Quality Report SQAS96-003

Software Quality:
A Guide to Responsibilities and Resources

December 1996

Software Quality Assurance Subcommittee
of the
Nuclear Weapons Complex Quality Managers

United States Department of Energy
Albuquerque Operations Office

Abstract

This Guide provides guidance to the software managers, software developers, and software quality engineers at each DOE site. It is designed to assist them in their efforts to advance the exchange and widespread usage of good software management concepts and techniques throughout the DOE and its associated contractors and Laboratories. In addition, this document serves to support the work being done by the Software Quality Assurance Subcommittee (SQAS).

Acknowledgments

This Guide was prepared for the Department of Energy (DOE) by a Working Group of the DOE Quality Managers' Software Quality Assurance Subcommittee (SQAS). At the time this Guide was prepared, the Working Group had the following members:

Mike Blackledge	SA
Patty Trellue	SA
Faye Brown	OR
Kathleen Canal	HQ
John Cerutti	LA
Charles Chow	LL
Orval Hart	LA
Mike Lackner	KC
Dave Peercy	SA
Larry Rodin	PX
Don Schilling	KC
Pat Tempel	SA
David Vinson	PX

The Subcommittee gratefully acknowledges the help of the many other individuals and organizations whose material contributed to the research, review, and development of this Guide. References for information contained in this document can be found in Section 7.

Additional information about the SQAS is available at http://www.pantex.com/sqas/sqas.htm. Additional copies of this Guide are available from the Points of Contact listed in Section 5 of this document.

Table of Contents

1. Introduction

1.1 Purpose

This Guide serves to promote the exchange and advancement of good software quality concepts and techniques throughout the Department of Energy (DOE), its associated contractors, and laboratories. It provides guidance to individuals responsible for the development and quality of software products.

This Guide also acknowledges the work being done by the Software Quality Assurance Subcommittee (SQAS) -- a Technical Advisory Group for the DOE Quality Managers. All DOE sites have the opportunity to be represented on the SQAS and to contribute to and benefit from its work.

1.2 Document Overview

This document describes the quality responsibilities and resources for three categories of individuals involved in the development of software products. These individuals are:

- managers - see Section 2,
- software practitioners - see Section 3, and
- software quality engineers - see Section 4.

Section 5 presents ways to share information relating to software quality practices with others and to learn about software quality practices as they are defined by industry and professional organizations.

Section 6 contains information about the Software Quality Assurance Subcommittee (SQAS). The SQAS is a resource to all DOE sites. Two activities sponsored by the SQAS, Software Quality Assurance Forum and Software Quality Mini-Forum, are discussed in Section 6.

Information in this document that was collected from other sources is followed by a reference number in brackets. This reference number refers to the list of all references found in Section 7.

2. Responsibilities and Resources for Management

2.1 Upper Management Support

Without upper management support, efforts to improve the process by which software is developed or to integrate quality methods into the software development life cycle have limited chance of success. Consequently, keeping upper management aware of software quality activities is essential. Management can become involved in quality activities by:

- becoming familiar with software engineering concepts and software quality assurance,
- supporting training in software quality techniques and methods,
- budgeting for software engineering tools that can increase productivity,
- planning for quality,
- establishing a program to measure the quality of the software process and products, and
- supporting forums for information sharing.

2.2 Quality Management

Quality management concentrates on all repetitive, cyclical, or routine work and its improvement. Quality management defines processes, process owners, requirements for those processes, measurements of the process and the output of the process, and feedback channels. Deming's "chain-reaction-theory" states that improvement in quality always and automatically results in reductions in cost and schedule with increases in productivity and performance. [1]

The most effective strategy for managing quality in software products is to focus on improving the processes used to produce those software products. By focusing on process improvement, this strategy emphasizes use of the techniques of defect prevention rather than defect detection within the software development life cycle. [3] This strategy emphasizes long-term, rather than short-term, improvements in software products. An appropriate strategy might involve the following actions:

- Use the best requirements, design, implementation, and test methods that you can find.
- Continuously measure the ability of your process to produce quality products.
- Experiment with and measure the effectiveness of new tools and methods.
- Incorporate into your process new tools and methods that improve your performance.
- Start a defect prevention program as your process becomes predictable.
- Identify the sources of the most troublesome problems, determine their causes, and work to eliminate them.

Managers should track the progress and assess the performance of their process improvement strategy to ensure that it remains dynamic. Managers should support, encourage, and influence process improvement in the following areas relative to the software life cycle:

- project and contract management
- project planning and estimating
- risk management
- reuse
- defect detection, removal, and prevention
- process and product measurement
- software reviews and audits
- configuration management

2.3 Project Planning and Estimating

Project managers should develop a project plan that ensures the tasks and subtasks that comprise the life cycle phases of a software development effort are controlled and directed. This project plan should address work breakdown structure, project schedule and resources, project standards and conventions, configuration management, quality assurance, risk management, project reviews and audits, and project deliverables. This project plan provides a primary means of communications between the software project manager and upper management and between the software project manager and the software developers.

The cost-estimating methodology should establish procedures for estimating costs related to:

- the work breakdown structure;
- application attributes, e.g., costs related to project size and complexity, reliability requirements, interface requirements, program language requirements;
- environment attributes, e.g., costs related to procedures, scheduling, life cycle methods, utility software, application package acquisition;
- project team attributes, e.g., costs related to project team size, skill level, familiarity, experience, staffing constraints; and
- training support requirements, e.g., costs of travel, tuition, lodging.

2.4 Risk Management

Risk is defined as *the potential for realization of unwanted, negative consequences of an event. [4]* Projects risk failure in three ways:

- the product does not meet specified performance levels,
- actual costs are higher than budgeted, and
- delivery of the product is too late.

To be an effective manager, risks must be addressed and mitigated throughout the entire project life cycle. The manager will be hard pressed to eliminate all risks, but should take action on those risks most critical to the success of the project at the point where the risks can be managed. The risk management actions that should be taken by the project manager are:

- Identify -- search for and locate risks before they become problems that adversely affect your program;
- Analyze -- process risk data into decision-making information;
- Plan -- translate risk information into decisions and actions (both present and future) and implement those actions;
- Track -- monitor the risk indicators and actions taken against risks;
- Control -- correct for deviations from planned risk actions; and
- Communicate -- provide the visibility and feedback data internal and external to the program on activities and current emerging risks.

For large software developments, a risk management plan is a smart way to guide the risk management process and to document the results or status of the risk management process. [1]

2.5 Resources

The following resources provide guidance on the performance of software management activities.

a. ANSI/IEEE Std 1058.1-1987 (R1993), *IEEE Standard for Software Project Management Plans*, The Institute of Electrical and Electronics Engineers, Inc., 1993

This standard specifies the format and contents of software project management plans. It does not specify the procedures or techniques to be used in the development of project management plans, or does it provide examples of project management plans, instead the standard sets a foundation for an organization to guild it sown set of practices and procedures for developing project management plans.

b. EIA/IEEE Std J-STD-016-1995 *Interim Standard for Information Technology - Software Life Cycle Processes - Software Development: Acquirer-Supplier Agreement* (Issued for Trial Use), The Institute of Electrical and Electronics Engineers, Inc., 1995.

This standard is based on MIL-STD-498 *Software Development and Documentation* and ISO/IEC 12207 *Information Technology Software Life Cycle Processes* and defines a set of software development activities and resulting software products. It provides a framework for software development planning and engineering.

c. SQAS93-002 *Management and Software Quality Assurance: A White Paper*, Software Quality Assurance Subcommittee, June 1993.

This paper supports the position that management must play an integral role in making Software Quality Assurance activities and Software Engineering an effective part of their organization, and it addresses what management's commitment should be and the benefits of making such a commitment.

d. SQAS93-003, *Preferred Practices for Software Quality Within the Nuclear Weapons Complex*, Software Quality Assurance Subcommittee, October 1993.

This report presents a strategy for improving software quality within the Nuclear Weapons Complex. A software management program framework of customer requirements, site policy, site processes, and organization-specific methods is recommended. Implementation recommendations for baseline assessment, site policy, and site-preferred processes are included.

3. Responsibilities and Resources for Software Practitioners

A strong quality assurance program should be initiated at the beginning of a project and conducted throughout the software lifecycle. Software quality activities performed by software practitioners focus on the processes used to assure that the software product complies with applicable standards, satisfies requirements, and reduces defect propagation by providing for the detection, reporting, and correction of defects earlier in the project lifecycle.

A Software Management Plan serves as a site-specific guideline for establishing and maintaining control of software quality and integrity; and managing software acquisition, development, change, maintenance, and disposition. The plan may address issues such as software reuse or sharing practices, a software development methodology, and the use of computer-aided software engineering tools. [6]

This chapter provides an overview of the quality assurance responsibilities of software practitioners. The last section in the chapter identifies resources that are available to assist practitioners in accomplishing their quality assurance objectives.

3.1 Software Project Planning

Planning involves selecting the goals and objectives of the project and the strategies, policies, programs, and procedures for achieving them. *Planning is deciding, in advance, what to do, how to do it, when to do it, and who is going to do it.*

The software quality activities planned for a project should be formally documented in a Software Development Plan. The Software Development Plan is the controlling document for the software project. It specifies the technical and management approaches to be used and incorporates plans for quality assurance, configuration management, and verification and validation.

Standards should be established and used for all deliverable software documentation. The standards should be clearly stated or referenced in the Software Development Plan and should include measures to be applied to assure delivery of accurate and complete documentation.

A typical Software Development Plan includes the following types of information that are specific to the project:

- Project Organization
  - Project Overview
  - Project Boundaries and Interfaces
  - Project Deliverables
- Management Approach
  - Management Objectives and Priorities
  - Assumptions, Dependencies, and Constraints
  - Risk Management
  - Project Tracking and Oversight
  - Staffing
- Technical Approach
  - Lifecycle Model/Methodology
  - Methods, Tools, and Techniques
  - Software Documentation
  - Project Support Functions - each function may require its own plan (e.g., Quality Assurance, Configuration Management, Verification and Validation)
  - Test Strategy - test planning may be documented in its own project-level Software Test Plan
- Work Products
- Budget and Resource Allocations
- Schedule

3.2 Software Configuration Management

Software configuration management uses technical and administrative processes to identify, track, and control configuration items and the changes that are made to those items. A configuration item is any software or document component that is designated for configuration management and treated as a single entity in the configuration management process. Some typical configuration management activities include:

- Identify and use configuration management tools that are compatible with the size and scope of the project.
- Identify and document the functional and physical characteristics of configuration items.
- Establish and maintain baselines -- baselined software should undergo approval procedures to authorize and document changes.
- Establish formal procedures for evaluating and implementing changes.
- Identify change authorities and their responsibilities and determine escalation for problems/decision making.
- Control and track changes to the configuration items, e.g., create an audit trail of each change that is made, who made the change, why the change was made, and the date of the change.
- Record and report change processing and implementation status.
- Assess proposed modifications, enhancements, or additions to determine the effect each change will have on the product.

Library controls are the procedures and controls, manual or automated, for the handling of source code and object code in their various forms and versions, from the time of their initial approval or acceptance until they have been incorporated into the final deliverable software. The objectives of these controls include:

- Assure that different computer program versions are accurately identified and documented.
- Assure that a consistent software release process is used.
- Assure that no unauthorized modifications are made to the source code or object programs.
- Assure that all approved modifications are properly integrated.
- Assure that the software submitted for testing is the correct version.
- Assure that infrequently used software is properly archived and stored.

3.3 Measurement/Metrics

Measurements are an essential tool for monitoring, controlling, and improving products and processes. Software quality metrics provide the basis for data-driven software project management, quality software engineering, and continuous process improvement. An effective measurement program addresses the critical goals of a project; collects data that is timely, accurate and relevant; and produces useful information that leads to product and process improvement.

The following steps can be used to establish a measurement program:

- Determine project goals
- Specify quality attributes to be measured
- Select metrics
- Collect and analyze data
- Report results
- Implement product and process improvements

The following are examples of typical product quality metrics:

- Requirements stability
- Defect density
- Defect-type distribution
- Complexity
- Failure intensity

The following are examples of typical process improvement metrics:

- Historical project variances
- Project rework
- Productivity

3.4 Peer Reviews

Peer reviews are one of the most effective activities used to identify defects in work products (e.g., a design document, user manual, or section of code). Reviews take three principal forms with increasing degrees of formality.

- Informal Review - A work product is sent to a group of knowledgeable people for review and comments. Written comments are returned to the author for determination of the action to be taken.
- Structured Walkthrough - A formal review process that involves a review meeting and requires reviewers to take a concur/nonconcur position on the work product. A nonconcurrence requires rework of the product and may necessitate another walkthrough at the reviewers' discretion.
- Inspection Review - The most comprehensive form of review. Inspections are intended to remove defects in a work product and improve the process by which future similar work products will be developed. Code inspections are one type of inspection review.

3.5 Defect Data Collection

Identifying defects in software is just the beginning of the defect prevention program. To have an effective prevention program, software practitioners should perform the following activities:

- Define and implement a defect data collection process.
- Establish a tracking mechanism, automated or manual, for reporting defects, causes, solutions, and corrective actions. Include measures for identifying the phase in which the defect occurred (e.g., requirements gathering) and the type of defect (e.g., computational or logic error).
- Collect and analyze defect data, search for the root cause of defects, and look for ways to avoid or eliminate similar defects in future projects.

3.6 Requirements

Central to software development is the performance of a detailed analysis of the user requirements and any additional requirements flowing from general purpose administrative practices and procedures. Maximize the use of structured techniques and tools to assure a thorough and comprehensive description of all requirements.

Subject each product of the requirements definition phase to a peer review to assure a clear and consistent set of requirements. The peer reviews check the quality of the items and should focus on the technical validity, testability, and completeness of the requirements. Inspect requirements diagrams and data dictionary entries as they are prepared. As each new diagram is generated, anticipate changes (usually refinements or improvements) to previously certified diagrams and follow established software configuration management procedures. [5]

The functional baseline, or system requirements baseline, is the main technical product of the requirements definition effort. For major software development projects, plan a system requirements review of the functional baseline. This review should demonstrate that the software capabilities, user interface, and security features satisfy all of the system requirements allocated to the software. Summarize the results of any prototyping effort and include a demonstration of prototypes used to derive or validate requirements. Present evidence that the software can be generated on time and within budget. [5]

Establish a technique for tracing requirements through the development process. A Requirements Traceability Matrix is one tool used to show how the requirements drive the design, coding, and testing processes. Show interrelationships in the matrix. [1]

3.7 Software Design

The design flows from the software requirements. Subject the products from the design phase to peer reviews to assure that the design fulfills the requirements; complies with specified standards; and is understandable, consistent, and complete. Use a traceability matrix to demonstrate that the design accounts for each software requirement. Audits of the design documentation should be accomplished prior to release of the software design for coding.

A Preliminary Design Review is held at the completion of the preliminary design phase and demonstrates the ability of the selected design approach to satisfy the functional, data, and interface requirements. Review the planned user interfaces to the software and, if applicable, the design of the data base to be managed. Demonstrate rapid design prototypes used to make design decisions. Identify potential high risk areas in the design and any requirements changes that could reduce risk. [6]

A Critical Design Review is held at the completion of the detailed design phase and demonstrates the ability of the design approach to supply the functionality specified in the preliminary design. Show that the design adequately accounts for all software and data requirements and that it can be implemented on the target computer platform. Review the validity of algorithms needed to perform critical functions. Present evidence that the design satisfies all relevant performance, interface, security, safety, resource, and reliability requirements. [5]

3.8 Implementation

To ensure consistent quality, compliance with software and security requirements, and meaningful status reports, subject each implementation product to inspection and certification before reporting it complete. The following are some typical implementation inspections and certifications.

- Inspect new or substantially changed units of code,
- Conduct program code inspections to inspect and certify for correctness and completeness,
- Inspect the program description, logic description, and test plan for each program,
- Conduct program test inspections to certify the correctness and completeness of each tested program, and
- For software with stringent security requirements, schedule comprehensive security inspections conducted by the organization's Assistant Computer Protection Program Manager or Computer System Security Officer.

3.9 Testing

Planning for software testing should start in conjunction with project planning. A project-level Software Test Plan should be developed for all software products within a software system. This test plan establishes the testing activities necessary to validate that the software requirements have been met and to verify the functionality of the software. The plan also documents a systematic approach to testing throughout the software lifecycle.

A comprehensive test plan includes the following types of information:

- Levels of testing (e.g., unit, integration, system, and acceptance);
- Types of tests to be performed (e.g., functional performance, usability, stress, regression, and real-time response);
- Testing strategies (e.g., top down, bottom up, automated, first, beta, black box, white box); and
- Test design (e.g., test cases, fault insertion/error handling, usage scenarios).

3.10 Resources

The following resources provide guidance on the performance of software engineering activities.

a. ANSI/IEEE Std 829-1983 (R1991), *IEEE Standard for Software Test Documentation*, The Institute of Electrical and Electronics Engineers, Inc., 1991.

This standard describes the form and content for a set of basic software test documents covering test planning, test specification, and test reporting.

b. ANSI/IEEE Std 830-1993, *IEEE Recommended Practice for Software Requirements Specifications*, The Institute of Electrical and Electronics Engineers, Inc.,1993.

This document describes alternate approaches to good practices in the specification of software requirements.

c. EP401045, *Engineering Procedure, Definition of Computer Software Configuration Items*, November 1990.

This Interagency Engineering Procedure (EP) explains the definition, identification, control, and storage of software used in the operation, evaluation, or acceptance of weapon product. The purpose of this EP is to standardize documentation and configuration control practices for such software.

d. ANSI/ASME NQA-1, Part II, Subpart 2.7 *Quality Assurance Program Requirements for Nuclear Facilities, Requirements of Computer Software for Nuclear Facility Applications*, American Society of Mechanical Engineers, released 1994. (Subpart 2.7 of NQA-1).

This document prescribes basic quality principles and requirements for nuclear facilitates.

e. SQAS93-003, *Preferred Practices for Software Quality Within the Nuclear Weapons Complex*, Software Quality Assurance Subcommittee, October 1993.

This report presents a strategy for improving software quality within the Nuclear Weapons Complex. A software management program framework of customer requirements, site policy, site processes, and organization-specific methods is recommended. Implementation recommendations for baseline assessment, site policy, and site-preferred processes are included.

f. SQAS96-001, *Preparation for a Software Quality Audit*, Software Quality Assurance Subcommittee, June 1996.

This document provides specific guidance that enables a site to prepare for a software quality audit. It also provides guidance for performing a software quality audit.

4. Responsibilities and Resources for Software Quality Engineers

Quality assurance ensures the development of high quality products on schedule and within the constraints specified by the user(s). In *Quality Assurance for Computer Software*, Dunn and Ullman define software quality assurance as preventing problems from occurring, removing defects, contributing to the usability and maintainability of software, and through the analysis of defect histories, improving the production rate of deliverable code. [2]

Quality assurance practices and techniques should be integrated into the management and technical activities of each project. The software quality assurance activities planned for a project should be formally documented in a Software Quality Assurance Plan. The plan describes all tasks that must be performed, provides standards against which both the software being developed and progress toward completion can be measured, and delineates the way in which quality will be determined, including the expected metrics.

Systems engineers performing quality assurance functions should have the responsibility and authority to evaluate software development activities, and to recommend improvements in accordance with the project's plans.

This chapter provides an overview of the quality assurance responsibilities of software quality engineers. The last section in the chapter identifies resources that are available to assist quality engineers in accomplishing their quality assurance objectives.

4.1 Audits and Reviews

An audit is an independent examination of a work product to assess compliance with specifications, standards, contractual agreements, or other criteria. A review is an independent evaluation of an activity or process to assess compliance with the Software Development Plan.

Software quality engineers should review processes and audit work products throughout the software lifecycle to verify that they comply with the applicable project plans, procedures, and standards. The results of the audits and reviews are shared with the appropriate project managers and teams to facilitate continuous improvement in processes and products and to promote a reduction in practices that produce defects.

Software quality engineers should establish procedures for the preparation and execution of audits and reviews to determine if customer needs and project objectives are being satisfied. The tools, techniques, and methodologies that will be employed in the performance of the quality assurance audits and reviews should be included in the Software Quality Assurance Plan.

The following list contains examples of typical audits and reviews exercised throughout the software life cycle.

- Audit of Software Development Plan
- Audit of software requirements to determine correctness, consistency, completeness, and testability
- Review of requirements traceability (e.g., requirements to design, design to code, code to testing)
- Review of software configuration management process
- Audit of preliminary and detailed designs (e.g., design satisfies requirements)
- Audit of program/source code (e.g., code adheres to standards, is well documented, and appears to be maintainable)
- Audit of test plans
- Review of test procedures
- External audits performed to determine supplier qualifications or certification

4.2 Corrective Action

Corrective action is any measure related to the prompt correction of defects or deficiencies that might result in noncompliant software. The following list contains some common corrective actions performed by quality engineers.

- Analysis of data and examination of problem and deficiency reports to determine extent and cause(s)
- Identification and analysis of trends in defect occurrences
- Providing feedback to software practitioners to prevent the propagation of defects
- Monitoring the implementation of corrective measures to determine their effectiveness

Developing a history of defects based on all work done according to a set of software lifecycle processes will allow identification of the effects of altering the processes and continuously improve software quality and productivity by offering evidence for adjusting the processes.[5]

4.3 Testing

Software quality engineers should develop a schedule of audits and reviews of the testing program based on the testing activities documented in the project's test plan. The following are typical testing audits and reviews.

- Audit of test plans for compliance with appropriate standards and requirements.
- Audit of test requirements and criteria for adequacy, feasibility, and satisfaction of requirements.
- Review of test procedures for compliance with appropriate standards and satisfaction of requirements.
- Monitoring of tests and certification that test results are the actual findings of the tests.
- Completion and certification of test reports.
- Verification that test-related documentation is maintained to allow repeatability of tests and regression testing.

4.4 Metrics

Software quality engineers use metrics to compare actual software development progress against expected results. The data obtained from the metrics can generate reliable projections of project status, provide early warnings of potential problems, and indicate areas needing process improvements.

The quality metrics that will be used on a project should be described in the Software Quality Assurance Plan. Examples of typical software product and process metrics are described in Section 3.3.

The following are examples of typical areas for software product quality metrics.

- Efficiency
- Functionality
- Maintainability
- Portability
- Reliability
- Usability

4.5 Software Process Self-Assessments

A software process self-assessment is an informal and less expensive implementation of the more formal assessment typically conducted by experienced Software Engineering Institute (SEI) approved assessors. Organizations that are not ready to invest the resources for a full formal assessment should evaluate each step of the formal assessment plan and determine if a less costly version of the assessment can be implemented by in-house personnel. The goal of the software quality engineers should be to develop an assessment plan that the organization has the ability and resources to implement.

4.6 Resources

The following resources provide guidance on the performance of software quality engineering activities.

a. ANSI/IEEE Std 1012-1986 (R1992), *IEEE Standard for Software Verification and Validation Plans,* The Institute of Electrical and Electronics Engineers, Inc., 1992.

This standard provides uniform and minimum requirements for the format and content of Software Verification and Validation Plans. Performing software verification and validation in accordance with this standard provides for a comprehensive evaluation throughout each phase of the software project to help assure errors are detected and corrected as early as possible in the lifecycle and software quality and reliability are enhanced.

b. ANSI/IEEE Std 1059-1993, *IEEE Guide for Software Verification and Validation Plans,* The Institute of Electrical and Electronics Engineers, Inc., 1993.

This guide provides guidance in preparing Software Verification and Validation Plans that comply with IEEE Std. 1012-1987. It recommends approaches to verification and validation planning.

c. ANSI/IEEE Std 1028-1988, *IEEE Standard for Software Reviews and Audits,* The Institute of Electrical and Electronics Engineers, Inc., 1988.

This standard defines the review and audit processes applicable to critical and noncritical software, and provides specific procedures required for the execution of these reviews and audits.

d. SQAS95-001, *Planning for a Software Process Assessment,* Software Quality Assurance Subcommittee, May 1995.

This report presents a guide for planning a software process assessment and how to use the assessment results to guide process improvement. It defines the steps associated with conducting a successful assessment and identifies the framework needed to establish a successful process improvement program. The assessment methodology is based on the Carnegie-Mellon Software Engineering Institute's software process assessment.

e. SQAS96-001, *Preparation for a Software Quality Audit,* Software Quality Assurance Subcommittee, June 1996.

This document provides guidance for performing a software quality audit. It also provides guidance that enables a site to prepare for a software quality audit.

f. SQAS96-002, *Guideline for NWC Processes for Handling Software Product,* Software Quality Assurance Subcommittee, June 1996.

This report provides guidelines for handling software as product within the Nuclear Weapons Complex through identification, qualification, acceptance, and delivery processes. The guidelines focus on weapon and weapon-related software products for customers such as NWC sites, the Department of Energy, and agencies of the Department of Defense.

g. SQAS96-003, *Guidelines for Software Measurement,* Software Quality Assurance Subcommittee, 1996

This report provides guidelines for a core set of software measures (i.e., size, effort, progress related to project schedule, and product defects) that can be used for development and support of software product.

5. Opportunities for Information Exchange

Dispersing SQA information at each DOE site is an on-going task involving several types of resources. Suggestions for ways to disperse SQA information include the following:

- Electronic Mail
- Newsletter
- Phone Line
- Software Quality Organization
- Quality Manager
- Site-specific Resources
- SQAS Publications
- Web Page (Local or WWW)

The remainder of this section provides examples of how information is being dispersed at various DOE sites and provides a contact at that site for additional information.

5.1 AlliedSignal Aerospace - Federal Manufacturing & Technologies

The Software Quality Assurance (SQA) program at the DOE Kansas City Plant operated by AlliedSignal Aerospace is incorporated into the formal on-line Command Media system that defines all management practices in support of the Plant's ISO9001 certification. The ISO SQA Process Description, included as an element of Quality Management and part of the Plant's Business Planning and Control system, outlines the SQA responsibilities for all line organizations. Implementation oversight of the SQA program is the responsibility of the Plant's SQA Group, which has promoted SQA awareness and knowledge since its inception in 1988.

The SQA group provides information and conducts software process assessments to continuously improve the Plant's software development, procurement, maintenance and use performance. Guidelines and activities supporting information sharing include:

- the "Software Quality Assurance Handbook",
- the "Purchased Software SQA Handbook",
- the SQA process assessment program, and
- routine SQA Information Exchanges.

Points of Contact:

Don Schilling
AlliedSignal - FM&T
MS 2B27
P.O. Box 419159
Kansas City, MO 64141-6159
Phone: (816) 997-4035
FAX: (816) 997-7325
E-Mail: dschilling@kcp.com

Cathy Kuhn
AlliedSignal - FM&T
MS 2B27
P.O. Box 419159
Kansas City, MO 64141-6159
Phone: (816) 997-4762
FAX: (816) 997-7325
E-Mail: ckuhn@kcp.com

5.2 DOE-Albuquerque Operations Office

Point of Contact: Gary Echert
DOE-AL, Weapons Quality Division (WQD)
P.O. Box 5400
Albuquerque, NM 87185
Phone: (505) 845-4255
FAX: (505) 845-4722
E-Mail: gechert@doeal.gov

5.3 DOE Headquarters

The Department of Energy Software Engineering Methodology (SEM) dated March 1996 is the primary vehicle used to communicate the standard methodology for software development and maintenance at Headquarters. The SEM is a major component of the Departmental Software Management Program.

The SEM offers a process-oriented approach to project management, software engineering, and quality assurance. Appendixes provide detailed information about project management practices such as structured walkthroughs (peer reviews), and In-Stage Assessments and Stage Exits (quality reviews and audits). Supplemental materials are available in areas such as project planning, configuration management, and testing.

Training, consultations, and other types of support are provided for the processes described in the SEM. Some of the most frequently requested services include the following:

- Project planning and tracking,
- In-Stage Assessments,
- Stage Exits,
- Structured walkthroughs,
- Requirements management,
- Software quality assurance, and
- Independent project assessment and follow-up consultations.

The Headquarters Human Resources and Administration Home Page on the World Wide Web (http://www.hr.doe.gov) is also used to communicate timely Software Management Program and software quality assurance information throughout the Department. Headquarters is currently looking at the viability of placing the SEM files on the Internet for Department-wide online access.

An Office of Information Management (HR-4) Technical Research Center is maintained at Headquarters. The Center serves as a central source of research support for information management/technology, as the collection and dissemination point for current awareness products and services, and as a repository of software project documentation and technical reference material.

Technical Operating Procedures are available for many routine software management practices such as change procedures for systems under development and in production, system acceptance and certification, and system retirement guidelines.

A wide range of lectures and programs of interest to the software and quality assurance communities are offered at Headquarters. Many programs are transmitted throughout the Headquarters internal cable television system and broadcast Department-wide over compressed video teleconferencing facilities and satellite links.

The Microcomputer Application System Library (MASL) is a Headquarters-based repository for software products that have successfully completed the acceptance and certification (where applicable) processes. Abstracts about the applications are available for online browsing and copies of the source code are available for sharing and reuse.

The Engineering Services Team, HR-433, is planning to communicate software quality assurance and other software management program information Department-wide by maintaining a HomePage within the Headquarters Human Resources and Administration HomePage on the World Wide Web (http://www.hr.doe.gov/sem/smpqa_01.htm).

Points of Contact: Wayne Jones, HR-433
Engineering Services Team
U.S. Department of Energy
Headquarters - Germantown
19901 Germantown Road
Germantown, MD 20585
Phone: (301) 903-4655
FAX: (301) 903-6223
E-Mail: wayne.jones@hq.doe.gov

Kathleen Centeno, HR-433
Engineering Services Team
U.S. Department of Energy
Headquarters - Germantown
19901 Germantown Road
Germantown, MD 20585
Phone: (301) 903-5472
FAX: (301) 903-2261
E-Mail: kathleen.centeno@hq.doe.gov

5.4 DOE-Nevada Operations Office

DOE/NV maintains and disperses Software Quality Assurance (SQA) information from the Technical Information Resource Center (TIRC). Presently, SQA information is dispersed from the TIRC to the Department Managers of each of the respective information systems organizations. The TIRC partners and teams with the Single Point of Contact, from within the DOE/NV Communication Services Division, to ensure that interested and required DOE/NV Federal, contractors, Laboratory, and Work for Others (the NV Enterprise) parties receive the latest information and news concerning software quality.

Point of Contact: Mike Maier
DOE/NV- Bechtel Nevada Corporation
Communications Service Division
2753 S. Highland Drive
Las Vegas, NV 89109
Phone: (702) 295-5836
FAX: (702) 295-6614
E-Mail: maier@nv.doe.gov

5.5 Lawrence Livermore National Laboratory

LLNL has established the Software Technology Center (STC) which supports software engineering and software quality assurance for the Laboratory. The STC is a multi-functional facility that supports the application of appropriate software engineering technologies and tools via regularly scheduled software engineering in-house classes. The STC home page (http://www.llnl.gov/stc/) supports communication of software engineering and software quality tenets. Through its working group, the STC facilitates the sharing of knowledge among software professionals including publishing the *Software Engineering Newsletter*, and STC consultants provide expertise in methods, technologies, and tools to LLNL and customer projects.

Points of Contact: Carolyn Owens
P.O. Box 808 L-792
Livermore, CA 94550
Phone: (510) 423-6085
FAX: (510) 422-3400
E-mail: owens1@llnl.gov

Bill Warren
P.O. Box 808 L-308
Livermore, CA 94550
Phone: (510) 422-5331
FAX: (510) 422-1226
E-mail: warren6@llnl.gov

5.6 Los Alamos National Laboratory

Point of Contact: John Cerutti
Group XHM, MS F663
P.O. Box 1663
Los Alamos, NM 87545
Phone: (505) 667-0387
FAX: (505) 665-5553
E-Mail: jhc@lanl.gov

5.7 Lockheed Martin Energy Systems, Inc.

The primary method of dispersing SQA information at Lockheed Martin Energy Systems Y-12 Plant is by using SQA Coordinators from each organization. Information relating to SQA is sent to each SQA Coordinator for distribution among their organization. In addition, the Energy System Newswire newsletter and an on-line news service from Inside Energy Systems, updated daily, is available to all employees.

Point of Contact: Y. Faye Brown
Building 9119, M/S 8234
Y-12 Plant, P.O. Box 2009
Oak Ridge, TN 37831-8234
Phone: (423) 574-3238
FAX: (423) 576-4968
E-Mail: brownyf@ornl.gov

5.8 Mason & Hanger - Silas Mason Co.

The formal system at Pantex consists of a policy directive and several plant procedures. This system is supplemented by the following for specific SQA information sharing:

- Plant Manual MNL-00036, "Software Quality Life Cycle Guidelines", and
- Computer Based Training Course 183.01, "Software Quality Life Cycle"

The manual includes information and interpretation of what is in the formal procedures, as well as blank templates and completed samples of all required documentation. The computer based training is interactive, and is specific to the users needs (customers, development team members, or software engineers take different paths through the training). In addition, SQA Questionnaires are distributed to Department Managers every two to three years.

All of the following general methods of information exchange have been used for SQA information:

- The Pantexan is newspaper that is published monthly and mailed to employees' homes. It contains information of interest to both employees and their families;
- The Pantex Pulse, a weekly publication, contains information on issues of employee interest; and
- The "Grapevine", an electronic medium on all networks, transmits information of interest Plant-wide and is updated daily.

Points of Contact: Larry Rodin
Pantex Plant, Quality 12-102
P.O. Box 30020
Amarillo, TX 79120
Phone: (806) 477-4259
FAX: (806) 477-4809
E-Mail: lrodin@pantex.com

Alvin Cowen
Pantex Plant, IMD 12-106A
P.O. Box 30020
Amarillo, TX 79120
Phone: (806) 477-3623
FAX: (806) 477-5636
E-Mail: acowen@pantex.com

5.9 Sandia National Laboratories

Sandia's primary mechanism for dispersing Software Quality information is through the Software Management Program. Within this program, there are several methods used to provide information, as described below.

Sandia established a Lab-wide Software Management Program (SMP) in 1992 to ensure that a consistent approach is applied to the management of all software assets being acquired, developed, supported, and used. The Sandia Software Management Program is established through teamwork and supported through policy. Each Vice President is invited to nominate two representatives to be members of a Quality Improvement Team to serve as the Technical Advisory Group to Sandia's upper management for the effective management of software. The Team presented an Implementation Plan to DOE/AL and has accomplished several Work Items. Some of the Work Items included establishment of a Corporate Software Management Policy via Sandia Laboratories Policy (SLP 1011), administration of a corporate-wide survey on software practices, and identification of liaisons with other software related initiatives within Sandia, the Albuquerque community, and the Nuclear Weapons Complex. In addition to the ongoing efforts in Software Quality awareness, two significant Work Items currently underway are: conducting Software Process Assessments of Sandia organizations and establishing a Software Management HomePage on the Sandia Internal Web.

The Single Point of Contact for the Software Management Program maintains a distribution list of Sandians who have indicated an interest in Software or Software Quality. This list is used for dispensing information about upcoming events or lectures concerning software quality. The SMP has linked with Phillips Laboratory and UNM/Computer Science Department to initiate and establish an active Software Process Improvement Network (SPIN) (http://www.highfiber.com/~waggoner/spin/) within the Albuquerque Area.

Points of Contact: David E. Peercy
Division 12326, M/S 0638
P.O. Box 5800
Albuquerque, NM 87185-0638
Phone: (505) 844-7965
FAX: (505) 844-3920
E-Mail: depeerc@sandia.gov

Patricia A. Trellue
Division 2612, M/S 0519
P.O. Box 5800
Albuquerque, NM 87185-00519
Phone: (505) 845-9734
FAX: (505) 844-9478
E-Mail: patrell@sandia.gov

5.10 Westinghouse Savannah River Co.

Point of Contact: Raymond Cullen
Savannah River Site, Bldg. 233-6H
P.O Box 616
Aiken, SC 29808
Phone: (803) 557-1914
FAX: (803) 557-1450
E-Mail: ray.cullen@srs.gov

5.11 Industry and Professional Organizations

The purpose of this section is to provide a pointer toward organizations, and individuals within the SQAS, who are knowledgeable about software quality assurance. The list of organizations is by no means exhaustive but does point out a few of the major liaisons in software quality assurance and related focus areas.

5.11.1 American Society for Quality Control (ASQC) - Software Division

Purpose: ASQC's mission is to be the world's leading authority and recognized champion on issues related to Quality

Address: American Society for Quality Control (ASQC)
Software Division
P. O. Box 3005
Milwaukee, WI 53201-3005

Contact: http://www.asqc.org/

5.11.2 Institute for Electrical and Electronic Engineers (IEEE) - Computer Society

Purpose: To facilitate the initiation, development, promulgation, international adoption and maintenance of computer and computer related standards

Address: Institute for Electrical and Electronic Engineers (IEEE) - Computer Society
1730 Massachusetts Avenue, N.W.
Washington, D.C. 20036-1903

Contact: http://www.computer.org/cshome.htm

5.11.3 International Standards Organization (ISO)

Purpose: ISO, along with IEC (International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with a particular field of technical activity.

Address:

Case Postale 56
CH-1211 Geneve 20
Switzerland

American National Standards Association
11 West 42nd Street
New York, NY 10036

Contact: http://www.iso.ch/

5.11.4 Joint Working Group - JOWOG 39

Purpose: To facilitate the exchange and peer review of information, as well as fostering collaborative programs, in all aspects of manufacturing between the United States through the NWC and the United Kingdom through the Atomic Weapons Establishment (AWE). An area of Focused Exchange is SQA which includes the review and common understanding of standards and business practices.

Address: None.

Contacts:

Don Schilling, US-JOWOG 39 SQA Liaison
AlliedSignal Aerospace
Federal Manufacturing & Technologies
P. O. Box 419159
Kansas City, MO 64141-6159
Phone: 816-997-4035

John Hare, UK- JOWOG 39 SQA Liaison
AWE-Aldermaston
Reading, Berkshire, RG7 4PR
England, UK
Phone: 1734 824094

6. Software Quality Assurance Subcommittee

The Software Quality Assurance Subcommittee (SQAS) serves as a Technical Advisory Group on Software Engineering and Quality initiatives and issues for the Department of Energy's Quality Managers. The Subcommittee provides a continuing forum as a vehicle to exchange information and work issues in the area of software quality engineering. The SQAS web site is located at http://www.pantex.com/sqas/sqas.htm.

6.1 Background

The Subcommittee grew out of a Software Quality Assurance Information Exchange Forum which was sponsored in March of 1988 by the DOE-AL Quality Engineering Division, hosted by Sandia National Laboratories, and with participation by all of the sites within the DOE's Nuclear Weapons Complex at that time. The participants felt that a continuing forum was needed as a vehicle to exchange information and work issues in the area of software quality engineering. At that time the Quality Managers from the Nuclear Weapon Complex's design agencies and production agencies were sponsoring subcommittees that would report back to them at their semi-annual meetings, and in response to the forum, DOE-AL agreed to sponsor a new subcommittee -- the SQAS. An organizational meeting was held in August of 1988 at Sandia, and following that, the draft Charter, Implementation Plan, Recommended Officers, and Initial Work Items were presented to, and approved by, the Quality Managers.

6.2 Structure

The Subcommittee operates under the sponsorship of the Quality Managers. Each Quality Manager from each participating DOE site names one primary and one alternate representative from that site. In recent years, the membership of the Subcommittee has been expanded to allow representation from any of the DOE laboratories and facilities that expressed an interest in participating, and sponsorship was extended to include the DOE Information Management (IM) Council, under the office of the Chief Information Officer. Details on the operation of the SQAS, including the Charter and Implementation Plan, are available in the SQAS Operating Procedures.

6.3 Services

The primary work of the Subcommittee is accomplished through formal Work Items, each of which is defined with specific objectives, deliverables, and milestones. Such Work Items may be proposed by the SQAS itself or by the Quality Managers, but all must be approved by the Quality Managers. No separate funding is provided for the operation and meetings of the Subcommittee, or for the accomplishment or publication of the deliverables. Subject matter for Work Items has included revision of the Software Quality section of DOE-AL's QC-1, review of DOE Order 1330.1D, and publication of several deliverables, including:

- SQAS90-001, NWC Glossary of Preferred Software Engineering Terminology
- SQAS90-002, Software Within the Nuclear Weapons Complex: 1989 Software Engineering Survey
- SQAS91-001, Abstracts of the Nuclear Weapons Complex Software Quality Assurance Requirements
- SQAS93-001, CASE Tools: Culture and Implementation
- SQAS93-002, Management and Software Quality Assurance: A White Paper
- SQAS93-003, Preferred Practices for Software Quality within the Nuclear Weapons Complex
- SQAS93-004, NWC Software Training Directory
- SQAS94-001, Status of Licensing and Certification of Software Professionals
- SQAS94-002, Software Transfer Guideline
- SQAS94-003, Software Engineering Certification: Contacts and Applications
- SQAS95-001, Planning for a Software Process Assessment
- SQAS96-001, Preparation for a Software Quality Audit
- SQAS96-002, Guideline for NWC Processes for Handling Software Product

An annotated bibliography of all SQAS publications is available from the SQAS web site at http://www.pantex.com/sqas/sqas.htm.

6.3.1 Software Quality Forum

The Software Quality (SQ) Forum is a three day forum held every three years. Past and scheduled SQ Forums are identified in the table below. Topics presented at the SQ Forum include: testing, metrics, safety/security/reliability, SQA practices, training, assessments, certification and licensing of software professionals, CASE tools, software project management, inspections, and management's role in ensuring SQA.

Date	Site
Spring 1988	Sandia National Laboratories
Spring 1991	Allied Signal Kansas City Plant
Spring 1994	Lawrence Livermore National Laboratory
Spring 1997	Sandia National Laboratories

The SQ Forum was originated by the SQAS as a opportunity for all those involved in implementing SQA programs to meet and share ideas and concerns. All DOE sites are encouraged to participate. This diversity of managers, quality engineers, and software professionals provides a ideal environment for identifying and discussing many issues and concerns raised by the Forum participants and speakers. The interaction provided by the Forum contributes to the realization of a shared goal -- high quality software product.

6.3.2 Software Quality Assurance Subcommittee Mini-Forums

Since the Software Quality Assurance Subcommittee (SQAS) meets only twice a year, and at a different DOE facility each time, a tradition has developed of having the host site sponsor a mini-forum in conjunction with the SQAS meeting. The purpose of these mini-forums is to provide a two-way exchange of information between the host site and the SQAS members. This allows site personnel the opportunity to learn about the activities of the SQAS and allows SQAS members the opportunity to become better acquainted with some of the software efforts underway at the host site.

7. References

[1] Department of Air Force Software Technology Support Center, *Guidelines for Successful Acquisition and Management of Software Intensive Systems: Weapon Systems, Command and Control Systems, Management Information Systems*, Volume 1, February 1995.

[2] Dunn, Robert and Richard Ullman, *Quality Assurance for Computer Software*, McGraw-Hill Book Company, 1982.

[3] Humphrey, Watts S., *A Discipline for Software Engineering*, SEI Series in Software Engineering, Addison-Wesley, May 1995.

[4] Rowe, W., *An Anatomy of Risk*, J. Wiley and Sons, New York, 1977.

[5] U.S. Department of Energy, *DOE/NV Software Management Plan*, Nevada Operations Office, May 24, 1991.

[6] U.S. Department of Energy, *Software Management Guide*, DOE/AD-0028, June 1992.