This document was prepared in conjunction with work accomplished under Contract No. DE-AC09-96SR18500 with the U. S. Department of Energy.

A "Toolbox" Equivalent Process for Safety Analysis Software

Kevin R. O'Kula

Tony Eng

Abstract

Defense Nuclear Facilities Safety Board (DNFSB) Recommendation 2002-1 (*Quality Assurance for Safety-Related Software)* identified a number of quality assurance issues on the use of software in Department of Energy (DOE) facilities for analyzing hazards, and designing and operating controls that prevent or mitigate potential accidents. The development and maintenance of a collection, or "toolbox," of multiple-site use, standard solution, Software Quality Assurance (SQA)-compliant safety software is one of the major improvements identified in the associated DOE Implementation Plan (IP). The DOE safety analysis toolbox will contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, recognized for DOE-broad, safety basis applications. Currently, six widely applied safety analysis computer codes have been designated for toolbox consideration. While the toolbox concept considerably reduces SQA burdens among DOE users of these codes, many users of unique, single-purpose, or single-site software may still have sufficient technical justification to continue use of their computer code of choice, but are thwarted by the multiple-site condition on toolbox candidate software. The process discussed here provides a roadmap for an equivalency argument, i.e., establishing satisfactory SQA credentials for single-site software that can be deemed "toolbox-equivalent".

The process is based on the model established to meet IP Commitment 4.2.1.2: *Establish SQA criteria for the safety analysis "toolbox" codes*. The primary criteria guiding the evaluation are those provided by ASME NQA-1-2000 and Subpart A to 10 CFR 830. Implementing criteria that establish the set of prescriptive SQA requirements are based on implementation plan/procedures from the Savannah River Site, also incorporating aspects of those from the Waste Isolation Pilot Plant (SNL component) and the Yucca Mountain Project. The major requirements are met with evidence of a software quality assurance plan, software requirements and design documentation, user's instructions, test report, a configuration and control procedure, an error notification and corrective action process, and evidence of available training on use of the software. The process is best performed with an independent SQA evaluator, i.e., a technically knowledgeable individual in the application area who is not part of the development team.

The process provides a consistent, systematic approach based on the experience gained with SQA evaluations of the toolbox codes. Experience has shown that rarely will existing software be fully compliant with SQA criteria. Instead, the typical case is where SQA elements are deficient. For this case, it is recommended that supplemental remedial documentation be generated. Situations may also arise where the SQA evaluator must weigh whether the entire SQA suite be reconstituted. Regardless, the process is described sufficiently to guide a comprehensive evaluation. If the candidate software is successful in meeting process requirements, the software is "toolbox-equivalent".

The benefit of the methodology outlined is that it provides a standard evaluation technique for choosing the most applicable software for a given application. One potential outcome is that the software of choice will be found to be applicable with ample SQA justification. Alternatively, the software in question may be found not to meet SQA process requirements. In this case, the analyst may then make an informed decision and possibly select one of the multiple-use, toolbox codes. With either outcome, the DSA is improved.

Introduction

In January 2000, the Defense Nuclear Facilities Safety Board (DNFSB) issued Technical Report 25, (TECH-25), *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*.[1] TECH-25 identified issues regarding computer software quality assurance (SQA) in the Department of Energy (DOE) Complex for software used to make safety-related decisions, or software that controls safety-related systems. Instances were noted of computer codes that were either inappropriately applied, or were executed with incorrect input data. Of particular concern were inconsistencies in the exercise of SQA from site to site, and from facility to facility, and the variability in guidance and training in the appropriate use of accident analysis software.

To expedite implementation of corrective actions in the SQA area, the DNFSB issued Recommendation 2002-1, *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*.[2] As part of its Recommendation, the DNFSB enumerated many of the points noted earlier in TECH-25, but noted specific concerns regarding the quality of the software used to analyze and guide safety-related decisions, specifically to analyze hazards, and design and operate controls that prevent or mitigate potential accidents, and the proficiency of personnel using the software.

The development and maintenance of a collection, or “toolbox,” of multiple-site use, standard solution, Software Quality Assurance (SQA)-compliant safety software is one of the improvement actions identified in the DOE Implementation Plan (IP)[3] for resolving Recommendation issues. The DOE Safety Software Toolbox will contain a set of quality-assured, configuration-controlled, safety analysis codes, recognized for DOE-broad, safety basis applications. Currently, six widely applied safety analysis computer codes have been designated for toolbox consideration, including ALOHA, CFAST, EPIcode, GENII, MACCS2 and MELCOR. While the toolbox concept considerably reduces SQA burdens among DOE contractor users of these codes, many users of unique, single-purpose, or single-site software may have sufficient technical justification to continue use of their computer code of choice. Consequently, there is a significant need to understand the protocol used in the SQA evaluation of the multiple-site, toolbox candidate software. The steps outlined here provide a roadmap for an equivalency argument, i.e., based on a procedure and a set of implementing criteria for SQA evaluation based on the designated toolbox software process. Following the procedure and meeting the criteria will provide sufficient basis to deem single-site software as “toolbox-equivalent”. Alternatively, a safety analysis contractor may be able to demonstrate that their SQA program, procedures, and practices meet or exceed those described here. The single-site software satisfactorily evaluated in this manner is applicable to support 10 CFR 830

Documented Safety Analyses (DSAs)[4], and can support decision-making on the identification of control sets for nuclear facilities.

Methodology

Safety analysis software for the DOE "toolbox" was designated by DOE/EH in March 2003.[5] Candidate software for toolbox status, and its version and area of applicability are listed in Table 1.

Table 1. Software Designated for DOE Safety Analysis Toolbox

Code	Version or Revision	Area of Applicability
ALOHA	5.2.3	Chemical Release/Dispersion and Consequence
CFAST	3.1.6	Fire Analysis
EPIcode	6.0	Chemical Release/Dispersion and Consequence
GENII	1.485 and 2.0	Radiological Dispersion and Consequence
MACCS2	1.12	Radiological Dispersion and Consequence
MELCOR	1.8.5	Leak Path Factor

The Implementation Plan for Recommendation 2002-1 recognized that the designated toolbox software, while widely used in the DOE Complex for safety analysis applications, have uncertain SQA pedigree. The Implementation Plan contains commitment 4.2.1.2 to address this situation, that provides

- A plan for evaluating the SQA characteristics of the programs, procedures, and practices for the designated safety-related toolbox codes
- The requisite criteria for evaluating the SQA adequacy of the DOE toolbox safety analysis computer codes.

Each of these six codes and their respective development programs has undergone evaluation of their SQA attributes relative to established requirements identified through Task 4.2.1.2 and is termed a SQA evaluation. The SQA evaluation assessed those measures requiring action, i.e., areas of improvement, before the individual codes meet current SQA-compliant standards.

Software Quality Assurance Primary Criteria

An over-arching framework of primary criteria to conduct assessments was established early in the SQA evaluation program for the designated toolbox software. The same framework is readily applicable to single-site software and is discussed here to provide a basis for subsequent, working level procedures to be applied for evaluation of safety analysis codes.

The primary criteria are those in the Quality Assurance rule, Subpart A to 10 CFR 830.[6] Subpart A establishes quality assurance requirements for DOE contractors conducting activities including providing items or services, that affect, or may affect, the nuclear safety of DOE nuclear facilities. Section 830.121 describes a requisite quality assurance program (QAP) its applicability, frequency of updates, and directs the contractor to describe how criteria (Section

830.122) are met. It also specifies integration with the Safety Management System and recommends use of voluntary consensus standards.

Ten broad quality assurance criteria are described in Section 830.122. Each quality assurance criterion is stated as a performance expectation without specification of the methods for achieving the desired result. Instead, contractors are directed to national and international standards to develop effective and efficient QAPs. The management, performance, and assessment criteria include:

1 – Management Program
2 - Management/Personnel Training and Qualification
3 - Management/Quality Improvement
4 – Management/Documents and Records
5 – Performance/Work Processes
6 – Performance/Design
7 – Performance/Procurement
8 – Performance/Inspection and Acceptance Testing
9 – Assessment/Management Assessment
10 – Assessment/Independent Assessment.

The DOE implementation guide for quality assurance requirements from the 10 CFR 830 rule is DOE G 414.1-2. DOE G 414.1-2 includes a discussion of standards use, and references the most widely accepted standards for quality assurance.

NQA-1-2000, Part II, Section 2.7 and Other Applicable Parts

While several national and international sets of software quality assurance partially meet the needs of assuring software quality in the nuclear sector and provide guidance to following the Quality Assurance rule, it is concluded that the ASME NQA-1[7] requirements best address safety analysis software and cover the full spectrum of needs for this type of software. NQA-1 is referenced in 10 CFR 830 Subpart A, and it provides guidance for complying with Nuclear Safety requirements. It incorporates the basic criteria from 10 CFR 50, Appendix B, [8] 10 CFR 830 Subpart A and references key criteria from Institute of Electronics and Electrical Engineers (IEEE) standards. An organizational structure and assignment of responsibilities is formally prescribed in NQA-1 such that

- management establishes overall expectations for effective QA program implementation and is ultimately responsible for the end result;
- quality is achieved and maintained by those performing work; and
- quality is verified by those not directly responsible for performing the work.

In other words, there are clear, unambiguous roles delineated in NQA-1, and defined independence in performing various phases of work. The functional roles and independence characteristics are prerequisites to developing and maintaining a controlled approach to developing sound safety analysis software.

Both NQA-1a-1999 and NQA-1-2000 emphasize performance-based practices and graded application, yet reduce prescriptive requirements and redundancy.[9] Thus, the NQA-1 standard is intended by its authors to be applied in a graded approach manner. This intent of the NQA Committee is clear from the recommendation of judicious application of the entire standard or portions of the standard. The standard goes on to indicate

> The extent to which this Standard should be applied will depend upon the specific type of nuclear facility, items, or services involved and the nature and scope and the relative importance of the activities being performed. The extent of application is to be determined by the organization imposing the Standard.[10]

A major theme to changes in NQA-1 has been protecting the health and safety of the public while performing work that meets requirements. This goal is in line with nuclear safety directives and guidance from the Department of Energy, including DOE-STD-3009-94[11] and other "safe harbor" methodologies listed in Table 2 in Subpart B to 10 CFR 830. While many of the requirements from ISO 9001[12] (or ISO 9000-3) can be considered to complement NQA-1, the fundamental intent of ISO 9001 is as a quality management standard. Moreover, it is not specifically directed at the health and safety concerns.

Finally, it should be noted that many contractors have already based their respective site software quality assurance programs on some version of NQA-1. To shift to another system for benchmarking safety analysis codes would demand high resource commitments without a commensurate increase in the level of software quality achieved.

The core set of requirements for quality assurance of safety analysis software is contained in Subpart 2.7 of ASME NQA-1-2000. Subpart 2.7 provides requirements for the acquisition, development, operation, maintenance, and retirement of software. However, implementation of these requirements by a code developer should follow a prescriptive set of instructions. Subpart 2.7 notes that "The appropriate requirements of this Subpart shall be implemented through the policies, procedures, plans, specifications, or work practices, etc., that provide the framework for software engineering activities". Thus, it is expected that the safety analysis software owners/vendors have used a documented procedural basis to develop their respective software.

Four broad elements are included in the scope of software engineering activities described in Subpart 2.7:

(a) software acquisition methods for controlling the acquisition process for software and software services;
(b) software engineering method(s) used to manage the software life-cycle activities;
(c) application of standards, conventions, and other work practices that support the software life cycle;
(d) controls for support software used to develop, operate, and maintain computer programs.

Section 200 covers General Requirements, including Documentation, Review, Configuration Management, and Problem Reporting and Corrective Action. Section 300 outlines software requirements according to the type of acquisition. Section 400 contains requirements on documentation, and the planning and performance of software life cycle activities. Included are

Software Design Requirements, Software Design, Implementation, Acceptance Testing, Operation, Maintenance, and Retirement.

Other requirements of NQA-1, specifically sections from Part I, are referenced in the body of Subpart 2.7 or are described as recommended practices, and should be consulted as appropriate (Table 2). Part IV, Subpart 4.1 is an application guide with a discussion of the requirements and how those requirements may apply in various situations where software is used. The supporting sections to Subpart 2.7 are typically cited as “applicable parts”.

Table 2. Applicable Sections from NQA-1 Supporting Software Development and Maintenance

Part	Requirement	Section
I	2 – Quality Assurance Program	100 – Basic 200 – Indoctrination and Training
I	3 – Design Control	400 – Design Analysis 800 – Software Design Control
I	4 – Procurement Document Control	Applicable Requirements to Software
I	7 – Control of Purchased Items and Services	Applicable Requirements to Software
I	11 – Test Control	100 – Basic 200 – Test Requirements 400 – Computer Program Test Procedures 500 – Test Results 600 – Test Records
IV	4.1 Application Appendix – Guide on Quality Assurance Requirements for Software	100 – General 200 – General Requirements 300 - Software Acquisition 400 – Software Engineering Method 500 – Standards, Conventions, and Other Work Practices 600 – Support Software 601 – Software Tools 602 – System Software

In summary, 10 CFR 830 Subpart A, and the NQA-1-2000, Subpart 2.7 and related Part I requirements, primarily Requirements 3 (*Design Control*/Section 800 *Software Design Control*) and 11 (*Test Control*/Section 400 *Computer Program Test Procedures*), are recommended as the primary set of SQA criteria for the evaluation of safety-related computer software. This selection is based on

- Nuclear industry precedent with ASME NQA standards
- Federal and commercial sectors continued involvement with, and maintenance of the ASME NQA standards
- Quality assurance perspective through connection with 10 CFR 50 Appendix B and 10 CFR 70
- Independence of roles in developing and maintaining software, among management, work performers, and work reviewers

- Graded application based on safety, risk, and hazard analysis of the function of the software
- Focus on protection of the public and workers
- Long-standing presence and incorporation with many DOE contractors' quality assurance programs, with focus on nuclear safety, and
- Completeness and relevance to scientific, applied research, design, analysis and nuclear engineering software.

In 1999 and 2000 versions of the Standard, Subpart 2.7 has been updated in its bases from ANSI/IEEE 729, *Glossary of Software Engineering Terminology* and ANSI/IEEE 1012, *Software Verification and Validation Plans*, to IEEE Std. 7-4.3.2-1993, *IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations* and ANSI/IEEE Std. 610.12-1990, *Glossary of Software Engineering Terminology*. Consistent with the change in IEEE standards as a basis, the Subpart uses software "design verification and testing," rather than "verification and validation".

Classification and Selection of Implementing Procedures

As noted previously, requirements from NQA-1-2000 are not met directly, but require implementing procedures with sufficient detail to guide appropriate actions for each computer code. The implementing procedures for meeting NQA-1-level requirements from the Savannah River Site (SRS), Sandia National Laboratories (SNL), and the Yucca Mountain Project (YMP) were reviewed as part of the SQA Implementation Plan project for application in the evaluation methodology. While the final procedural basis discussed is a merged set composed of procedures from these sources, it is primarily based on procedures from SRS.

The primary SQA criteria, discussed in the following section, recommend that the level of SQA associated with a computer code be commensurate with the importance of the software application. Thus, the second determination to be made prior to formally beginning the evaluation is software classification. The classification of the software level for a specific computer code is a determination of the importance of the software and its intended use for a given application. The classification and the category of the software drive the requirements that must be satisfied, and is based on graded application considerations. The review of the three sites noted above illustrated similar classification systems. However, in all cases, the classification of safety analysis software was placed at the top in terms of meeting requirements. Therefore, it was recommended that either an A or B classification be used (based on the five-tier system applied at SRS).

In terms of application to safety analysis software, both level A and B classifications imply that the most stringent requirements must be met. However, Level A is reserved for software *whose output is used directly, with no additional evaluation or review prior to taking action*. This intent is in contrast to Level B software *whose output is used indirectly, i.e. it is subject to evaluation or review prior to taking action*. Recognizing that the designated toolbox software is used in applications where the output of the software is part of the evaluation in accident analysis, and is typically subject to thorough technical review, the most applicable classification for the designated safety analysis toolbox software is Level B, i.e.,

- *Software applications whose failure to properly function may have an indirect effect on nuclear safety protection systems or toxic materials hazard systems that are used to keep nuclear or toxic material hazard exposure to the general public and workers below regulatory or evaluation guidelines, or*
- *Software applications whose results are used to make decisions that could result in death or serious injury or are part of the evaluation in accident analyses.*

Table 3 lists the fourteen primary requirements from NQA-1 that were defined based on the implementing procedures reviewed from during the SQA IP project and are applicable safety analysis software. Safety analysis software classified as Level B software is evaluated based on the origin of the software, i.e., depending on whether *the software is under development, exists but did not follow NQA-1-2000 or similar primary criteria,* or *is being purchased*. Review of the six safety analysis codes designated for the toolbox as well as most single-site software suggests that the specific requirements listed in Table 3 under “Level B Existing” will be applicable.

The extent to which the Table 3 requirements are used depends not only on the origin of the software but also whether the software developer or the software implementing organization is evaluated. In the case of the developer, ten criteria are used, including; (1) software classification; (2) SQA procedures and plans; (5) requirements; (6) design; (7) implementation; (8) testing; (9) user instructions; (10) acceptance test; (11) configuration control; and (12) error impact. Requirements 3 (Dedication), 11 (Operation and Maintenance), and 14 (Access Control), are not applicable for the software development process, and thus are not evaluated in this review. Requirement 4 (Evaluation) is an outline of the minimum steps to be undertaken in a software review, and is complied with by performing the steps listed above during the full evaluation process. For an implementing organization, all fourteen criteria should be applied to assess the software. A final area of training is evaluated in either case.

The full implementing procedures for demonstrating compliance with the fourteen NQA-1-2000 fourteen requirements are included in the DOE document “SQA criteria (full title: Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes)”, maintained on the DOE SQA Knowledge Portal (website: http://www.eh.doe.gov/sqa/central_registry.htm). Table 3-3 in the document provides detailed criteria for each applicable requirement, and the matching ASME NQA-1-2000 section and consensus standard(s). Table 4 illustrates the level of information for two of the fourteen requirements, the Design and Implementation Phases (6 and 7).

Table 3. Software Requirements Matrix for Safety Analysis Applications

REQUIRE-MENT	NQA-1 Section& Related Standard	Computer Software Origin		
		Level B Development	**Level B Existing**	**Level B Purchased**
1. Software Classification	ASME NQA-1 2000 Section 200	Required*	Required	Required
2. SQA Procedures/Plans	ASME NQA-1 2000 Section 200; IEEE Std. 730, *IEEE Standard for Software Quality Assurance Plans*	Required	Required	Required
3. Dedication	ASME NQA-1 2000 Section 300; EPRI NP-5652	Graded**	Graded	Required
4. Evaluation	ASME NQA-1 2000 Section 302	Graded	Required	Graded
5. Requirements	ASME NQA-1 2000 Section 401; IEEE Standard 830, *Software Requirements Specifications*	Required	Required	Required
6. Design	ASME NQA-1 2000 Section 402; IEEE Standard 1016.1, *IEEE Guide for Software Design Descriptions;* IEEE Standard 1016-1998, *IEEE Recommended Practice for Software Design Descriptions*	Required	Required	Graded
7. Implementation	ASME NQA-1 2000 Section 204; IEEE Standard 1016.1, *IEEE Guide for Software Design Descriptions;* IEEE Standard 1016-1998, *IEEE Recommended Practice for Software Design Descriptions*	Required	Required	Graded
8. Testing	ASME NQA-1 2000 Section 404; IEEE Std. 829, *IEEE Standard for Software Test Documentation;* IEEE Standard 1008, *Software Unit Testing*	Required	Required	Required

Table 3. Software Requirements Matrix for Safety Analysis Application (continued)

9. User Instructions	ASME NQA-1 2000 Section 203; IEEE Standard 1063, *IEEE Standard for Software User Documentation*	Required	Required	Required
10. Acceptance Test	ASME NQA-1 2000 Section 404; IEEE Std. 829, *IEEE Standard for Software Test Documentation;* IEEE Standard 1008, *Software Unit Testing*	Required	Required	Required
11. Operation & Maintenance	ASME NQA-1 2000 Section 405; ASME NQA-1 2000 Section 406	Required	Required	Required
12. Configuration Control	ASME NQA-1 2000 Section 203	Required	Required	Required
13. Error Impact	ASME NQA-1 2000 Section 204; IEEE Standard 1063, *IEEE Standard for Software User Documentation*	Graded	Graded	Graded
14. Access Control	ASME NQA-1 2000 Section 405	Required	Required	Required

* Required for the computer software; ** Graded required depending on the application, and based on judgment of SQA Evaluator.

Table 4. Software Documentation Compliance Matrix for Safety Analysis Application – Design Requirement

REQUIREMENT		PROCEDURE	ASME NQA-1 2000 Section / Consensus Standards
6.	**Design Phase**	Verify a software design was developed, documented, and reviewed and controlled. The code developer should have prescribed and documented the design activities to the level of detail necessary to permit the design process to be carried out and to permit verification that the design met requirements. The following design elements should be present and documented: a. The design should specify the interfaces, overall structure (control and data flow) and the reduction of the overall structure into physical solutions (algorithms, equations, control logic, and data structures). b Computer programs were designed as an integral part of an overall system. Therefore, evidence should be present that the software design considered the computer program's operating environment. c. Evidence of measures to mitigate the consequences of problems should be an integral part of software design. These potential problems include external and internal abnormal conditions and events that can affect the computer program. A **Software Design Document**, or the equivalent, should be available, and should contain: - a description of the major components of the software design as they relate to the software requirements; - a technical description of the software with respect to the theoretical basis, mathematical model, control flow, data flow, control logic, data structure, numerical methods, physical models, process flow, process structures, and applicable relationship between data structure and process standards; - a description of the allowable or prescribed ranges for inputs and outputs; - the design described in a manner that can be translated into code; and - a description of the approach to be taken for intended test activities based on the requirements and design that specify the hardware and software configuration to be used during test execution. Review and approval: The organization responsible for the design should have identified and documented the particular verification methods to be used and assured that an Independent Review was performed and documented. This review should have evaluated the technical adequacy of the design approach; assure internal completeness, consistency, clarity, and correctness of the software design; and verified that the software design is traceable to the requirements. The organization responsible for the design should have assured that the test results adequately demonstrated the requirements were met.	ASME NQA-1 2000 Section 402; IEEE Standard 1016.1, *IEEE Guide for Software Design Descriptions;* IEEE Standard 1016-1998, *IEEE Recommended Practice for Software Design Descriptions*

Table 4. Software Documentation Compliance Matrix for Safety Analysis Application – Design and Implementation Requirements (Continued)

	REQUIREMENT	PROCEDURE	ASME NQA-1 2000 Section / Consensus Standards
6.	**Design Phase (continued)**	The Independent Review shall be performed by competent individual(s) other than those who developed and documented the original design, but who may be from the same organization. The results of the Independent Review shall be documented with the identification of the verifier indicated. When review alone is not adequate to determine if requirements are met, alternate calculations shall be used, or tests shall be developed and integrated into the appropriate activities of the software development cycle. Software design documentation shall be completed prior to finalizing the Independent Review. The extent of the IR and the methods chosen should be shown to be a function of: (a) the importance to safety, (b) the complexity of the software, (c) the degree of standardization, and (d) the similarity with previously proven software.	
7.	**Implementation Phase**	Verify that there is evidence of the implementation process resulting in software products such as computer program listings and instructions for computer program use. There should be evidence that implemented software was analyzed to identify and correct errors. The source code finalized at this time should have been placed under configuration control. Documentation for this phase shall include a copy of the software, test case description and associated criteria that are traceable to the software requirements and design documentation.	ASME NQA-1 2000 Section 204; IEEE Standard 1016.1, *IEEE Guide for Software Design Descriptions;* IEEE Standard 1016-1998, *IEEE Recommended Practice for Software Design Descriptions*

Plan for Safety Analysis Non-Toolbox Software

The suggested approach, or plan, for conducting a review of the Software Quality Assurance programs, practices, and procedures for a non-toolbox code and applying the requirements and criteria is described in this section. The plan was followed in the evaluation of the six designated toolbox codes during the SQA IP Project. In principal, the participants in the evaluation of the non-toolbox, single-site software include the following:

SQA Evaluator - an independent reviewer of the computer software, who is not affiliated with the code developing organization. It is required this individual

- knows the SQA requirements at the level of rigor for accident analysis applications
- understands and has applied the software in question, and
- is aware of the overall context for the use of the software as part of the DOE accident analysis process.

Software Developer/ – the originator of the software, who is responsible for documenting SQA protocols associated with the software, developing new versions of the subject software, addressing user questions, and resolving technical and programmatic issues,

(Optional: Software User - if the implementing organization is evaluated) – the contractor responsible for correctly implementing the software for safety analysis application, who is responsible for documenting SQA protocols associated with software use, ensuring that the software is applied in its valid domain, training users in concert with the software developer, and forwarding issues regarding software use to the developer.

The evaluation plan is followed based on submittal of required information from the software developer/user. The latter organization is requested to provide information on the programs and procedures associated with the development, maintenance, and use of their software. An input template for this purpose was used in the SQA IP Project, and is recommended as a mechanism to solicit information from the software developing/implementing organization. (An electronic copy may be obtained from the authors).

The input template seeks the following signed and approved information:

- Software Quality Assurance Plan
- Software Requirements
- Software Design
- Test Case Description and Outcome
- Software Configuration and Control
- Error Notification and Corrective Action Process
- User's Instructions, and other relevant instruction documentation (model description, weekly or monthly reports to code sponsor, etc.).

Files, reports, telephone conferences, and other documented communications can provide a confirmatory indication that actions have been performed in a SQA program, and these can be used in lieu of the availability of formal documents. However, formal documents explicitly demonstrate compliance with the primary criteria and are preferred. They are not sufficient in themselves but they tend to reduce uncertainty in verifying whether an action has been taken. The requirement and specific document for helping determine compliance are listed in Table 5.

Table 5. SQA Topical Area and Corresponding Documentation for Demonstrating Compliance

SQA Requirement	**Topical Area**	**SQA Document**
1	Software Classification	-
2	SQA Procedures/Plans	SQA Plan
5	Requirements Phase	Software Requirements Document
6	Design Phase	Software Design Document
7	Implementation Phase	5, 6, 8
8	Testing Phase	Test Case Description and Report
9	User Instructions	User's Manual
10	Acceptance Test	User Instructions
12	Configuration Control	Software Configuration and Control Document
13	Error Notification.	Error Notification and Corrective Action Report
	Training and Qualification of Users	Training Package and User Qualification

The SQA Evaluator will then perform and document a review of the software, using the inputs from the code developer, including documentation, resource estimates, and other communications. In cases where the software developer is unable to supply inputs to the SQA Evaluator, the gap analysis will proceed with alternative sources of information. Examples of alternative information are previous reviews, older documentation from the code developer, technical and journal articles, and previous software comparison studies.

Evaluation Process for Existing Software

As described in the earlier section, the SQA Evaluator obtains the appropriate input documentation from the code developer, or performs a review of the documentation from the code developer and/or other alternative sources. The input template questionnaire and other written and verbal communications can be used to expedite the evaluation.

Table 6 contains a plan for evaluating existing software, developed outside of the full requirements of the primary SQA criteria, and defined as Level B software for safety analysis applications. This procedure, to be followed by the SQA evaluator, provides instructions for the evaluation of existing safety analysis software, referencing the detailed procedures and criteria discussed earlier.

Phases 6 and 7 on Software Training and Engineering Planning (Upgrades), respectively, have been added to the overall evaluation process.

Table 6. – Plan for SQA Evaluation of Existing Safety Analysis Software

Phase	**Procedure**
1. Prerequisites	a. Determine that sufficient information is provided by the software developer to allow it to be properly classified for its intended end-use. b. Review SQAP per applicable requirements in Table 3-3.
2. Software Engineering Process Requirements	a. Review SQAP for: • Required activities, documents, and deliverables • Level and extent of reviews and approvals, including internal and independent review. Confirm that actions and deliverables (as specified in the SQAP) have been completed and are adequate. b. Review engineering documentation identified in the SQAP, e.g., • Software Requirements Document • Software Design Document • Test Case Description and Report • Software Configuration and Control Document • Error Notification and Corrective Action Report, and • User's Instructions (alternatively, a User's Manual), Model Description (if this information has not already been covered). c. Identify documents that are acceptable from SQA perspective. Note inadequate documents as appropriate.
3. Software Product Technical/ Functional Requirements	a. Review requirements documentation to determine if requirements support intended use in Safety Analysis. Document this determination in gap analysis document. b. Review previously conducted software testing to verify that it sufficiently demonstrated software performance required by the Software Requirements Document. Document this determination in the gap analysis document.
4. Testing	a. Determine whether past software testing for the software being evaluated provides adequate assurance that software product/technical requirements have been met. Obtain documentation of this determination. Document this determination in the gap analysis report. b. (Optional) Recommend test plans/cases/acceptance criteria as needed per the SQAP if testing not performed or incomplete.
5. New Software Baseline	a. Recommend remedial actions for upgrading software documents that constitute baseline for software. Recommendations can include complete revision or providing new documentation. A complete list of baseline documents includes: • Software Quality Assurance Plan • Software Requirements Document • Software Design Document • Test Case Description and Report • Software Configuration and Control • Error Notification and Corrective Action Report, and • User's Instructions (alternatively, a User's Manual) b. Provide recommendation for central registry as to minimum set of SQA documents to constitute new baseline per the SQAP.

Table 6. – Plan for SQA Evaluation of Existing Safety Analysis Software (continued)

Phase	Procedure
6. Training	a. Identify current training programs provided by developer. b. Determine applicability of training for DOE facility safety analysis.
7. Software Engineering Planning	a. Identify planned improvements of software to comply with SQA requirements. b. Determine software modifications planned by developer. c. Provide recommendations from user community. d. Estimate resources required to upgrade software.
8. Document Evaluation	Use one of six gap analysis reports as reference to document evaluation.

The actual SQA evaluation may be performed by one, or a team of independent personnel, experienced in use of the software but also knowledgeable of the evaluation criteria. It is recommended that the evaluation of the eleven topical areas covered in Table 5 use a sub-matrix of finer criteria to adequately evaluate the constituent parts of the requirement. Qualitative ranking of compliance was used with the designated toolbox codes; the four terms applied were yes, no, uncertain, and partial. Upon completion of evaluation of each of the eleven areas, the SQA evaluator can review results as a whole and render an overall assessment. The process leads to a firm basis to document findings in a verifiable, objective manner.

The gap analysis reports performed on the six designated toolbox codes are a reasonable level of detail for SQA evaluation documentation. Copies can be reviewed at the Central Registry website (http://www.eh.doe.gov/sqa/central_registry.htm).

Application

The evaluation of a given computer code using the recommended methodology will expedite evaluation of eleven topical areas, based on 10 CFR 830 Subpart A and NQA-1, including ten SQA requirements and the added area of training and user qualification. However, the actual schedule and level of effort will depend on the status of the SQA materials and the skill level of the SQA Evaluator or team.

The MACCS2 SQA evaluation required approximately three full-time equivalent months and lasted over eight months. Much of this time was spent in setting up the criteria, developing an evaluation plan, and document transmittal delays. It is estimated that for a given single-site, non-toolbox code will require one to three FTE-months. The actual schedule and level of effort will be dictated by the availability of SQA information and documentation, knowledge of software developer/implementer of the program, practices and procedures used to establish the version of the software under evaluation.

Table 7 provides a summary of the areas of improvement identified using this process evaluating Version 1.12 of the MACCS2 code. Of the ten topical areas, two were found to be satisfactory, one was partially met and seven provided an opportunity for improvement. The training and user qualification area for MACCS2 and other safety software is the joint responsibility of both the

Table 7. — Summary of Important Exceptions, Reasoning, and Suggested Remediation

No.	Criterion	Reason Not Met	Remedial Action(s)
1.	SQA Procedures/Plans (Section 4.2)	Earlier versions of MACCS (version 1.5.11.1 and older) followed SNL software engineering guidance. Although initially followed, SNL SQA Plan and Procedures for Version 1.12 of MACCS2 software were not explicitly followed.	As part of the new software baseline, the SQA Plan covering version 1.12 and successor versions of MACCS2 should be addressed as a stand-alone report or as part of another SQA document. Any new SQA procedures that provide prescriptive guidance to the MACCS2 software developers should be made available to a SQA evaluator for confirmatory review.
			• Document a written and approved SQA plan eliminating draft or non-compliant informal process of development.
			• Upgrade SQA program documentation, especially those procedures used for new features added in MACCS2.
2.	Requirements Phase (Section 4.3)	The Software Requirements documents for Version 1.12 of MACCS2 software, although filed for a 3 – 4 year period, were not maintained. Consequently the Software Requirements Document was never completed.	As part of the new software baseline for MACCS2, a concise listing of the software requirements should be documented. This can be reported as a stand-alone Software Requirements report, or as part of another MACCS2-specific document. Specific MACCS2 requirements need to be documented. Those from MACCS may be added to supplement the MACCS2 information, but are not as critical. In contrast, some MACCS-attributes are no longer present in the code, and it would facilitate understanding of the current code requirements to know which ones have been deleted.
3.	Design Phase (Section 4.4)	A Software Design Document was not made available for the gap analysis. Thus, design information was not directly available. Instead, it was necessary to infer the intent of MACCS2 design from incomplete model description and user guidance documents, some of which address MACCS, not MACCS2.	As part of the new software baseline for MACCS2, software design information should be provided. This can be reported as a stand-alone report, or as part of another MACCS2-specific document, such as the model description. .
4.	Implementation Phase (Section 4.5)	Written documentation on implementation of Version 1.12 was not produced for MACCS2.	No action needed at this time. The gap analysis inferred from other documentation that source code and other software elements were finalized prior to transmittal of the code to RSICC.
5.	Testing Phase (Section 4.6)	A Software Testing Report Document has not been produced for MACCS2, and therefore, test process and methodology could not be evaluated directly. Thus, testing process and methods had to be inferred from other information. A draft validation study has never been published.	A test document was prepared by the University of New Mexico (Summa, 1996), but never approved. As part of the new software baseline for MACCS2, this report should be finalized.

No.	Criterion	Reason Not Met	Remedial Action(s)
6.	Acceptance Test (Section 4.8)	An Acceptance Test protocol was not provided to the gap analysis. No documentation exists that indicates how the code developers tested the code. There is no known formal procedure to assure that an installed version of MACCS2 is working properly.	As part of the new software baseline for MACCS2, an acceptance test process should be documented. This instruction can be made part of an upgraded User's Guide, and proceduralized in the installation files provided by RSICC or SNL.
7.	Configuration Control (Section 4.9)	A MACCS2 Configuration and Control document was not provided for the gap analysis, despite indication that a configuration control system was in place for MACCS2. Files to support this area were not maintained.	It is recommended that a full-scope Software Configuration and Control document be issued as part of the new software baseline. If this document has been generated, then it should be made available for review.
8.	Error Notification (Section 4.10)	An Error Notification and Corrective Action Report process is in place at SNL, but limited documentation was provided.	While a Software Problem Reporting system is apparently functional at SNL, written documentation should be provided to demonstrate its effectiveness.

developer and the implementing organization. Nonetheless, for this eleventh area, the SQA evaluator recommended improvements to meet the established criteria.

While the number of improvement actions in MACCS2 and the other designated toolbox software are many, the use of this software in DSA applications is still warranted. This is because: (1) MACCS2 is used as a tool, and as such its output is applied in safety analysis only after appropriate technical review; (2) User-specified inputs are chosen at a reasonably conservative level of confidence; and (3) Use of MACCS2 is limited to those analytic applications for which the software is intended. A similar situation is probable for single-use software.

A final point should be underscored, that of independence of the SQA Evaluator. The evaluation process provides the most insight and value to the DSA process if it is conducted by personnel that are knowledgeable in the type of analysis area that the software supports, and aware of the software standards that should be met. It is understood that there are varying degrees on for understanding the analytical area and software requirements. However, there can be no compromise in the independence of the SQA evaluator(s) from the code developer and code user organizations.

Path Forward

Once the SQA evaluation has been conducted and documented, the software implementer and the code developer must prioritize the improvement actions before the software can be claimed as "Toolbox-Equivalent". The organizations involved must evaluate each situation before deciding if the subject software can be used in a DSA-support activity.

It is likely that one of three situations will be identified through the independent SQA evaluation. One possible outcome is that the software is found to be wholly compliant with SQA criteria and satisfactorily meets each of the eleven requirements listed in Table 5. It is then deemed “toolbox-equivalent” for DSA support tasks as applicable.

A far more likely case is that most software will be found deficient in two or more of the eleven SQA requirement areas. With this outcome, compensatory steps should be taken if the safety contractor concludes that the single-site software still merits use. A code guidance report may be generated for those contractors as one of the conditions prior to use. With respect to the designated toolbox safety analysis software, code-specific guidance reports have been written to inform users on the appropriate applications of the software, suggested input ranges for DSA application, sample cases that can be tailored for cases of interest, and avoidance of known software errors. The code guidance reports, also posted on the central registry website, are a commonsense approach to use of the designated toolbox codes, given that SQA recommendations may not be implemented in the near-term. Once the software has been satisfactorily modified, the code guidance report may no longer be useful, and withdrawn. Note that compensatory measures should only be performed if no evidence has been found of programming, logic, or other types of software errors in the code under evaluation that have led to non-conservatisms in nuclear facility operations, or in the identification of facility controls.

The other potential outcome is a SQA evaluation that leads to the conclusion that the software is not technically defensible for the intended DSA application. In this case, the safety contractor may then make an informed decision to discontinue use of the single-site software and select an alternative, possibly one of the multiple-use, designated toolbox codes.

With any of the three outcomes, the technical basis for use of specific software in the DSA is improved.

Conclusions

A limited-duration, limited-resource methodology to provide a technical basis for use of non-toolbox software to support DSAs, has been introduced. The methodology includes both a set of SQA requirements and evaluation criteria, and a plan for implementation. The approach is based on work performed under the DOE SQA Implementation Plan to address software improvements for six designated toolbox codes, and is compliant with requirements identified in 10 CFR 830 Subpart A, applicable parts of NQA-1, and web-based U.S. Department of Energy, *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes*.

The methodology provides a systematic, standardized strategy for dealing with safety analysis software that is not included in the current set of codes designated for the DOE Safety Software Toolbox. It guides the safety contractor in determining whether the single-use software is:

- Fully compliant with the SQA criteria and directly applicable supporting DSAs
- Partially compliant with the SQA criteria, but generally applicable to DSAs – code guidance is strongly recommended in this situation
- Not cost-effective to remediate, and therefore establishes a basis for possible use of alternatives, including the designated safety analysis software.

Furthermore, the discussion here provides a clear roadmap for critical examination of other potential software for use in supporting DSAs, rather than blindly applying the designated toolbox software. The following benefits are achieved:

- Improved awareness of the strengths, limits, and vulnerable areas of non-toolbox software prior to use in safety support work
- Information on areas where software quality assurance improvements are needed to comply with industry SQA standards and practices
- Basis for schedule and prioritization to upgrade software
- Templates for documentation of SQA review and code guidance reports
- Specific areas for improvement in terms of new versions of the software.

References

1. Defense Nuclear Facilities Safety Board, *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, Technical Report DNFSB/TECH-25, (January 2000).
2. Defense Nuclear Facilities Safety Board, *Recommendation 2002-1, Quality Assurance for Safety-Related Software*, (September 2002).
3. U.S. Department of Energy, *Implementation Plan for Defense Nuclear Facilities Safety Board Recommendation 2002-1: Quality Assurance for Safety Software at Department of Energy Nuclear Facilities*, Report, (March 13, 2003).
4. Code of Federal Regulations (CFR). 10 CFR 830, Nuclear Safety Management Rule.
5. U.S. Department of Energy Office of Environment, Safety and Health, *Designation of Initial Safety Analysis Toolbox Codes*, Letter, (March 28, 2003).
6. Code of Federal Regulations (CFR). 10 CFR 830, Nuclear Safety Management Rule, Subpart A.
7. ASME NQA-1a-1999, Addenda to ASME NQA-1-1997 Edition, Quality Assurance Requirements for Nuclear Facility Applications; ASME NQA-1-2000, Quality Assurance Requirements for Nuclear Facility Applications.
8. 10 CFR 50, Appendix B, *Quality Assurance Criteria for Nuclear Power Plants and Fuel Reprocessing Plants.*
9. American Society of Mechanical Engineers, *Re: Comments on the Benefits of National Nuclear Quality Assurance Standards for NNSA and DOE Nuclear Activities and Oversight*, Letter to Linton F. Brooks, NNSA (2002).
10. American Society of Mechanical Engineers NQA-1-2000, Foreword to Quality Assurance Requirements for Nuclear Facility Applications (2000).
11. U.S. Department of Energy, *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports*, DOE-HDBK-3010-94, Change Notice 2 (April 2002).
12. International Organization for Standardizations, ISO 9001-1994, *Quality systems -- Model for quality assurance in design, development, production, installation and servicing*; ISO 9001-2000, Quality management systems – Requirements; ISO 9000-3, *ISO Quality management and quality assurance standards - Part 3: Guidelines for the application of ISO 9001:1994 to the development, supply, installation and maintenance of computer software*.