. Choose a general model and establish factor influence levels based on industry experience, judgment of project personnel, and analysis of historical data.
8. Provide calibration algorithms for the estimation models.
9. Collect further data on completed projects to be used in refining models and as calibration references. Historical data can be used to adjust basic productivity relationships, to adjust factor weights, and to add new factors to the model.

The principal AT&T R&D models developed to date are similar to the model proposed by Jones.[5] Equations that describe the AT&T models are in Panel 1. Items—including environmental factors—that influence the base model used for *standard* projects are listed in Panel 2. The list of items evolved from consultations with project personnel, as described above.

Figure 1 is a graph of the influence of one of the environmental factors—staff experience—on the overall design effort. The effect of this factor almost doubles (from 0.8 to 1.47) as the value of the factor changes from 5 (very inexperienced) to 1 (very experienced). Therefore productivity and the total effort, according to the model, might be almost doubled if an initial very inexperienced work force were replaced or augmented by more experienced people.

Panel 2. Inputs to Model for Standard Projects

Size of product (lines of code)
Type of product (new or enhancement)
Estimate goals
Project class
Project novelty
Office facilities
Requirements stability and clarity
Design support and reuse
Documentation support and staff
Throughput and response time
Staff experience
Product complexity
User community

To accommodate the unique properties of the 5ESS switch and on the basis of input from the 5ESS switch development organization, the algorithms described in Panel 1 were modified to use experience with the 5ESS switch and to better suit the 5ESS switch process. The 5ESS switch model differs in over half its environmental factors from standard projects. In addition, it uses a modified version of the Albrecht function-point paradigm, with different inputs and a different computation process, to develop as output an estimate of the product size in lines of code. Panel 3 lists the input information needed for the 5ESS switch estimation model.

Creating Estimates for Standard Projects

In the estimation process for AT&T standard projects, unlike that for 5ESS switch projects, many different managers must be served and many more estimates must be generated. This means that standard projects need either more support personnel or tools by which clients can formulate estimates themselves with only a modest amount of assistance from the estimation support staff.

Estimation usually begins at or near the beginning of a product's life cycle. The usual sequence of events for development of a cost and schedule estimate is as follows:

- Project personnel contact the estimation support organization and arrange the necessary support agreements.

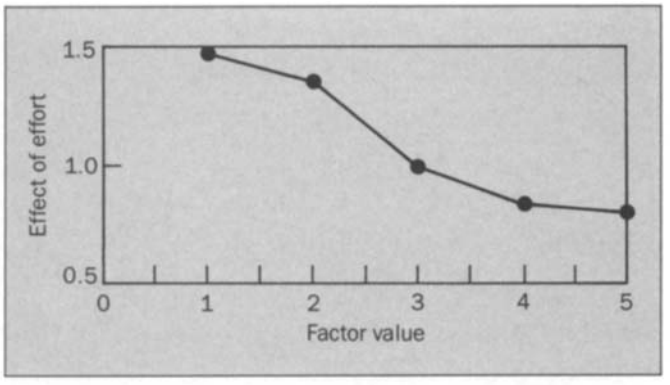

Figure 1. Effect of staff experience on design effort.

Panel 3. Inputs to Model for 5ESS Switch

Feature requirement
Feature hardware
Feature novelty
Feature development process
Feature program complexity
Feature performance constraints
Work environment
Staff experience
Software architecture
Feature interaction
Static and dynamic data

- If local history information is not already in the database, historical data on one or more recently completed projects are collected with a "postproject" questionnaire.
- A team of experienced project personnel answers early "preproject" questions. A member of the estimation support organization may participate in the preproject meetings to clarify question intent and help with answers.
- Support or project personnel generate cost estimates (under alternative planning assumptions if necessary), using tools and assistance from the estimation support organization.
- They select a "best" estimate and planning scenario, modify it as necessary to account for factors outside the scope of the model, and on the basis of the estimate, commit the project to corporate management.

The final estimate is archived in a project database. Further estimates may be run as updated information—on project size, for example—becomes available.

Staff hours of effort are recorded during the project to generate accurate data for the historical database. Information on lines of code developed for the product and updated environmental factors are collected with the postproject questionnaire and added to the project history file. (Environmental factor information in both the preproject and postproject phases has been expanded to include most factors that experienced software engineers deem to affect software productivity. This information provides the basis for additional project comparison and allows eventual refinement of the model. The set of about 100 questions covers information in the categories listed in Panel 4.)

Creating Estimates for 5ESS Switch Applications

The 5ESS switch is a modern digital electronic switching system with a distributed hardware and software architecture. The size of the latest 5ESS switch generic software has reached millions of lines of source code. At present, hundreds of software engineers are involved in the development process for a single release.

For 5ESS switch projects, estimates are done separately for each feature of a software release. The generic estimation process for the 5ESS switch is illustrated in Figure 2. Estimates for the feature development effort are in average technical staff years. Estimates for the feature size are ultimately transformed into source lines of code.

When certain key documents on a new 5ESS switch release are completed— documents containing most of the input values for the estimation model—the preproject questions are distributed to the design engineers and an estimation meeting is scheduled. Before the estimation meeting, the estimation support staff contacts key development engineers to refine the input to

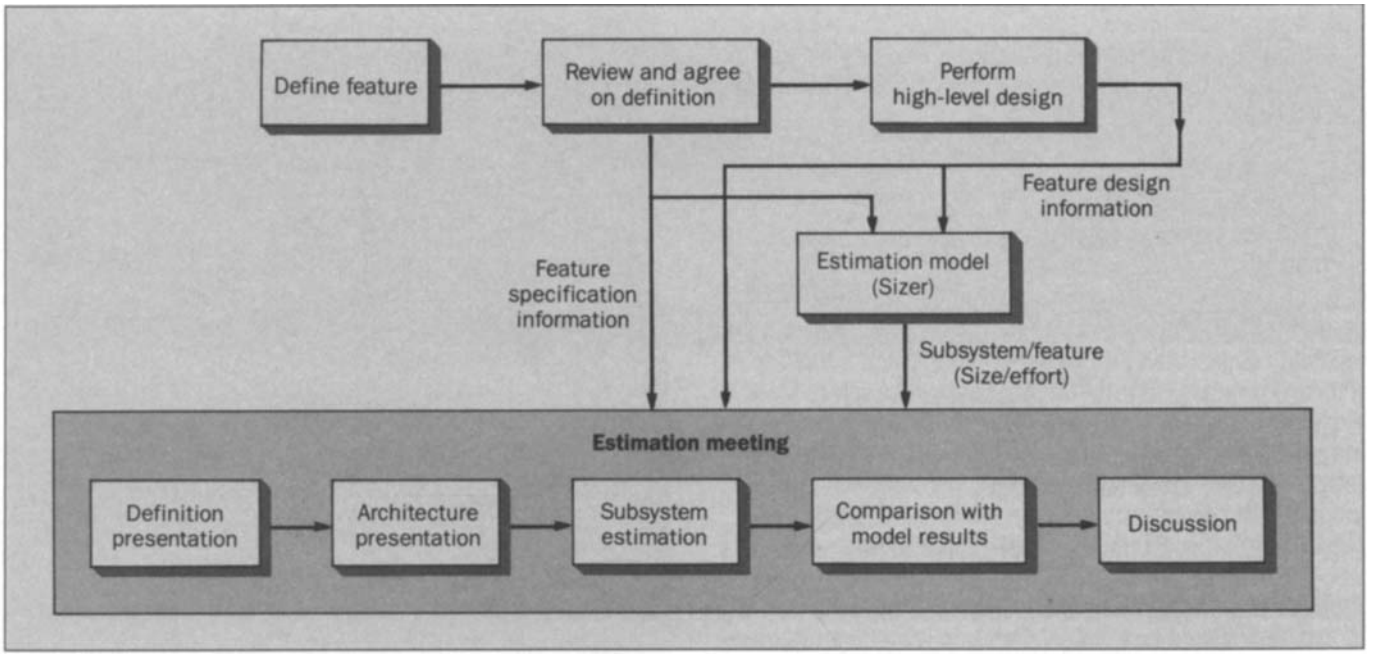

Figure 2. Estimation process for 5ESS switch.

the model. The estimation support staff develops the model estimates, and they are brought to the estimation meeting.

In the meeting, the system engineer and architectural engineer review information from the feature specification and design proposal documents. These participants have the expertise to resolve easily most of the open issues related to the feature. The meeting yields the information needed to derive estimates.

If misunderstandings are found about the impact of a proposed new feature, the estimation staff revises the model input and regenerates the model estimates. The staff then compares the model estimates for the feature with estimates from development engineers. If there are any large discrepancies between the model estimates and engineer-generated estimates, a reasonable explanation of the differences between the two is sought.

Final estimates are added to the project database. At the completion of the project, final results, such as lines of code developed, are added to the project database.

Comparison of the Two Processes

The estimation processes for standard projects and for the 5ESS switch differ in a number of respects. The 5ESS switch estimation staff is concerned with understanding a new feature and directly assisting in applying the model input values to the feature. On standard projects, this degree of insight is not attempted, and estimates are developed on a project basis rather than on a feature basis.

In addition, there is difference in the timing of the interaction between the estimation staff and project personnel. On standard projects, the interaction is earlier than on 5ESS switch features and generally before the requirements documents are available. Moreover, the standard project model requires as input a view of eventual product size in lines of code, and it provides outputs for effort, staff, and schedule. The 5ESS switch models, in contrast,

provide as output lines-of-code size estimates based on a function point equivalent. The 5ESS switch models also provide staff effort estimates but no output for staff level and schedule.

Support Tools

Just as processes differ between 5ESS switch projects and standard projects, so do the tools used for estimation.

Tools for Standard Projects. One approach to minimizing estimation effort is to build as much estimation capability as possible into a support tool. Such a tool for standard projects, called SUMMS (Support System for Modeling and Measures), provides on-line support of cost and schedule estimation. It furnishes on-line screens for entry of project size estimates and environmental factors, and it stores this information in a relational database. SUMMS provides flexibility in analyzing results.

As part of its estimation algorithms, SUMMS offers several calibration modes defining how historical information should be used. Calibration can currently be based on:

- A commercial reference
- An average of completed projects in the database
- Projects from the database selected according to criteria such as software size or organizational identity, or a combination of such criteria.

SUMMS collects information at project completion so that its reference database grows with new R&D experiences. Finally, it supports collection of answers to factor questions not included in the initial model (see Panel 4); this allows future expansion of the model. Figure 3 is a diagram of the SUMMS main menu.

Tools for the 5ESS Switch Project. A tool called the 5ESS Switch Sizer helps users develop size estimates, in lines of code, and make planning estimates. It includes a 5ESS switch project database and refers to the data when making new estimates. It automatically records each estimation session in an "estimation worksheet" file for file retrieval and cross-referencing.

The Sizer was developed in the UNIX system environment and written in Lisp language. It is available for 5ESS switch project managers and feature estimators.

Panel 4. Categories for Full Factor Set

General:
- Administrative
- Data collection effort
- Project classifications

Product:
- Architecture
- Complexity
- Requirements
- Characterization
- Code size/languages
- Technical constraints

Project/Process:
- Tools
- Methods
- Organization
- Schedule
- Staff resources
- Support environment
- Effort

The tool has an interactive user-friendly interface and refers to a knowledge base for help in generating its output.

The expertise gained from developing and applying the Sizer has been used in developing the 5ESS Switch Development Estimator, a tool that gives detailed estimates of effort and schedule for each development stage. It can use the inputs to the Sizer but requires more detailed inputs.

Results of Software Estimation Technology

The new estimation technology has recently been introduced, and its effect is beginning to be felt. Feedback from users is summarized below.

More Thorough Planning. Users of the estimation model have found that the broad scope of the input questions allows them to focus on and refine major planning assumptions or decisions leading to the possibility of higher productivity and quality. Assumptions and decisions

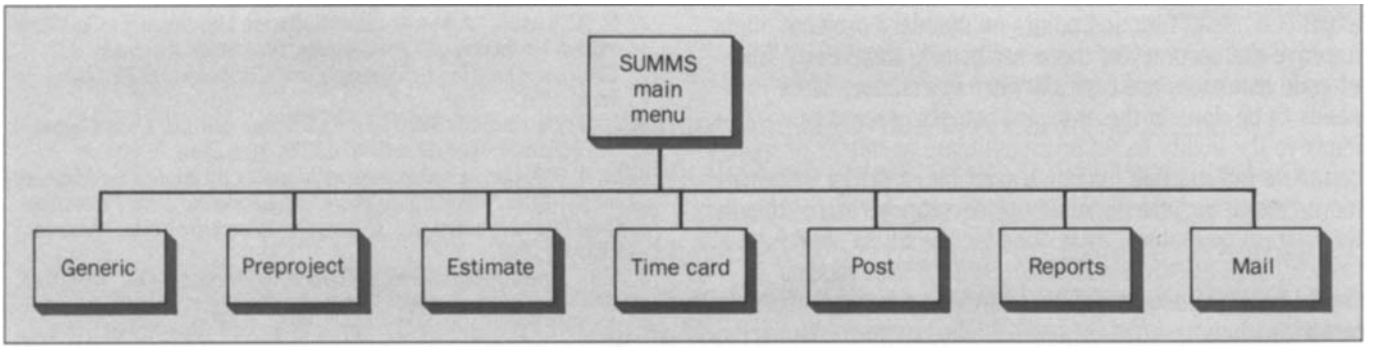

Figure 3. Structure of SUMMS main menu.

for reconsideration include those related to the attributes of planned new features and to all environmental conditions, such as experience of the work force and quality of the support environment.

Establishing Data Collection Standards. In the past, collecting project data and analyzing historical results have been complicated by the lack of guidelines and standards. Now, however, to accommodate our estimation needs and our new data collection tools, standards and methods have been defined. Standards have been developed for counting lines of code and for counting development effort, for example. Because of these new standards, the quality of historical project data has greatly improved.

Using Historical Records. Project managers have found historical information valuable. They use historical data to calibrate general productivity.

Productivity Studies. With a set of well-defined software productivity metrics, the estimation support staff can compare projects, study the influence of cost factors on productivity, and update the model as appropriate.

Refining the Models. In discussions with experienced R&D project managers, it is clear that important factors were omitted from the initial models—factors related to distributed architecture, concurrent development of multiple software releases, and organizational structure, for example. These additional factors have been included in the environmental factor question set described in Panel 4, and future models will include those additional factors that prove to be useful.

Model Estimation Accuracy. Measures of success include:

1. Comparison of model-derived estimates with independently derived estimates from the development engineers
2. Comparison of final estimates with actual project results.

For measure 1, model estimates have been generally within 20 percent of developer-generated estimates.

For measure 2, estimates were within 5 to 20 percent of actual results for the 5ESS switch. SUMMS estimates for projects agreed with actual results within about 20 percent. The SUMMS comparison was based, however, on its commercial reference because it was made before a sufficient local database was available.

User Expense. Because the input questions of the estimation model are generated from locally defined factors or are customized to the language and context of R&D efforts, users can understand the questions easily and respond promptly.

For standard projects, trials have shown that it takes less than a staff day of effort to establish the initial inputs for generating estimates.

Future Directions

Considerable progress has been made in selecting cost and schedule estimation models and in incorporating methods to use the models in the R&D community. A number of large and small projects have successfully used the models.

Much remains to be done. Alternatives such as an

expanded use of function points on standard projects might improve the accuracy of these estimates, since early lines-of-code estimates are typically very inaccurate. More needs to be done in the standard project process to improve the ability to refine an estimate as better or more complete information becomes available. Further improvements might include the ability to decompose and estimate the system by feature, as is done for the 5ESS switch.

The SUMMS and 5ESS Switch Sizer models should be expanded to include factors or to replace current factors with more efficient ones. Links between estimation systems and other mechanized systems should be established to simplify obtaining data such as the lines-of-code counts for completed projects.

The models must be adapted to changes in technology, many of which are reported in this issue of the *AT&T Technical Journal.* Changes in software development, such as reusable software modules and greater use of the C++ language, alter many model assumptions and the way in which historical results can be applied. Adapting to these changes will require new project reference data and modified models.

A longer-term outgrowth of the estimation effort may be that similarities between projects that today are organizationally different could become apparent. This will help to create a greater sense of community among organizations and promote further sharing, thereby stimulating further improvements in productivity and quality.

References

1. B. W. Boehm, *Software Engineering Economics,* Prentice-Hall, Englewood Cliffs, New Jersey, 1981.
2. R. W. Jensen, "A Macro-Level Software Development Cost Estimation Methodology," *Proceedings, Fourteenth Asilomar Conference on Circuits, Systems, and Computers,* IEEE, New York, 1981.
3. L. H. Putnam, *Software Cost Estimating and Life Cycle Control: Getting the Software Numbers,* IEEE, New York, 1980.
4. H. A. Rubens, "Productivity and Quality Strategies for Measurement," *Fifth National Conference on Measuring Data Processing Quality and Productivity,* Quality Assurance Institute, Orlando, Florida, 1987.
5. T. C. Jones, *Programming Productivity,* McGraw-Hill, New York, 1986.
6. ANSI/IEEE Standard 729-1983, "Standard Glossary of Software Engineering Technology."
7. A. J. Albrecht, "Function Points as a Measure of Productivity," *Proceedings, 53rd Meeting of GUIDE International,* Guidance for Users of Integrated Data Processing Equipment Conference, Dallas, 1981.

Biographies (continued)

of the software development process for the 5ESS® switch. He received a B.S.E.E. from Brigham Young University and an M.S.E.E. and Ph.D. from Stanford University. He joined AT&T in 1967. Mr. Yu works on software productivity study, software estimation, cost modeling, and software requirements traceability in the 5ESS switch project. He received a B.S. in mathematics from Fu-Jen Catholic University, Taiwan, Republic of China, an M.S. in computer science from the State University of New York at Albany, and a Ph.D. in computer science from Northwestern University. He joined AT&T in 1983.

(Manuscript received April 15, 1988)