SQA—A Proactive Approach to Assuring Software Quality

M. Hosein Fallah
Ahmad M. Jrad

As AT&T participates in increasingly competitive global markets, the need to provide consistently high-quality software products becomes more crucial. Use of the Best Current Practices (BCP) guide is a key element of the AT&T Bell Laboratories approach to world-class quality. Software quality assurance (SQA) is a function that helps product-development organizations proactively measure, utilize, and improve development processes in order to produce high-quality software. The goals of SQA are to:
- Facilitate the establishment of process standards,
- Achieve the desired level of product quality,
- Assess compliance with established standards and processes,
- Identify instances of noncompliance for corrective action, and
- Take corrective action as required.

This paper presents the essential elements of an SQA function, and summarizes industry and AT&T experiences with SQA. The paper also discusses approaches for customizing SQA to differently sized projects and various phases of the software-development process.

The Need for SQA

SQA is used by many industry leaders for competitive advantage, as a requirement for ISO-9000[1] certification, and for advancing from a Software Engineering Institute (SEI) Capability-Maturity-Model[2] Level 1. The purpose of SQA is to assure the quality of software and its development processes. SQA used to mean inspecting for quality only when new software was completed. Now, SQA encompasses the entire production process—measuring and verifying quality at each step of product development.

SQA recognizes the need for independent verification of the software-development process. If developers understand the steps being followed, they can analyze them, identify their weaknesses, and improve on them. Such activities require continuous attention by the project team and an unwavering commitment by management. SQA helps to integrate these important steps into the production and software-development processes.

ISO-9000 standards for software development, and the SEI capability maturity model, are encouraging many software-development organizations and suppliers to place increasing importance on quality-assurance functions. Today, customers are as concerned about how software is developed as they are about the products themselves. For example, the European community is taking steps to ensure that all of its suppliers are ISO-9000 certified.

AT&T software-development organizations are beginning to acknowledge the need for SQA. This need has been reinforced by the software-process assessments conducted by QUEST over the past several years. These assessments have shown the need for a formal SQA function for many projects, raising awareness about SQA's importance. In addition, many AT&T software-development organizations have been getting ISO-9000 certification, which requires SQA, to maintain international competitiveness.

Panel 1. Abbreviations, Acronyms, and Terms

BCP	Best Current Practices
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
MR	modification request
QUEST	Quality, Engineering, Software, and Technologies
SEI	Software Engineering Institute
SQA	software quality assurance
SQAP	software quality-assurance plan

SQA Best Current Practices

In support of the AT&T software-development community, QUEST developed a BCP. The BCP describes SQA deployment, AT&T experience with SQA, and the extensive benchmark studies completed by experienced researchers. The BCP determined that the key elements of an SQA practice include:

- *Standards and guidelines for the development process*, including criteria for evaluating levels of product and process quality at all life-cycle phases;
- *Internal audits*, designed to evaluate periodically the compliance of projects to their stated processes;
- *Corrective action*, to ensure that noncompliance is addressed and corrected in a manner consistent with the standards;
- *Organization*, including SQA personnel, their roles and responsibilities, reporting structure, and project interfaces;
- *Metrics*, designed to measure effective compliance with the quality criteria for each process phase;
- *Tracking*, the procedures for monitoring SQA activities;
- *Project-specific SQA plan (SQAP)*, which defines all SQA activities; and
- *Exception reporting*, the procedures for handling deviations.

The SQA Process

Figure 1 illustrates a typical software-development model that consists of three phases: requirements, design, and testing. Each phase takes the preceding input and transforms it into output that can be used by the next phase. The SQA model "shadows" the development model with the evaluation activities of each phase. For instance, the design-evaluation activity appraises the conformance of the design phase against certain standards and guidelines, producing an overall score that ranges from zero percent (no conformance) to 100 percent (total conformance).

This score is compared with one suggested in the quality criteria within the SQAP to determine if the design phase should be earmarked for corrective action. Figure 1 shows that the requirements phase achieves 100 percent conformance, the design phase 60 percent, and the testing phase 85 percent. It also shows that the testing phase needs corrective action, because the testing does not meet the established quality criteria.

In Figure 1, the boxes at the bottom of the illustration depict software-development activities. The diamond-shaped outlines represent the corresponding SQA activities. Note that the SQA activities have feedback paths. This allows for noncompliance reporting to the software developers, who can then attempt to resolve any problems.

The SQA process consists of two essential functions: *assessment*, and *control and improvement*. SQA is not complete without both.

Assessment is the function that enables the SQA staff to measure the quality of a product at any given point in its life cycle. This function is shown by the solid lines in Figure 2.

In the control-and-improvement function, the SQA staff has the responsibility of providing regular feedback about quality (both positive and negative) to the developers and product management, thereby closing the quality loop. This function is shown by the dotted lines in Figure 2. Thus, SQA ensures that proper corrective action is taken whenever needed to rectify nonconformance with established processes.

SQA Organization and Goals

In planning for SQA, a product-development organization must establish a number of specific goals. These goals help define the SQA function, and they serve as criteria for evaluating SQA effectiveness.

Organizational goals can have a specific hierarchy, and high-level goals often have lower-level objectives associated with them. High-level goals can include organizational and product oriented goals. Typical high-

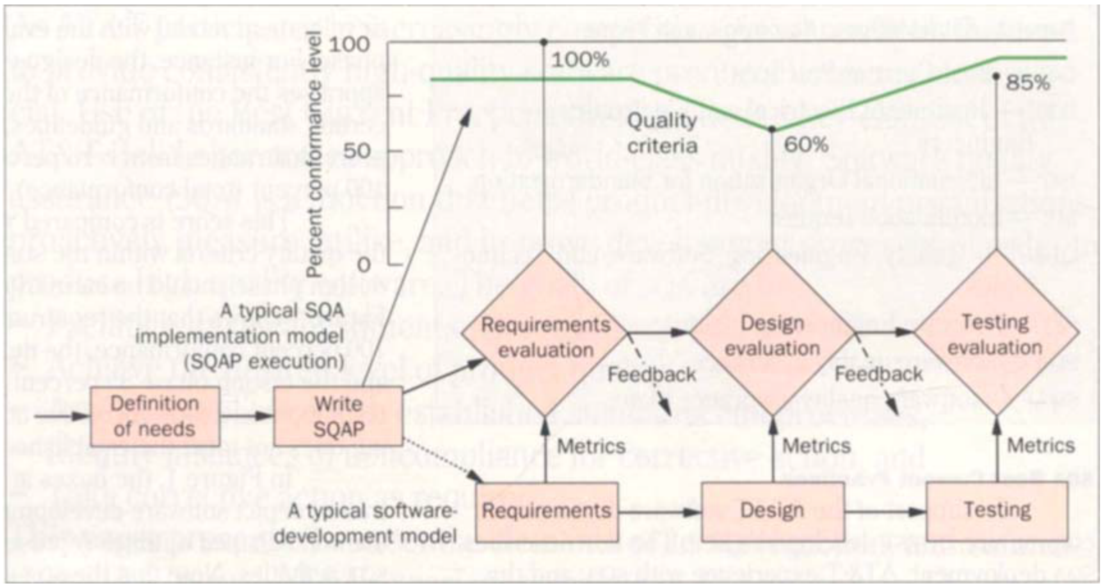

Figure 1. This illustration of the SQA process depicts a typical software-development model consisting of three phases: requirements, design, and testing. The SQA model "shadows" the development model with the evaluation activities of each phase. The boxes at the bottom depict software-development activities, and the diamond-shaped outlines represent the corresponding SQA activities.

level organizational goals include:

- Achieving ISO-9000 software registration,
- Establishing a more predictable development process,
- Moving to SEI Level 2 and higher, and
- Reducing development costs.

Typical lower level organizational objectives include:

- Determining the absolute or relative number of deficiencies per audit;
- Recording the number of modification requests (MRs) detected in a system test;
- Ascertaining the number of faults per lines of code, detected after software release to the field; and
- Reporting the development cost per line of code or function point.

Among the most important product oriented goals is *achieving greater overall product quality.* This goal encompasses factors affecting reliability, efficiency, usability, and maintainability.

SQA Tasks and Responsibilities. Figure 1 infers that SQA personnel review, audit, inspect, monitor, and track the activities and progress of software development for conformance with established procedures. The SQA function is normally integrated with all phases of the software-development cycle.

In addition, SQA can accommodate exceptions—in certain instances—when a project must deviate from established procedures. SQA tracks any instances of identified nonconformance, ensuring an appropriate resolution.

Other tasks and responsibilities are defined in the SQAP, which is supported by the SQA charter. These tasks and responsibilities can vary, depending on the specific needs of the organization, and can include:

- Selecting an appropriate process methodology for the project; (Benchmark studies indicate that, in most instances, management selects the development methodology and the SQA group approves it before a project starts.)
- Reviewing the initial contract proposal, and participating in the bidding process;
- Reviewing development plans for completeness and adherence to processes;
- Participating in selected reviews and walks-through;
- Inspecting test results to ensure conformance with standards;

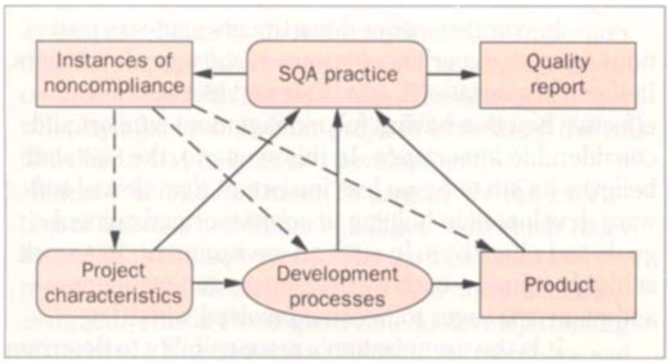

Figure 2. The SQA process consists of two essential functions: *assessment*, and *control and improvement*. The assessment function, shown by the solid lines, enables the SQA staff to measure product quality at any given life-cycle point. The control-and-improvement function, shown by the dotted lines, allows the SQA staff to provide regular feedback about quality to the developers and product management, thus closing the quality loop.

- Performing periodic internal audits on projects, and issuing reports of findings;
- Tracking corrective action on instances of process nonconformance;
- Tracking corrective action on problems reported by customers;
- Collecting and analyzing data that is based on metrics defined by the organization; and
- Assessing the effectiveness of the existing training program for new hires, within both the SQA and software-development groups.

SQA Resources. The SQA function must be fully and visibly supported by upper management. This means that all SQA groups must be given the proper tools and resources necessary to assure the quality of software-development projects. It is important that such support be clearly articulated in a formal policy statement.

In order to minimize initial start-up costs, some organizations may consider initiating SQA trials in areas of the development process that will show immediate improvement. This is best done during the later stages of the development life cycle—for example, in the implementation and testing phase. When positive results are achieved, SQA can be extended to include earlier life-cycle stages. This approach can help overcome any management skepticism by providing tangible results soon after SQA-function implementation. Another benefit of this approach is gaining early credibility for the SQA group.

SQA Reporting Structure. The SQA organizational structure should be such that functional effectiveness is preserved. There is always a potential for conflict of interest between SQA and development roles. Essentially, developers should not be performing their own SQA or monitoring themselves for compliance with standards in any official capacity. It is true that most developers monitor themselves informally to ensure that they stay on schedule and meet their customers' needs. However, studies show that developers and project managers may feel pressured to use short cuts, thus compromising process and project quality. In some cases, there may be a temptation to abandon SQA work and divert all resources to software development in order to meet a production schedule. It is important, therefore, to have an impartial observer, one who can assess the situation objectively, performing the SQA function.

This procedure is necessary for organizations seeking to climb the SEI ladder. SEI suggests that SQA personnel can be effective only when reporting through an independent management chain.[3] The SQA staff should not report to a project manager, and should not have any more than one management position between it and the senior location manager. In addition, the SQA staff should have a direct-report relationship (a dotted line within the organizational structure) with the senior corporate executive.

ISO-9000 guidelines[1] and the Institute of Electrical and Electronics Engineers (IEEE)[4] do not require a specific reporting structure. They do require that the organizational and technical interfaces be identified and documented.

Establishing an SQA Function

Initially, it is important to plan for an SQA charter, determine responsibilities and duties, agree on a reporting structure, and ascertain resources. An effective SQA function cannot exist without thorough planning. Many development organizations may think they already have a complete SQA function in place. But, in reality, they have only bits and pieces scattered about. Experience shows that if a total SQA function was never formally established, then it probably does not exist. A disjointed collection of SQA segments does not necessarily comprise a comprehensive SQA function. There must be a planned and coordinated program, in place, before gaining the benefits SQA can provide.

Management Commitment. Total management commitment is a prerequisite to establishing an effective SQA function. Without this crucial ingredient, SQA would lose much of its effectiveness. A survey by the Quality Assurance Institute found that one-third of all SQA groups fail within two years of formation. This failure rate indicates several underlying problems that impede the success of the SQA process. One is a lack of genuine commitment by upper management. Another is the absence of the necessary resources to ensure success.

Management support will grow, both from experience and from early progress. Nevertheless, success stories from similar organizations that have implemented SQA functions should be used, initially, to encourage management acceptance and support.[5]

Management should focus primarily on providing an environment conducive to cooperation and teamwork. By doing this, a good working relationship will be established between the software-development and SQA groups.

SQA Charter. As previously discussed, a strong beginning requires the establishment of a clear and unambiguous charter, which must be linked directly to business goals. The charter must be defined, documented, and published within the organization. The SQA charter provides:

- Information to managers regarding SQA goals and objectives,
- The means to align the SQA function with the organization's long-term objectives,
- An overall high-level map of the SQA function, and
- Mechanisms to ensure long-term management support for the SQA function.

SQA Personnel. When staffing for the SQA function, many departments have differing philosophies and practices. Benchmark studies show that some organizations hire recent college graduates who, after gaining experience with SQA, move on to software-development work. The studies indicate that this practice results in an ineffective SQA function. In addition, the entire staff often becomes frustrated due to the apparent lack of management commitment and the perception that SQA is not valued work.

In contrast, other departments staff SQA positions from an experienced group of software developers. In these organizations, SQA is viewed as a useful and effective function having full management support and considerable importance. In this scenario, the SQA staff believes its job to be no less important than that of software developers in helping to achieve organizational goals and objectives. In such an environment, SQA work is highly valued—both by developers and managers—and many are eager to become involved with it.

It is the organization's responsibility to determine its SQA charter, function, and direction. If an SQA group is established as one in which important work is done, then the best people will be attracted to the group and will want to work in it. Due to their experience and abilities, such individuals help make an SQA group successful, as well as inviting to other highly qualified specialists.

Conversely, if an SQA group is established merely as a token organization, without any real responsibility or authority, most people will probably disassociate themselves from it. As a result, those who do staff the group may not have the proper qualifications, and might not command the respect and esteem of their colleagues.

Ten Steps To Establishing an SQA Function. The following steps are suggested for establishing a new SQA function:

1. *Obtain Management Commitment.* Management must be committed to SQA, must understand its purpose, and should have realistic expectations of its goals. An SQA manager must be appointed, and SQA team members must be designated for initial planning work.
2. *Develop the SQA Charter.* The SQA team, consisting of manager and staff, refines and further defines SQA goals, and then obtains management concurrence. Next, the team documents the SQA charter and tasks, and designates individuals responsible for completing the tasks.
3. *Obtain Software Development's Concurrence.* The SQA team seeks the development organization's concurrence with the SQA charter, and requests its commitment of time and effort in assisting with later SQA activities. Every project should have a team of developers that will dedicate a portion of its time to SQA definition work.
4. *Write the SQA Plan.* The SQAP is developed and co-authored by the SQA team and the software-development organization. The SQAP should describe the formal development process for a product, as well

as the use of standards and templates. It is important that developers take ownership of the SQAP through co-authorship with the SQA team. The first SQA project will play a key role in later projects in defining the relationship of SQA to software development.

5. *Activate the SQA Function.* To prepare for the execution of the SQAP, additional qualified individuals must be added to the SQA group. The various tasks and responsibilities described in the SQAP must also be assigned. This is also a good time to develop training materials or tutorials for new SQA team members and software developers.
6. *Train Team Members and Developers.* The SQA team must be trained to carry out the SQAP. In addition, developers should be introduced to the SQA function and trained in its processes. A presentation introduces the development organization to SQA. This provides a good opportunity for the SQA team to discuss questions and concerns with developers.
7. *Complete the SQA Work.* SQA implementation should be done gradually. At first, it should be applied to just one or two projects before being disseminated throughout the development organization. A schedule for completing each SQA activity should be established, and a deviation-resolution system should be implemented.
8. *Assess the Development Process.* When implementation of the SQAP has begun, the effectivity of the development process—including areas where it is the least effective—should become clear. Root-cause problem analysis is a good method of evaluating the development process as a first step to improvement.
9. *Assess the SQA Function.* An independent third party should be commissioned to assess the performance of the entire SQA function. This individual or group should be asked to identify weaknesses, make recommendations, and suggest corrective action.
10. *Share Successes with Others.* While SQA is being established and implemented, success stories should be shared with other projects teams and development groups. This sharing should lead to the acceptance and eventual dissemination of SQA throughout the entire organization.

SQA Planning

The first and most important SQA staff activity is planning for quality. Obviously, the achievement of quality does not happen by accident. In fact, sometimes it does not even happen when planned. At the minimum, early and comprehensive planning is required to help ensure a good understanding of quality goals and objectives.

SQAP Structure. SQA planning, accomplished in Step 4 in the previous subsection, is completed after forming an SQA group and developing a charter. The SQAP serves as a contract between the SQA group and the development organization. As such, the SQAP identifies the roles and responsibilities of both the software-development and SQA groups. The essential elements of the SQAP include:

- Goals and objectives;
- Reference documents;
- Management of the SQA function;
- Identification of software-maintenance documentation;
- Monitoring compliance with standards, practices, and conventions;
- Reviews and audits;
- Software-configuration management;
- Problem reporting and corrective action;
- Tools, techniques, and methodologies;
- Code control;
- Supplier control; and
- Record collection, maintenance, and retention.

In addition to SQA goals and objectives, the SQAP should have two other major sections. The first additional section is a *description of the development process*, which outlines software developers' respective roles at various stages in the product-development cycle. Developers are assured of following these guidelines when conforming with the SQA function. Thus, it is important to obtain the development organization's concurrence with the SQAP as early as possible. This section also contains a configuration-management guide, as well as information about standards, conventions, practices, reviews, and audits.

The second additional section of the SQAP is an *overview of management processes and the SQA role.* It contains guidelines for data collection, measurement, metrics, problem reporting, corrective action, resolution handling, and a supplier-management model.

Development Standards and Templates. When several individuals are working on different portions of a new product, standards are needed to ensure that all the pieces will fit together during product integration and testing.[6]

In a monitoring operation, standards and guidelines determine a project's characteristics, documentation requirements, and quality-assurance criteria. Lack of standards would adversely affect the productivity and maintainability of the development process, as well as the reliability of the finished software.

Standards and guidelines can vary widely from one organization to another, and there is no obvious reason why those of one group would always apply to another. It is each organization's responsibility to develop its own customized standards and guidelines before implementing the SQA process. This does not preclude the possibility of reusing some existing industry standards and building on them. New standards can be developed from various sources, including:

- An organization's experience and historical data,
- Existing BCPs,
- Existing best area practices, and
- Industry standards.

Experience with SQA

To date, AT&T has had limited experience with formal SQA functions, although its development organizations have long been involved in quality initiatives and process-improvement programs. The experience of the rest of the software-development industry varies. Some companies have performed extensive SQA, and some firms have had no experience at all with the SQA function. The SQA BCP that was developed by QUEST relied on internal and external benchmark studies to obtain real-world-experience data from SQA practitioners.

SQA Benchmark Experience

AT&T researchers interviewed individuals representing six different SQA organizations. These included two internal (within AT&T research-and-development groups) and four external organizations. This benchmark study was instrumental in demonstrating the various ways in which different groups adapted SQA to fit individual group needs.

The study determined that there is no single way of best performing the SQA function. In fact, six different ways of implementing SQA emerged from the six organizations interviewed. Several different, customized, SQA implementation plans also were discussed.

The motivation behind the development of disparate SQA strategies results from two major needs:

- Meeting some overriding business objective, such as the requirement for ISO-9001 registration; and
- Reducing expenditures by lowering SQA's cost.

There are widely varying approaches to SQA implementation among organizations that are driven by the need to remain—or become—cost effective.

It is important to note that minimizing the resources allocated for SQA—with the goal of reducing costs—may well have the opposite effect. If SQA cannot serve its intended purpose due to insufficient resources, efforts at error prevention may eventually slacken, causing development and testing costs to rise. Hence, it is important to maintain SQA at an appropriately high level.

Data from the benchmark studies suggested that SQA resources should be maintained on the order of five percent to seven percent of the total development cost. This implies that for every 100 developers, five to seven individuals dedicated to SQA are needed to support such a function effectively.

Current AT&T Experience

Many AT&T organizations have already begun to deploy SQA functions. In one organization, the SQA group was heavily focused on audits. It had tailored the function entirely toward ISO-9001 registration. The SQA staff audited their processes much as they would an International Standards Organization (ISO) registrar. Therefore, the staff used the SQA function to conform with the ISO requirement on internal auditing. They satisfied some of the key ISO-standard elements directly—through the SQA function. For instance, SQA was used to maintain quality records and to track corrective actions. By using this approach, the organization was successful in obtaining ISO registration.

In another AT&T group, SQA was more of an ongoing activity. In this case, many other activities were performed that did not fall under the SQA "umbrella." For instance, SQA staff members acted as meeting facilitators for two development teams. Often, the staff assumed a hands-on approach, looking for potential problems by any means possible, including the use of electronic tools to access development data. The goal was to uncover problems quickly and to report them—through proper channels—for corrective action. The SQA group was also engaged in investigating and recommending tools that would improve the software-development environment.

Current External Experience

During the benchmark study of an outside organization, still another strategy emerged. In this case, researchers conducted spot audits at specific points in the life cycles of several projects. The objective was to evaluate especially high-risk projects in order to make informed decisions on how to proceed. The organization did not consider SQA to be a continuous activity (although it used to), but rather as a "snapshot" of selected high-risk projects. The motivation for this strategy was to reduce spending. However, the effectiveness of the SQA function was not clear.

In each of the above instances, the SQA group was not involved before development activities began. But, in yet another benchmark study, an outside organization placed the SQA team in a key position from the very beginning, at a time when a customer's contract was being negotiated. In this case, the estimated cost for the SQA group was based on the customer's quality requirements, and this estimated cost was reflected in the overall contract cost. The typical cost for SQA was six percent of the overall development cost. This figure changes, depending on a project's size and the customer-requested quality requirements.

Conclusion

The SEI capability-maturity model and the competition to obtain ISO-9000 registration have led to a significant "push" within AT&T—and the software industry in general—to implement a formal SQA value-added function. SQA is required, not only for meeting specific industry standards, but also for an effective and efficient software-development process. However, full implementation requires certain prerequisites, such as a comprehensive understanding of SQA, unwavering management support, and the ability to customize SQA processes to meet organizational needs. Attaining these prerequisites has accelerated development of a BCP on SQA for application within AT&T, and in determining the steps needed to implement the BCP.

References

1. *ISO 9000-1, ISO 9000-2, and ISO 9000-3*, International Organization for Standardization, CP 56, Rue de Varembe 1, CH-1211, Geneva, Switzerland, Phone: (41 22) 749 01 11.
2. M. C. Paulik et al., *Capability Maturity Model for Software, Version 1.1*, Technical Report CMU-SEI-93-TR-24, Carnegie Mellon University, Software Engineering Institute, Pittsburgh, Pennsylvania, August 1991, Phone: 412-268-7700.
3. W. S. Humphrey, *Managing the Software Process*, ISBN 0-201-18095-2, Addison-Wesley Publishing Company, Reading, Massachusetts, January 1989.
4. *ANSI/IEEE Standards 730-1984 and 984-1985*, Institute of Electrical and Electronics Engineers, New York City, Phone: 212-705-7900.
5. T. Gilb, *Principles of Software Engineering Management*,.ISBN 0-201-19246-2, Addison-Wesley Publishing Company, Reading, Massachusetts, January 1988.
6. J. D. Aron, *The Program Development Process, Part II—The Programming Team*, ISBN 0-201-14463-8, Addison-Wesley Publishing Company, Reading, Massachusetts, January 1983.

(Manuscript approved December 1993)

M. Hosein Fallah *is a department head in Quality, Engineering, Software, and Technologies (QUEST) at AT&T Bell Laboratories in Holmdel, New Jersey. He develops methods and practices—and provides consultative support—to the AT&T business units and divisions in total quality management, ISO-9000 quality standards, software quality, benchmark studies, and customer-satisfaction measurement and management. Mr. Fallah received a B.S. in engineering from the Abadan Institute of Technology in Iran. He also received M.S. and Ph.D. degrees in applied science from the University of Delaware in Newark. Mr. Fallah joined AT&T in 1978.*

Ahmad M. Jrad *is a member of the technical staff in Quality, Engineering, Software, and Technologies (QUEST) at AT&T Bell Laboratories in Holmdel, New Jersey. He is responsible for software development, quality-process engineering, and research on ISO 9001, as applied to software organizations and quality-assurance techniques. Mr. Jrad, who is an active member of the IEEE and the IEEE Computer Society, received B.S., M.S., and Ph.D. degrees in electrical engineering from the University of Pittsburgh in Pennsylvania. He joined AT&T in 1987.*