Department of Energy
Washington, DC 20585

May 12, 2004

The Honorable John T. Conway
Chairman
Defense Nuclear Facilities Safety Board
625 Indiana Avenue, NW
Washington, D.C. 20004-2941

Dear Mr. Chairman:

In January 2004, the Department reported interim completion of Commitment 4.2.1.3 in the Department's Implementation Plan for Software Quality Assurance (SQA) in response to Board Recommendation 2002-1. This commitment required the Department to conduct a gap analysis to determine the actions needed to bring the six toolbox codes into compliance with SQA criteria and to develop a schedule with milestones to upgrade each code based on the gap analysis results.

The Department has completed additional code developer and peer reviews to further improve the accuracy of the gap analysis effort as described in the Secretary's letter to you on May 3, 2004. The attached final gap analyses reports for the MACCS2, ALOHA, EPICODE, MELCOR, GENII, and CFAST toolbox codes are complete. While SQA improvement actions are recommended for the six toolbox codes, no evidence has been found of programming, logic, or other types of software errors that would have led to non-conservatisms at defense nuclear facilities. In parallel with completion of these reports, we have contacted each of the code developers and will be evaluating further improvement actions, as described in the Secretary's May 3, 2004 letter.

The Guidance Reports for each of these codes are being finalized by the end of the month and provide information to users on the effective use of these codes. Questions concerning this commitment may be directed to Chip Lagdon at (301) 903-4218 or me at (301) 903-8008.

Sincerely,

Sharon A. Root for

Frank B. Russo
Deputy Assistant Secretary
for Corporate Performance Assessment

Attachments

cc:
Mark B. Whitaker, DR-I
Richard H. Lagdon, EH-31

SEPARATION

PAGE

Defense Nuclear Facilities Safety Board Recommendation 2002-1
Software Quality Assurance Improvement Plan
Commitment 4.2.1.3:

Software Quality Assurance Improvement Plan: MELCOR Gap Analysis

Final Report

U.S. Department of Energy
Office of Environment, Safety and Health
1000 Independence Ave., S.W.
Washington, DC 20585-2040

May 2004

INTENTIONALLY BLANK

FOREWORD

This report documents the outcome of an evaluation of the Software Quality Assurance (SQA) attributes of the MELCOR computer code for leak path factor applications, relative to established software requirements. This evaluation, a "gap analysis," is performed to meet Commitment 4.2.1.3 of the Department of Energy's Implementation Plan to resolve SQA issues identified in the Defense Nuclear Facilities Safety Board Recommendation 2002-1.

Suggestions for corrections or improvements to this document should be addressed to –

Chip Lagdon
EH-31/GTN
U.S. Department of Energy
Washington, D.C. 20585-2040
Phone (301) 903-4218
Email: chip.lagdon@eh.doe.gov

INTENTIONALLY BLANK

REVISION STATUS

Page/Section	Revision	Change
1. Entire Document	1. Interim Report	1. Original Issue
2. Entire Document	2. Final Report, May 3, 2004	2. Updated all sections per review comments.

INTENTIONALLY BLANK

CONTENTS

Section			Page
FOREWORD			**iii**
REVISION STATUS			**v**
EXECUTIVE SUMMARY			**xiii**
1.0	**Introduction**		**1-1**
	1.1	BACKGROUND: OVERVIEW OF DESIGNATED TOOLBOX SOFTWARE IN THE CONTEXT OF 10 CFR 830	1-1
	1.2	EVALUATION OF TOOLBOX CODES	1-2
	1.3	USES OF THE GAP ANALYSIS	1-2
	1.4	SCOPE	1-2
	1.5	PURPOSE	1-3
	1.6	METHODOLOGY FOR GAP ANALYSIS	1-3
	1.7	SUMMARY DESCRIPTION OF SOFTWARE BEING REVIEWED	1-5
2.0	**Assessment Summary Results**		**2-1**
	2.1	CRITERIA MET	2-1
	2.2	EXCEPTIONS TO REQUIREMENTS	2-1
	2.3	AREAS NEEDING IMPROVEMENT	2-2
	2.4	CONCLUSION REGARDING SOFTWARE'S ABILITY TO MEET INTENDED FUNCTION	2-3
3.0	**Lessons Learned**		**3-3**
4.0	**Detailed Results of the Assessment Process**		**4-1**
	4.1	TOPICAL AREA 1 ASSESSMENT: SOFTWARE CLASSIFICATION	4-2
		4.1.1 Criterion Specification and Result	*4-2*
		4.1.2 Sources and Method of Review	*4-3*
		4.1.3 Software Quality-Related Issues or Concerns	*4-3*
		4.1.4 Recommendations	*4-3*
	4.2	TOPICAL AREA 2 ASSESSMENT: SQA PROCEDURES AND PLANS	4-3
		4.2.1 Criterion Specification and Result	*4-5*
		4.2.2 Sources and Method of Review	*4-6*
		4.2.3 Software Quality-Related Issues or Concerns	*4-7*
		4.2.4 Recommendations	*4-7*
	4.3	TOPICAL AREA 3 ASSESSMENT: REQUIREMENTS PHASE	4-7
		4.3.1 Criterion Specification and Results	*4-8*
		4.3.2 Sources and Method of Review	*4-8*
		4.3.3 Software Quality-Related Issues or Concerns	*4-8*
		4.3.4 Recommendations	*4-9*
	4.4	TOPICAL AREA 4 ASSESSMENT: DESIGN PHASE	4-9
		4.4.1 Criterion Specification and Result	*4-9*

4.4.2 Sources and Method of Review *4-12*
4.4.3 Software Quality-Related Issues or Concerns *4-13*
4.4.4 Recommendations *4-13*
4.5 TOPICAL AREA 5 ASSESSMENT: IMPLEMENTATION PHASE 4-13
4.5.1 Criterion Specification and Result *4-13*
4.5.2 Sources and Method of Review *4-14*
4.5.3 Software Quality-Related Issues or Concerns *4-14*
4.5.4 Recommendations *4-14*
4.6 TOPICAL AREA 6 ASSESSMENT: TESTING PHASE 4-14
4.6.1 Criterion Specification and Result *4-14*
4.6.2 Sources and Method of Review *4-16*
4.6.3 Software Quality-Related Issues or Concerns *4-16*
4.6.4 Recommendations *4-16*
4.7 TOPICAL AREA 7 ASSESSMENT: USER INSTRUCTIONS 4-16
4.7.1 Criterion Specification and Result *4-17*
4.7.2 Sources and Method of Review *4-17*
4.7.3 Software Quality-Related Issues or Concerns *4-17*
4.7.4 Recommendations *4-18*
4.8 TOPICAL AREA 8 ASSESSMENT: ACCEPTANCE TEST 4-18
4.8.1 Criterion Specification and Result *4-18*
4.8.2 Sources and Method of Review *4-19*
4.8.3 Software Quality-Related Issues or Concerns *4-19*
4.8.4 Recommendations *4-19*
4.9 TOPICAL AREA 9 ASSESSMENT: CONFIGURATION CONTROL 4-19
4.9.1 Criterion Specification and Result *4-19*
4.9.2 Sources and Method of Review *4-20*
4.9.3 Software Quality-Related Issues or Concerns *4-20*
4.9.4 Recommendations *4-20*
4.10 TOPICAL AREA 10 ASSESSMENT: ERROR IMPACT 4-20
4.10.1 Criterion Specification and Result *4-20*
4.10.2 Sources and Method of Review *4-22*
4.10.3 Software Quality-Related Issues or Concerns *4-22*
4.10.4 Recommendations *4-22*
4.11 TRAINING PROGRAM ASSESSMENT 4-23
4.12 SOFTWARE IMPROVEMENTS AND NEW BASELINE 4-23

5.0 Conclusions **5-1**

6.0 Acronyms and Definitions **6-1**

7.0 References **7-5**

APPENDIX A. SOFTWARE INFORMATION TEMPLATE A-1

TABLES

Page

Table 1-1 — Plan for SQA Evaluation of Existing Safety Analysis Software 1-4

Table 1-2 — Summary Description of the MELCOR Software in the Context of LPF Analysis 1-7

Table 1-3 — Software Documentation Reviewed for MELCOR (LPF Applications) 1-10

Table 2-1 — Summary of Important Exceptions, Reasoning, and Suggested Remediation 2-1

Table 2-2 — Summary of Important Recommendations for MELCOR for LPF Applications 2-2

Table 3-1 — Lessons Learned 3-3

Table 4.0-1 — Cross-Reference of Requirements with Subsection and Entry from DOE (2003e) 4-1

Table 4.1-1 — Subset of Criteria for Software Classification Topic and Results 4-3

Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results 4-6

Table 4.3-1 — Subset of Criteria for Requirements Phase Topic and Results 4-8

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results 4-9

Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results 4-13

Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results 4-14

Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results 4-17

Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results 4-18

Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results 4-19

Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results 4-21

Table 4.12-1 — Comparison of SQA Upgrade Steps Discussed in Bixler (2000) with the Approach Discussed in DOE (2003e) 4-24

INTENTIONALLY BLANK

FIGURES

Page
Figure 1-1 MELCOR Execution Flowchart.....1-6

INTENTIONALLY BLANK

Software Quality Assurance Improvement Plan: MELCOR Gap Analysis

EXECUTIVE SUMMARY

The Defense Nuclear Facilities Safety Board (DNFSB) issued Recommendation 2002-1 on *Quality Assurance for Safety-Related Software* in September 2002 (DNFSB 2002). The Recommendation identified a number of quality assurance issues for software used 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 high-use, Software Quality Assurance (SQA)-compliant safety analysis codes is one of the major improvement actions discussed in the *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities*. A DOE safety analysis toolbox would contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed and maintained for DOE-broad safety basis applications.

The Methods for Estimation of Leakages and Consequences of Releases (MELCOR) software is one of the codes designated for the toolbox. It is being evaluated for leak path factor (LPF) applications. To determine the actions needed to bring the MELCOR code into compliance with the SQA qualification criteria in the context of LPF applications and develop an estimate of the resources required to perform the upgrade, the Implementation Plan has committed to sponsoring a code-specific gap analysis document. The gap analysis evaluates the software quality assurance attributes of MELCOR against identified criteria.

The balance of this document provides the outcome of the gap analysis compliant with NQA-1-based requirements as contained in U.S. Department of Energy, *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes*, (DOE, 2003e). It was determined that MELCOR code does meet its intended function for use in supporting documented safety analysis. However, as with all safety-related software, users should be aware of current limitations and capabilities of MELCOR for supporting safety analysis. Informed use of the software can be assisted by the current set of MELCOR reports (refer to Table 1-3), and the code guidance report for DOE safety analysts, *MELCOR Computer Code Application Guidance for Leak Path Factor in Documented Safety Analysis*, (DOE, 2004). Furthermore, while SQA improvement actions are recommended for MELCOR, no evidence has been found of programming, logic, or other types of software errors in MELCOR that have led to non-conservatisms in nuclear facility operations, or in the identification of facility controls.

Of the ten primary SQA requirements for existing software at the Level B classification ("important for safety analysis but whose output is not applied without further review"), five requirements are met at acceptable level, i.e., *Software Classification, Implementation Phase, User Instructions, Acceptance Test,* and *Configuration Control*; Requirements 1, 5, 7, 8, and 9 respectively. Improvement actions are recommended to meet SQA criteria for the remaining five requirements. This evaluation outcome is deemed acceptable because: (1) MELCOR 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 MELCOR is limited to those analytic applications for which the software is intended.

By order of priority, it is recommended that MELCOR software improvement actions be taken, especially:

1. Correcting known defects in the SQA process
2. Upgrading existing SQA documentation
3. Providing training on a regular basis, and
4. Revising and developing new software documentation.

A new software baseline set of documents is recommended for MELCOR to demonstrate completion of the revision to software documentation item (above). The list of revised baseline documents includes:

- Updated Software Quality Assurance Plan
- Software Requirements Document (Specific to LPF)
- Software Design Document (Specific to LPF)
- Test Case Description and Report (Specific to LPF)
- Updated Software Configuration and Control
- Updated Error Notification and Corrective Action Report Procedure, and
- Updated User's Manual.

Approximately two full-time equivalent years is conservatively estimated to upgrade MELCOR software to be compliant with NQA-1-based requirements for existing software. While most of this effort is logically to be used by the code developer, independent review of the end products is necessary.

A new version of MELCOR is planned for release in the future. It is recommended that this version be evaluated upon issue relative to the software improvement and baseline recommendations, as well as the full set of SQA criteria discussed in this report. If this version is found to be satisfactory, it should replace Version 1.8.5 as the designated version of the software for the toolbox.

Approximately one FTE-month per year would be needed to maintain a web-based error notification and corrective action process for MELCOR (Section 4.10). However, such a process has not been defined in depth for MELCOR and the other designated toolbox codes.

1.0 Introduction

This document reports the results of a gap analysis for Version 1.8.5 of the MELCOR computer code in the context of LPF applications. The intent of the gap analysis is to determine the actions needed to bring the specific software into compliance with established SQA criteria. A secondary aspect of this report is to develop an estimate of the level of effort required to upgrade each code based on the gap analysis results.

1.1 Background: Overview of Designated Toolbox Software in the Context of 10 CFR 830

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* (DNFSB, 2000). 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.

While progress was made in resolving several of the issues raised in TECH-25, the DNFSB issued Recommendation 2002-1 on *Quality Assurance for Safety-Related Software* in September 2002. 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, the quality of the software used to design or develop safety-related controls, and the proficiency of personnel using the software. The Recommendation identified a number of quality assurance issues for software used in the 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 high-use, SQA-compliant safety analysis codes is one of the major commitments contained in the March, 2003 *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities* (IP). In time, the DOE safety analysis toolbox will contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed and maintained for DOE-broad safety basis applications.

Six computer codes, including ALOHA (chemical release dispersion/consequence analysis), CFAST (fire analysis), EPIcode (chemical release dispersion/consequence analysis), GENII (radiological dispersion/consequence analysis), MACCS2 (radiological dispersion/consequence analysis), and MELCOR (leak path factor analysis) have been designated by DOE for the toolbox (DOE/EH, 2003). It is found that this software provides generally recognized and acceptable approaches for modeling source term and consequence phenomenology, and can be applied as appropriate to support accident analysis in Documented Safety Analyses (DSAs).

As one of the designated toolbox codes, MELCOR Version 1.8.5 will likely require some degree of quality assurance improvement before meeting current SQA standards. The analysis documented herein is an evaluation of MELCOR, in the context of LPF applications, relative to current software quality assurance criteria. It assesses the extent of the deficiencies, or gaps, to provide DOE and the software developer the extent to which minimum upgrades are needed. The overall assessment is therefore termed a "gap" analysis.

1.2 Evaluation of Toolbox Codes

The quality assurance criteria identified in later sections of this report are defined as the set of established requirements, or bases, by which to evaluate each designated toolbox code. This gap analysis evaluation is Commitment 4.2.1.3 in the IP:

> Perform a SQA analysis to the "toolbox" codes to determine the actions needed to bring the codes into compliance with the SQA qualification criteria, and develop a schedule with milestones to upgrade each code based on the SQA evaluation results.

This process is a prerequisite step for software improvement. It will allow DOE to determine the current limitations and vulnerabilities of each code as well as help define and prioritize the steps required for improvement.

Early in the SQA evaluation program, it was anticipated that each toolbox code owner would provide input information on the SQA programs, processes, and procedures used to develop their software. However, most of the designated toolbox software, including MELCOR, was developed without complete conformance to software quality standards. Furthermore, many of the software developer organizations cannot confirm that key processes were followed. Therefore, most of the SQA evaluation has been preceded with reconstructing software development processes based on anecdotal evidence and limited, supporting documentation.

For independence reasons, the gap analysis is performed by a SQA evaluator, not affiliated with the MELCOR development program. While independent of the code developer, the SQA evaluators responsible for MELCOR are knowledgeable in the use of the software for accident analysis applications, and understand current software development standards.

1.3 Uses of the Gap Analysis

The gap analysis provides key information to DOE, code developers, and code users.

DOE obtains the following benefits:

- Estimates of the resources required to perform modifications to designated toolbox codes
- Basis for schedule and prioritization to upgrade each designated toolbox code.

Each code developer is provided:

- Information on areas where software quality assurance improvements are needed to comply with industry SQA standards and practices
- Specific areas for improvement to guide development of new versions of the software.

DOE safety analysts and code users benefit from:

- Improved awareness of the strengths, limits, and vulnerable areas of each computer code
- Recommendations for code use in safety analysis application areas.

1.4 Scope

This gap analysis is applicable to the MELCOR code, one of the six designated toolbox codes for safety analysis, for applications of LPF analysis. While the MELCOR code is the subject of the current report,

other safety analysis software considered for the toolbox in the future may be evaluated with the same process applied here. The template outlined in this document is applicable for any analytical software as long as the primary criteria are ASME NQA-1, 10 CFR 830, and related DOE directives discussed in DOE (2003e).

1.5 Purpose

The purpose of this report is to document the gap analysis performed on the MELCOR code for LPF applications as part of DOE's implementation plan on SQA improvements.

1.6 Methodology for Gap Analysis

The gap analysis for MELCOR (LPF applications) is based on the plan and criteria described in *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes* (DOE 2003e). The overall methodology for the gap analysis is summarized in Table 1-1. The gap analysis utilizes ten of the fourteen topical areas listed in DOE (2003e) related to SQA to assess the quality of the MELCOR code in the context of LPF applications. The ten areas are those particularly applicable to the software development, specifically: (1) Software Classification, (2) SQA Procedures/Plans, (5) Requirements Phase, (6) Design Phase, (7) Implementation Phase, (8) Testing Phase, (9) User Instructions, (10) Acceptance Test, (12) Configuration Control, and (13) Error Impact. Each area, or requirement, is assessed individually in Section 4.

Requirements 3 (Dedication), 4 (Evaluation), 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 evaluating the areas listed above. Requirement 11 (Operation and Maintenance) is only partially applicable to software development, and is interpreted to be applicable mostly to the software user organization.

An information template was transmitted to the Safety Analysis Software Developers on 20 October 2003 to provide basic information as input to the gap analysis process. The main section of the template is attached as Appendix A to the present report, with an example section and references removed. No written response to the information template has been provided by the MELCOR software developers. Instead, SNL personnel were interviewed in January 2004 to obtain needed information to perform this analysis. The information in Appendix A was used as a guide during this interview, and the results are captured in the details of this report, Section 4.0.

Table 1-1 — Plan for SQA Evaluation of Existing Safety Analysis Software[1]

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 of DOE (2003e).
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 Procedure, 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: • SQA Plan • Software Requirements Document • Software Design Document • Test Case Description and Report • Software Configuration and Control • Error Notification and Corrective Action Procedure, 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.

[1] Originally documented as Table 2-2 in DOE (2003e).

Table 1-1 – 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.

1.7 Summary Description of Software Being Reviewed

The gap analysis was performed on Version 1.8.5 of the MELCOR code in the context of LPF applications. MELCOR (Gauntt, 2000a) is a generalized mass transport and thermal hydraulic computer program. MELCOR is available for the UNIX workstation platform as well as the PC platform. MELCOR 1.8.5 is the latest released version of MELCOR as of the beginning of this assessment. A patch was released 10/23/2001 (see the SNL MELCOR site http://melcor.sandia.gov/.) MELCOR Version 1.8.5 includes NRC and DOE sponsored changes made over the years.

MELCOR is a fully integrated, engineering-level computer code whose primary purpose is to model the progression of accidents in light water reactor nuclear power plants. A broad spectrum of severe accident phenomena in both boiling and pressurized water reactors is treated in MELCOR in a unified framework. MELCOR estimates fission product source terms and their sensitivities and uncertainties in a variety of applications. The MELCOR code is composed of a number of major modules, or packages, that together model the major systems of a reactor plant and its generally coupled interactions.

MELCOR was initially developed at the Sandia National Laboratories (SNL) under the sponsorship of the USNRC to assess reactor severe accident conditions. MELCOR was developed as a "research" code by the NRC and SNL. It was intended to be used to perform parametric studies, scoping studies, and studies to check the results of other models. For the last several years, MELCOR has been used in the DOE Complex to model release of radioactive airborne material from nuclear facilities and structures. The amount released is termed leakage and is usually expressed as a fraction of the amount considered available for release. This fraction released is referred to as the Leak Path Factor, LPF.

Although the MELCOR computer code was developed to model the progression of accidents in light water reactor nuclear power plants, the modeling capabilities of MELCOR are sufficiently flexible that it can be applied to the analysis of nonreactor problems. When performing LPF studies for nuclear facilities the modules used are reduced (through input specification) to those which will enable the modeling of the release and transport of aerosolized materials – the code activates modules based on the input card identification field. The most common modules used for Leak Path Factor analyses are:

- Executive Package (EXEC)
- Non-Condensable Gas Package (NCG)
- Control Volume Hydrodynamics Package (CVH)
- Flow Path Package (FL)
- Heat Structures Package (HS)
- Radio-Nuclide Package (RN)

- Control Function Package (CF)
- Tabular Function Package (TF)

Both NRC and the DOE have sponsored changes to the code, with NRC being the primary sponsor. For example, modifications were made to a version of MELCOR to model K reactor severe accidents at the DOE operated Savannah River Site. Some of this work factored into later updates of the code.

Figure 1-1 depicts a basic flowchart showing the steps required to successfully execute MELCOR.

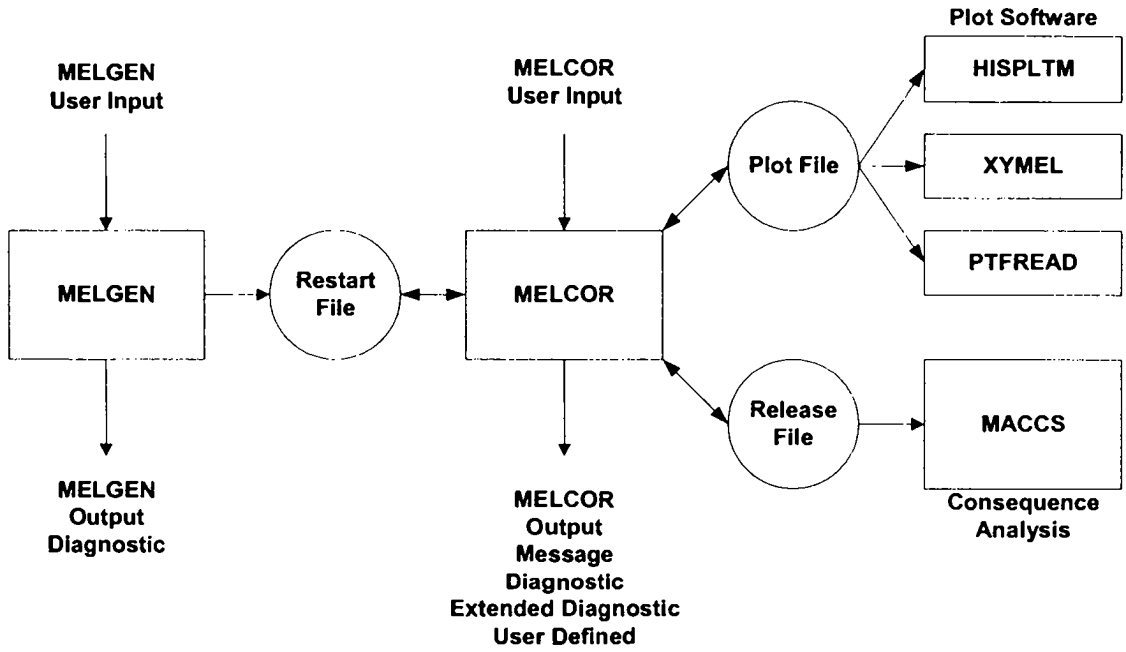


Figure 1-1 MELCOR Execution Flowchart

A brief summary of MELCOR is contained in Table 1-2.

The documents reviewed as part of the gap analysis are listed in Table 1-3.

Table 1-2 — Summary Description of the MELCOR Software in the Context of LPF Analysis

Type	Specific Information
Code Name	MELCOR - Methods for Estimation of Leakages and Consequences of Releases
Developing Organization and Sponsor	Sandia National Laboratories (SNL) for the U.S. Nuclear Regulatory Commission (primary), International Cooperative Severe Accident Research Program (CSARP) and U.S. Department of Energy (minor contribution)
Version of the Code	Version 1.8.5
Auxiliary Codes	AUXILIARY CODES: The plotting software distributed with MELCOR includes HISPLTM, XYMEL, and PTFREAD. The output from MELCOR can be input into the MACCS2 (or earlier version MACCS) code to perform consequence analysis. MELCOR INSTALL Installs software.
Software Platform/Portability	FORTRAN 77/90, PC based some system dependencies. Also runs on Unix (not tested for every platform), source code is available for HP, SUN and others.
Coding and Computer	Fortran 77, PC based 80486 or Pentium processor (C00652/PC486/00).
Technical Support	R. O. Gauntt Sandia National Laboratories P.O. Box 5800 Albuquerque, NM 87185-0748 (505) 284-3989 rogaunt@sandia.gov;
Code Procurement	The MELCOR program and comprehensive set of MELCOR documentation is available through SNL. MELCOR has a website: http://melcor.sandia.gov/. Permission from NRC is needed to acquire the code.
Code Package	Included are the references cited below. Also included are the Fortran source code, and an executable file. Training slides and a sample input deck are also available on the web site.

Table 1-2 — Summary Description of MELCOR Software in the Context of LPF Analysis (Continued)

Documentation Supplied with Code Transmittal	1. Gauntt, 2000a, Gauntt et al., *MELCOR Computer Code Manuals, Vol. 1: Primer and Users' Guide*, Version 1.8.5, NUREG/CR-6119 Rev. 2, SAND2000-2417/1, May 2000. 2. Gauntt, 2000b, Gauntt et al., *MELCOR Computer Code Manuals, Vol. 2: Reference Manuals*, Version 1.8.5, NUREG/CR-6119 Rev. 2, SAND2000-2417/2, May 2000. 3. Gauntt, 2001, Gauntt et al., *MELCOR Computer Code Manuals, Vol. 3: Demonstration Problems*, Version 1.8.5, NUREG/CR-6119 Rev. 0, SAND2001-0929P, May 2001. (Available upon request) 4. File of electronic input decks. 5. MELCOR INSTALLER. 6. Instructions for installing MELCOR for use with Digital Fortran 5/6 and Developer Studio.
Nature of Problem	MELCOR is a fully integrated, relatively fast-running code that models the progression of severe accidents in nuclear power plants. An entire spectrum of severe accident phenomena is modeled in MELCOR. Characteristics of severe accident progression that can be treated with MELCOR include the thermal-hydraulic response in the reactor coolant system, reactor cavity, containment, and confinement buildings; core heatup and degradation; radionuclide release and transport; hydrogen production, transport, and combustion; core-concrete attack; heat structure response; and the impact of engineering safety features on thermal-hydraulic and radionuclide behavior. For applications in non-reactor facilities of the DOE Complex, MELCOR has been used primarily to model in-facility transport of the release of radioactive airborne material. Deposition inside the building is calculated and the leakage to the outside environment is expressed as a fraction of the amount considered available for release and is termed the LPF.
Method of Solution	MELCOR can be used to model in-facility transport that involves the two broad areas of mixing/transport of a hazardous gas and/or aerosol transport of a hazardous material. MELCOR employs the control volume approach with lumped parameter models. MELCOR has detailed mechanistic aerosol dynamics models for the transport, deposition, and agglomeration of aerosols. Major assumptions in MELCOR include: • Each control volume gas space is well mixed, except each cell does allow for a pool covered by a gas volume. • Each gas species has the same velocity in the flow path connections. • Noncondensable gases are assumed to be ideal. • Turbulence and species diffusion within a control volume are not modeled, except in the aerosol model and condensation/evaporation on surfaces.

Table 1-2 — **Summary Description of MELCOR Software in the Context of LPF Analysis (Continued)**

Restrictions or Limitations	The control-volume, lumped-parameter approach of MELCOR does not model multi-dimensional effects, such as stratification of gases within a room. (To overcome this, one approach is to break the room into more volumes sometimes coupling the approach with computational fluid dynamics (CFD) code results.)
Run Time	The typical execution time depends on machine, detail of the model, and the length of the transient. Runtimes on the CRAY vary from 0.1 s to on the order of 1 h.[2] Runtimes for the Marviken-V Aerosol Transport Tests ATT varied from 3442 cpu(s) on a CRAY XMP-24, to 26,700 cpu(s) on a SUN Sparc2. Detailed code calculation of 24-h LaSalle Station Blackout calculation was 2 h on an HP. Simplified code calculation runtime for a 4-h sample problem transient was 15 min on an HP. The ratio of real time to runtime can vary from 0.5 to 100, depending on the nodalization.
Computer Hardware Requirements	Memory requirement is 5 MB. Depending on the model application Gigabytes of storage for output files may be required.[2]
Computer Software Requirements	MELCOR is available for the UNIX workstation platform as well as the PC platform. The execution of MELCOR on a PC is very efficient and user friendly. While either platform may be used, simply because of ease of use the latter is recommended. (A benefit of running on a PC is the ease with which output data can be processed in spreadsheet or text file programs.)
Other Versions Available	No other versions are available from SNL. INEEL and SRS both have developed specialized versions, but these are not supported by SNL and the sponsors.

[2] The data in this paragraph is dated by about 10 years. Typical run times on today's computers would be a few minutes. The most complicated models run approximately one week. Storage (output file size) is often more of limit today than run time. Actual conditions will depend on the hardware and the type of problem being executed.

Table 1-3 — Software Documentation Reviewed for MELCOR (LPF Applications)

No.	Reference
1.	Gauntt, 2000a, Gauntt et al., *MELCOR Computer Code Manuals, Vol. 1: Primer and Users' Guide*, Version 1.8.5, NUREG/CR-6119 Rev. 2, SAND2000-2417/1, May 2000.
2.	Gauntt, 2000b, Gauntt et al., *MELCOR Computer Code Manuals, Vol. 2: Reference Manuals*, Version 1.8.5, NUREG/CR-6119 Rev. 2, SAND2000-2417/2, May 2000.
3.	Gauntt, 2001, Gauntt et al., *MELCOR Computer Code Manuals, Vol. 3: Demonstration Problems*, Version 1.8.5, NUREG/CR-6119 Rev. 0, SAND2001-0929P, May 2001.
4.	SNL, 2001, Sandia National Laboratories. *5th MELCOR User's Workshop*, Bethesda, MD, May 10th – 15th, 2001.
5.	SNL 2003, Sandia National Laboratories. Nuclear Waste Management Procedure, NP 19-1, *Software Requirements*, Revision 10, Waste Isolation Pilot Plant, (May 2003).
6.	East, 1998, J.M. East and E.P. Hope, *Independent Evaluation of the MACCS2 Software Quality Assurance Program (U)*, WSRC-RP-98-00712, Westinghouse Savannah River Company, Aiken, SC (August 1998).
7.	DNFSB, 2000, 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).
8.	DOE 2004, U.S. Department of Energy. *MELCOR Computer Code Application Guidance for Leak Path Factor in Documented Safety Analysis*, (May 2004). Updates to this report are available at the DOE/EH Central Registry: http://www.eh.doe.gov/sqa/central_%20registry/MELCOR/melcor.htm
9.	SNL 1992, Sandia National Laboratories. *Software Quality Assurance Procedures for MELCOR*, Revision 1.2, (August 1992).

2.0 Assessment Summary Results

2.1 Criteria Met

Of the ten general topical quality areas assessed in the gap analysis, five satisfactorily met the criteria. The analysis found that the MELCOR SQA program (in the context of LPF applications) in general, met criteria for *Software Classification, Implementation Phase, User Instructions, Acceptance Test,* and *Configuration Control*, Requirements 1, 5, 7, 8, and 9 respectively. Five topical quality areas were not met satisfactorily. The major deficiency areas are covered below in Section 2.2 (Exceptions to Requirements). Detail on the evaluation process relative to the requirements and the criteria applied are found in Section 4.

2.2 Exceptions to Requirements

Some of the more important exceptions to criteria found for MELCOR are listed below in Table 2-1. The requirement is given, the reason the requirement was not met is provided, and remedial action(s) are listed to correct the exceptions. The ten criteria evaluated are those predominantly executed by the software developer However, it is noted that criteria for SQA Procedures/Plan, Testing, Acceptance Test, Configuration Control, and Error Notification also have requirements for the organization implementing the software. These criteria were assessed in the present evaluation only from the code developer perspective.

Table 2-1 — Summary of Important Exceptions, Reasoning, and Suggested Remediation

No.	Criterion	Reason Not Met	Remedial Action(s)
1.	SQA Procedures/Plans (Section 4.2)	SQA Plan and Procedures for Version 1.8.5 of MELCOR software were lacking components to match present day requirements. Portions of the existing version are out of date or are not currently followed.	As part of the new software baseline, the SQA Plan covering version 1.8.5 and successor versions of MELCOR should be provided to the Central Registry. SQA procedures that provide prescriptive guidance to the MELCOR software developers should be made available to a SQA evaluator for confirmatory review.
			Establish a written and approved SQA plan eliminating draft or non-compliant informal processes of development.
			Upgrade SQA program documentation, especially those procedures used for new features added in MELCOR that have an effect on modules that are typically used in LPF applications. Ensure prompt defect/error reporting.
2.	Requirements Phase (Section 4.3)	A Software Requirements Document for Version 1.8.5 of MELCOR is not available.	As part of the new software baseline for MELCOR, a Software Requirements Document should be prepared.

No.	Criterion	Reason Not Met	Remedial Action(s)
3.	Design Phase (Section 4.4)	A Software Design Document is not available. Thus, design information was not directly available. Instead, it was necessary to infer the intent of MELCOR design from model description and user guidance documents.	As part of the new software baseline for MELCOR, a Software Design Document should be prepared.
4.	Testing Phase (Section 4.6)	A Software Testing Report Document has not been produced for MELCOR, and therefore, test process and methodology could not be evaluated directly. Thus, testing process and methods had to be inferred from other information. Isolated validation studies have been previously documented for various phenomenological areas, including aerosol transport, which is the key area for LPF applications. While these studies promote confidence in the models for LPF applications, the necessary formality is lacking to make a complete evaluation.	As part of the new software baseline for MELCOR, a test case report should be prepared. An important part of the new baseline set of documentation should specifically address aerosol transport phenomena and LPF applications.
5.	Error Notification (Section 4.10)	An Error Notification and Corrective Action Report process is in place at SNL, but limited documentation is available. Users are not necessarily notified of errors. Follow up with the notifying agent is not always guaranteed, and the impact is not always assessed and reported.	While a Software Problem Reporting system is in place at SNL, it requires revision to ensure affected users are notified, closure occurs with the originator, and impact determinations are completed promptly.

2.3 Areas Needing Improvement

The gap analysis, communications with DOE, oversight organizations, safety analysts, and inputs from the long-term MELCOR users have identified a few improvements that could be made related to the code and its quality assurance. The major areas to be addressed are described in this section.

The key recommendations for improvements to MELCOR are summarized in Table 2-2.

Table 2-2 — Summary of Important Recommendations for MELCOR for LPF Applications

No.	UI – User Interface Enhancements TM – Technical Model Upgrade	Recommendation
1.	UI	Expand selection of sample problems to include those problems and releases type that are often treated in LPF analysis for Documented Safety Analyses (DSAs).
2.	UI	Provide the user more control over the printed output by allowing only selected items to print. This will help avoid

No.	UI – User Interface Enhancements TM – Technical Model Upgrade	Recommendation
		lengthy output files, and enhance post-processing. As an example, similar print options as used in MACCS2 would be useful. Consider adding in this same update an option to print summary information on the aerosol mass balance amongst volumes. This would consolidate information currently available that the user must manually extract at present, and would lessen the likelihood of error.

Item 1 in the above table will serve at least two functions. First, it will serve to enhance training for LPF. Second, it will support the LPF testing and SQA changes identified in other areas of this report.

2.4 Conclusion Regarding Software's Ability to Meet Intended Function

The MELCOR code was evaluated to determine if the software, in its current state, meets the intended function in a safety analysis context as assessed in this gap analysis. When the code is run for the intended applications as detailed in the code guidance document, MELCOR *Computer Code Application Guidance for Leak Path Factor in Documented Safety Analysis*, (DOE 2004), and also utilizing information from documentation available from SNL and other sources (Table 1-3), it is judged that it will meet the intended function.

Current software concerns and issues can be avoided by understanding MELCOR limitations and capabilities. The software can be applied for modeling those types of scenarios where precedents exist, and there is confidence that alternative analysis or experimental data would adequately confirm the code predictions.

3.0 Lessons Learned

Table 3-1 provides a summary of the lessons learned during the performance of the MELCOR gap analysis.

Table 3-1 — Lessons Learned

No.	Lesson
1.	Use of NQA-1 or other SQA criteria could not be fully verified. It is obvious that many actions characteristic of sound SQA practices have been applied in developing MELCOR, but independent confirmation of the SQA program, practices, and procedures is not possible due to lack of documentation.
2.	Observance of SQA requirements in the development of safety analysis software has not been consistent. It appears to be sporadic in application, poorly funded, and performed as an add-on activity. (Note that this is consistent with the "research" specification as given to the code.) Funding level during program development has been a key factor in determining the level of attention to SQA and the adequacy of documentation.
3.	While some evidence of pre-development planning is found for the MELCOR software, documentation is not maintained as would be expected for compliance with Quality Assurance criteria in Subpart A to 10 CFR 830 (Nuclear Safety Management).

4.	A new software baseline can be produced with "modest" resources. Initial rough estimates are 2 full-time equivalent years and should be a high priority. As time passes, knowledgeable personnel may become unavailable and it will become more difficult and costly (if not impossible) to document the QA status of the code.
5.	Additional opportunities and venues should be sought for training and user qualification on safety analysis software. This is a long-term deficiency that needs to be addressed for MELCOR LPF applications and other designated software for the DOE toolbox.

4.0 Detailed Results of the Assessment Process

Ten topical areas, or requirements, are presented in the assessment as listed in Table 4.0-1. Training and Software Improvements sections follow the 10 topical areas. Included in the software improvements section is an estimate of the resources required to upgrade MELCOR.

In the tables that follow, the topical areas or requirements are labeled as (1.x, 2.x,...,10.x) with the first value corresponding to the topical area and the second value (x), the sequential table order of each criterion. Four qualitative values shall be used to evaluate whether a specific criterion is met:

- Yes – evidence is available to confirm that the program, practices, and/or procedures followed in developing the version of code satisfy the criterion
- No – sufficient evidence does not exist to demonstrate that the code meets the criterion
- Partial - some evidence exists that the criterion is met, but has not been finalized or is incomplete
- Uncertain – no basis is available to confirm that the criterion is met.

The overall evaluation for a specific requirement is based on the evaluation of the software against the criteria.

Table 4.0-1 — Cross-Reference of Requirements with Subsection and Entry from DOE (2003e)

Subsection (This Report)	Corresponding Entry Table 3-3 from DOE (2003e)	Requirement	ASME NQA-1 2000 Section/Consensus Standards
4.1	1	Software Classification	ASME NQA-1 2000 Section 200
4.2	2	SQA Procedures/Plans	ASME NQA-1 2000 Section 200; IEEE Std. 730, *IEEE Standard for Software Quality Assurance Plans*
4.3	5	Requirements Phase	ASME NQA-1 2000 Section 401; IEEE Standard 830, *Software Requirements Specifications*
4.4	6	Design Phase	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*
4.5	7	Implementation Phase	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*

4.6	8	Testing Phase	ASME NQA-1 2000 Section 404; IEEE Std. 829, *IEEE Standard for Software Test Documentation;* IEEE Standard 1008, *Software Unit Testing*
4.7	9	User Instructions	ASME NQA-1 2000 Section 203; IEEE Standard 1063, *IEEE Standard for Software User Documentation*
4.8	10	Acceptance Test	ASME NQA-1 2000 Section 404; IEEE Std. 829, *IEEE Standard for Software Test Documentation;* IEEE Standard 1008, *Software Unit Testing*
4.9	12	Configuration Control	ASME NQA-1 2000 Section 405; ASME NQA-1 2000 Section 406
4.10	13	Error Notification	ASME NQA-1 2000 Section 203

4.1 Topical Area 1 Assessment: Software Classification

This area corresponds to the requirement entitled Software Classification in Table 3-3 of DOE (2003e).

4.1.1 Criterion Specification and Result

Table 4.1-1 lists the subset of criteria reviewed for this topical area and summarizes the findings. Sufficient documentation is provided with the software on the MELCOR website (see Table 1-2, under "Documentation Supplied with Code Transmittal"), to make an informed determination of the classification of the software. A user of the MELCOR software for LPF calculations in safety analysis applications would be expected to interpret the information on the software in light of the requirements that are discussed in Appendix A to DOE-STD-3009-94 to decide on an appropriate safety classification.

For most organizations, the safety class or safety significant classification, or Level B in the classification hierarchy discussed in DOE (2003e), would be selected. In the software requirements procedure provided by SNL, the MELCOR software would be deemed Compliance Decision (CD) software (SNL 2003).

Table 4.1-1 — Subset of Criteria for Software Classification Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
1.1	The code developer must provide sufficient information to allow the user to make an informed decision on the classification of the software.	Yes	Sufficient information is provided by the MELCOR users' manuals that are available from the software developer and the MELCOR website. Interpreted in light of Appendix A to DOE-STD-3009-94.

4.1.2 Sources and Method of Review

Documentation supplied with the MELCOR software package.

4.1.3 Software Quality-Related Issues or Concerns

There are no SQA issues or concerns relative to this requirement.

4.1.4 Recommendations

This requirement is met. No recommendations are required at this time to improve compliance with the requirement.

4.2 Topical Area 2 Assessment: SQA Procedures and Plans

This area corresponds to the requirement entitled SQA Procedures and Plans in Table 3-3 of DOE (2003c).

Use is made of an earlier independent review of the MACCS2 SQA Program (East 1998) coupled with an interview of the Sandia National Laboratories authors to determine the level of compliance with this requirement.

While the (East 1998) review focused on the MACCS2 computer code, much information was obtained on the general SQA program that existed at SNL around the time that both MACCS2 and the MELCOR software were being developed. The documented review was preceded by an in-depth review at Sandia National Laboratories in 1997. The following, based on the earlier review, provides a good synopsis of the SQA program that existed in the late 1980s and early 1990s.

SNL established a SQA program for Laboratory software in the late 1980s and early 1990s that was compliant with the IEEE Standard for SQA Plans. The final volume was put into place in 1995. The guidelines[3] are documented as shown:

[3] - The SNL documentation is clearly described as guidance. The management directing the project may choose not to follow any part, or all, of the recommendations outlined in the guidelines.

Volume 1 – Software Quality Planning [SNL, 1987]
Volume 2 – Documentation [SNL, 1995]
Volume 3 – Standards, Practices, and Conventions [SNL, 1986]
Volume 4 – Configuration Management [SNL, 1992a]; and
Volume 5 –Tools, Techniques, and Methodologies [SNL, 1989].

The following is a list and description of the necessary documents required for a complete SNL SQA package [SNL, 1986]:

Project Plan: The project plan is a brief overview of the project. It defines the project, describes the organization, proposes schedules and milestones, and defines procedures to ensure the quality of the final product.

Software Requirements Specification (SRSp): The SRSp is a description of the external interfaces and essential requirements of the software in terms of functions, performance, constraints, and attributes. Requirements are objective and measurable. The SRSp is concerned with what is required, not how to achieve it. This document is reviewed by project members, users, and management. They verify that the intent of the SRSp is clear, the software proposed by the SRSp is what is desired, and that the project can proceed to the next development stage.

Design Description: A Design Description documents the design work accomplished during the design phase. Documenting the design prior to coding avoids (or reduces) any design misunderstandings and subsequent re-coding.

Design Review Results: The results of the Design Review are documented in a report, which identifies all deficiencies discovered during the review along with a plan and schedule for corrective actions. The updated design description document, when placed under configuration control, will establish the baseline for subsequent phases of the software life cycle.

Structured Source Code: Implementation is the translation of the detailed design into a computer language; a process commonly called *coding*.

Test Set: The Test Set includes "rich" test data and relevant test procedures and tools to adequately test the application's response to valid as well as invalid data.

Test Set Documentation: The Test Set Documentation (or Software Test Plan) describes the test data, procedures, tools, and overall plan.

Test Results: The results of the tests should be documented to identify all deficiencies discovered.

Maintenance Documentation: Well-documented code and the software design document provide the backbone of maintenance documentation and the starting point for determining training needs.

Training Plan: The preparation of a well thought out training plan is an essential part of bringing a system into smooth operation. If the people, documents, and training techniques are not considered in the early planning for a new system, resources may not be available and training will be haphazard.

User's Manual or Operating Procedures: A user's manual is organized to contain practical information for the individuals required to put the software into action. Depending on the size and type of system, operating procedures may be required as a separate document to cover management of the logical and physical components. Without a properly prepared user's guide or operator instructions, either the time of the user will be wasted determining what to do, or the system will be inappropriately used, or both.

Configuration Management Plan: The Configuration Management Plan lists all modules used by the project, module locations, personnel responsible for controlling changes, and change procedures.

Baseline Table: The Baseline Table lists modules and versions in the project's baselined system.

Change Table: The Change Table lists all changes and enhancements made to the modules. Additional update supporting documents reflect changes and enhancements made to the system.

During the interview conducted with SNL personnel in January 2004, the MELCOR SQA procedures document (SNL-1992b) was provided and reviewed. The SNL(1992b) document provides SQA plan detailed information specific to MELCOR. It references (SNL 1986, SNL 1987, and SNL 1989) discussed above as primary documents. Topics covered include:

- Maintenance Procedures
  - Configuration Identification
  - Alternate Software Packages
- The Defect Investigation Report (DIR) Process
  - Request Description
  - Diagnosis
  - Resolution Plan
  - Change/Testing
  - Update Implementation
- Documenting Actions Not Involving Code Changes
- Configuration Status Accounting
- Validation and Verification of MELCOR
- MELCOR User's Guides and Reference Manuals
- Testing and Review for Code Release
- Tools, Techniques and Methodologies
- Code Written by External Suppliers
- Special Purpose Code Modifications

This plan was followed during the 1990's as MELCOR was developed and modified. The authors continue to follow the plan today, with less rigidity and with some modification as funding allows.

4.2.1 Criterion Specification and Result

Table 4.2-1 lists the subset of criteria reviewed for this topical area and summarizes the findings. Based on the SQA Program review from 1997-1998 (J. East), and East (1998), it can be inferred from the general SNL SQA information and MACCS2-specific details that most elements of a compliant SQA plan and procedures were likely in place and followed during the development of MELCOR version 1.8.5. This was confirmed by meetings with the code authors in January 2004. However, definitive confirmation through written, approved documentation is not always available.

Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
2.1	Verify that procedures/plans for SQA (**SQA Plan**) have identified organizations responsible for performing work; independent reviews, etc.	Yes.	(SNL 1992b) outlines the MELCOR software assurance plan and the procedures in place when MELCOR was developed.
2.2	Verify that procedures/plans for SQA (**SQA Plan**) have identified software engineering methods.	Yes.	(SNL 1992b) provides coding guidelines as well as steps for modifying or adding code.
2.3	Verify that procedures/plans for SQA (**SQA Plan**) have identified documentation to be required as part of program.	Yes.	(SNL 1992b) Section 4.0 provides direct reference to and plans for user's guides and reference manuals
2.4	Verify that procedures/plans for SQA (**SQA Plan**) have identified standards, conventions, techniques, and/or methodologies that shall be used to guide the software development, methods to ensure compliance with the same.	Yes.	(SNL 1992b) provides standards for coding, techniques for modifying the coding and methods to be used in program development.
2.5	Verify that procedures/plans for SQA (**SQA Plan**) have identified software reviews and schedule.	Partial.	Elements of this existed based on discussions with the authors. Software reviews were conducted. Schedules for the reviews and evidence for the thoroughness of the reviews were not found in the available documentation. (SNL 1992b) discusses testing and review in Section 5.0.
2.6	Verify that procedures/plans for SQA (**SQA Plan**) have identified methods for error reporting and corrective actions.	Yes. (Recently less rigor)	(SNL-1992b) provides discussion of the DIR (Defect Investigation Report) process. Discussion with SNL in January 2004 indicates the DIR process was rigorously followed during the 90's. With decreasing funding, error reporting has continued, but is less rigorous, with corrective actions requiring more time. Documentation and notification is less rigorous.

4.2.2 Sources and Method of Review

This review was based initially on the general SNL SQA information and the MACCS2-specific information from East (1998) and making inferences to the MELCOR code that was developed around the same timeframe as MACCS2 (MELCOR 1.8.0 released in March of 1989 and the current version 1.8.5 was released October 2000; development of MACCS2 began in 1992 with the release of the current

version 1.12 occurring in 1997). This was later supported by meetings with SNL in January 2004 specifically to discuss SQA for MELCOR. The primary reference for the SQA plan was provided in this meeting as (SNL-1992b). This plan refers to the same governing SQA documents as used by MACCS2 and reported on by East.

4.2.3 Software Quality-Related Issues or Concerns

An SQA plan for MELCOR exists. The plan is dated and consideration should be given to revising it to conform to current practices being followed for MELCOR and current day SQA expectations.

The SQA plan lacks guidance for providing design requirements for modifications being made for the code.

The SQA plan lacks detailed guidance on testing of newly developed software or modifications. Guidance should concentrate on level of testing required, type of testing, and independent verification of coding. Documentation requirements for code testing appear to be lacking. Currently modifications are made and tested against experimental results. In fact, most recent modifications are planned specifically to match to a particular type of result or experiment. This gives a level of confidence in the overall results. Testing of the coding on a line-by-line basis and for quality was not evident in the available documentation for the SQA plan although it is known this was done with varying degrees of rigor during development.

The SQA plan should address prompt error and impact notification to users. Currently (SNL-1992b) requires users be notified if funding is available. Errors or deficiencies are usually reported via email. These are then logged and if code modifications are made, they are incorporated into a future version of the code. Recently no major errors have been discovered. It may take many months for modifications resulting from any given email to be incorporated into the code and released. Not all users are notified of code modifications being made due to these emails. Documentation of detailed closure with the original email author is lacking or not formalized.

4.2.4 Recommendations

Recommendations related to this topical area are provided as follows:

- Develop an updated SQA plan for Version 1.8.5 of MELCOR (at least as the code relates to LPF analysis). (Revise as needed for future updates released for public distribution).
    - Ensure the update is consistent with the current technology and practices.
    - Ensure the plan provides specific guidance regarding design requirements and documentation of design requirements.
    - Ensure the plan addresses prompt defect/error notification to users. (At least as the errors relate to LPF analyses)

4.3 Topical Area 3 Assessment: Requirements Phase

This area corresponds to the requirement entitled Requirements Phase in Table 3-3 of DOE (2003e).

4.3.1 Criterion Specification and Results

Table 4.3-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.3-1 — Subset of Criteria for Requirements Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
3.1	Software requirements for the subject software have been established.	Partial	A verifiable, written set of software requirements is lacking. Requirements for modifications are given verbally/contractually with NRC.
3.2	Software requirements are specified, documented, reviewed and approved.	Partial.	In earlier MELCOR development efforts, written hypothetical coding plans were generated. In practice, this was found not to be beneficial and the plans would be completely rewritten or pitched. Current modifications do not generate comparable initial guidance. A verifiable, written set of software requirements is lacking.
3.3	Requirements define the functions to be performed by the software and provide detail and information necessary to design the software.	Partial.	A verifiable, written set of software requirements is lacking.
3.4	A **Software Requirements Document**, or equivalent defines requirements for functionality, performance, design inputs, design constraints, installation considerations, operating systems (if applicable), and external interfaces necessary to design the software.	Partial.	A verifiable, written set of software requirements is lacking. The contractual agreements for code development with NRC do lay out top-level direction year to year.
3.5	Acceptance criteria are established in the software requirements documentation for each of the identified requirements.	No.	A verifiable, written set of software requirements is lacking. Judgment is used as modeling progresses to discern the adequacy of model changes, usually against experiments.

4.3.2 Sources and Method of Review

This review was based on based on discussion with SNL in January 2004 and information contained in East (1998), Gauntt (2000a), Gauntt (2000b), Gauntt (2001), and (SNL 1992b).

4.3.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written Software Requirements Document for MELCOR should be addressed as part of the written SQA Plan and Procedures for this software.

4.3.4 Recommendations

Develop a Software Requirements Document for MELCOR. At a minimum, this document should address requirements related to LPF applications for meeting the prerequisites for the DOE toolbox. A broader approach would consider NRC-specified needs for the software as well and address the full capabilities of the code.

4.4 Topical Area 4 Assessment: Design Phase

This area corresponds to the requirement entitled Design Phase in Table 3-3 of DOE (2003e).

A Software Design Document has not been provided by the MELCOR software developers. To permit a limited evaluation, an alternative process was employed of reviewing MELCOR documentation for evidence that criterion requirements were met at least partially in an informal manner.

4.4.1 Criterion Specification and Result

Table 4.4-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.1	The software design was developed, documented, reviewed and controlled.	Partial.	Elements of this criterion may be inferred from code user documentation, reference manuals and discussions with SNL.
4.2	Code developer 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.	Partial.	(SNL 1992b) provides significant detail in some area on code design and modeling constraints. Similar constraints were understood by the developers when not documented on paper. Documented design requirements were lacking, therefore, documentation of having met requirements is lacking.
4.3	The following design should be present and documented: 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).	Yes.	Inferred from MELCOR documentation.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.4	The following design should be present and documented: that 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.	Yes.	Inferred from MELCOR documentation.
4.5	The following design should be present and documented: evidence of measures to mitigate the consequences of software design problems. These potential problems include external and internal abnormal conditions and events that can affect the computer program.	Partial.	The documentation of a systematic effort in this area is lacking. Practical steps were taken by the code developers to handle abnormal conditions. For example, the code developers do not let the code stop execution without a message log. Bugs and problems have been corrected over the years when found.
4.6	A Software Design Document, or equivalent, is available and contains a description of the major components of the software design as they relate to the software requirements.	No.	While there is some evidence of the design relating back to requirements as set out for the code contractually with the sponsor, there was no formal documentation available and little evidence of a systematic effort to tie final design to a set of initial requirements.
4.7	A Software Design Document, or equivalent, is available and contains 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.	Partial.	A set of the listed elements is addressed in documentation (see Section 4.4.2 of this report). Most of the models, etc. are described in detail. A formal design document was not initially generated as a part of each modification process. The authors would informally sketch out the modifications to be made. Final models as developed would normally be incorporated in the User's Manual or Reference Manuals, for major changes.
4.8	A Software Design Document, or equivalent, is available and contains a description of the allowable or prescribed ranges for inputs and outputs.	Partial	Formal design documents are lacking. However, with the supplied documentation and some experience it is possible to understand if inputs/outputs are logical and within range.
4.9	A Software Design Document, or	Yes.	Formal design documents are

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	equivalent, is available and contains the design described in a manner that can be translated into code.		lacking. However, with the supplied documentation and some experience, it is possible to translate the models and theories as described to code.
4.10	A Software Design Document, or equivalent, is available and contains 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.	Partial.	Documentation is lacking. Most modifications are initiated as part of a project to compare to test data or experiment.
4.11	The organization responsible for the design identified and documented the particular verification methods to be used and assured that an Independent Review was performed and documented. This review evaluated the technical adequacy of the design approach; assured internal completeness, consistency, clarity, and correctness of the software design; and verified that the software design is traceable to the requirements.	Partial.	Evidence of substantial peer review exists. Documentation of completeness is difficult to corroborate. Documentation of pre-planning in software design documents is lacking.
4.12	The organization responsible for the design assured that the test results adequately demonstrated the requirements were met.	Partial.	A verifiable, written set of documentation of software design requirements is lacking. Evidence exists that substantial testing was performed.
4.13	The Independent Review was performed by competent individual(s) other than those who developed and documented the original design, but who may have been from the same organization.	Partial.	Significant independent review has been performed. Documentation of reviewer qualifications and independence is lacking. For example, there is evidence of peer review during the 1990-91 timeframe from training slide material that is available from the MELCOR website (SNL, 2001). The NRC reviews code modules when completed by SNL.
4.14	The results of the Independent Review are documented with the identification of the verifier indicated.	Partial.	Significant independent review has been performed. Complete documentation is lacking.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.15	If review alone was not adequate to determine if requirements are met, alternate calculations were used, or tests were developed and integrated into the appropriate activities of the software development cycle.	Partial.	A verifiable, written set of documentation of software design requirements is lacking. Significant independent review has been performed. The code has been modified over the years and tested to provide reasonable assurance the models are adequate.
4.16	Software design documentation was completed prior to finalizing the Independent Review.	Partial.	Some review was known to have been conducted in parallel with design documentation preparation or before preparation of its equivalent.
4.17	The extent of the Independent Review and the methods chosen are shown to be a function of: the importance to safety, the complexity of the software, the degree of standardization, and the similarity with previously proven software.	Partial.	Integrated documentation of the design requirements is lacking, as is documentation of the review detail and its bases. Judgment was used by the code developers to determine what would be reviewed and when. MELCOR has undergone many man-years of independent review and is believed to be robust. Elements of this activity have been documented by various organizations at various times for varying applications and models.

4.4.2 Sources and Method of Review

SNL personnel were interviewed in January 2004. Design requirements were evaluated through review of the following documents:

Gauntt, 2000a, Gauntt et al., *MELCOR Computer Code Manuals, Vol. 1: Primer and Users' Guide*, Version 1.8.5, NUREG/CR-6119 Rev. 2, SAND2000-2417/1, May 2000.

Gauntt, 2000b, Gauntt et al., *MELCOR Computer Code Manuals, Vol. 2: Reference Manuals*, Version 1.8.5, NUREG/CR-6119 Rev. 2, SAND2000-2417/2, May 2000.

Gauntt, 2001, Gauntt et al., *MELCOR Computer Code Manuals, Vol. 3: Demonstration Problems*, Version 1.8.5, NUREG/CR-6119 Rev. 0, SAND2001-0929P, May 2001.

SNL, 2001, Sandia National Laboratories. 5th *MELCOR User's Workshop*, Bethesda, MD, May 10th – 15th, 2001.

SNL 2003, Sandia National Laboratories. Nuclear Waste Management Procedure, NP 19-1, *Software Requirements*, Revision 10, Waste Isolation Pilot Plant, (May 2003).

SNL (1992b). *Software Quality Assurance Procedures for MELCOR*. Sandia National Laboratories

4.4.3 Software Quality-Related Issues or Concerns

A verifiable, written Software Design Document for MELCOR should be part of the written SQA Plan and Procedures for this software. Upgrades to the Model Description and other documentation can meet the intent of the Software Design Document for an interim period. However, in reconstituting the baseline for MELCOR, it is highly desirable that a new Software Design Document be developed. At a minimum, the Software Design Document should cover those modules that are used in LPF calculations.

4.4.4 Recommendations

Model descriptions in the MELCOR reference manual and other documentation and undocumented practices followed meet the intent of the software design document for the time being. Internal and independent testing of the existing code modules is believed to be robust. However, a software design report addressing the above table elements should be prepared. It is recommended that existing information on aerosol transport (theory, models, model results, tests, experiments, etc.) be gathered and consolidated and that the MELCOR LPF models be verified and validated against these within the context of the elements in Table 4.4-1.

4.5 Topical Area 5 Assessment: Implementation Phase

This area corresponds to the requirement entitled Implementation Phase in Table 3-3 of DOE (2003e).

4.5.1 Criterion Specification and Result

Table 4.5-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
5.1	The implementation process resulted in software products such as computer program listings and instructions for computer program use.	Yes.	User guide, model description, and code listing from the MELCOR transmittal confirm that the essential features of this criterion are met.
5.2	Implemented software was analyzed to identify and correct errors.	Yes.	Test problems exercising the model components are run prior to each release.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
5.3	The source code finalized during verification (this phase) was placed under configuration control.	Yes.	(SNL-1992b) is followed and configuration control is maintained on beta versions as well as release versions.
5.4	Documentation during verification included a copy of the software, test case description and associated criteria that are traceable to the software requirements and design documentation.	Yes.	Copy of software and test case description are available. Not possible to trace to requirements and design documents which are lacking documentation.

4.5.2 Sources and Method of Review

Documentation listed in Table 1-3 was reviewed to complete review of this criterion. The code listing is available from SNL with transmittal of MELCOR to requesting user groups.

4.5.3 Software Quality-Related Issues or Concerns

Not all criteria can be confirmed due to the lack of written records on implementation. However, based on available information, it is inferred that most of these requirements were met.

4.5.4 Recommendations

No recommendations related to this topical area are made.

4.6 Topical Area 6 Assessment: Testing Phase

This area corresponds to the requirement entitled Testing Phase in Table 3-3 of DOE (2003e). A Software Test Report has not been provided by the MELCOR software developers. Instead, a limited evaluation is performed applying Gauntt (2001), and the related documents listed in Table 1-3 as a basis to address the criteria in Table 4.6-1.

4.6.1 Criterion Specification and Result

Table 4.6-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
6.1	The software was validated by executing test cases.	Yes.	Documentation, especially Gauntt (2001), supports the satisfaction of this criterion.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
6.2	Testing demonstrated the capability of the software to produce valid results for test cases encompassing the range of permitted usage defined by the program documentation. Such activities ensured that the software adequately and correctly performed all intended functions.	Yes.	A series of test cases are run prior to release exercising most of the modules. Other testing is performed ad-hoc by the code authors.
6.3	Testing demonstrated that the computer program properly handles abnormal conditions and events as well as credible failures	Yes.	A series of test cases are run prior to release exercising most of the modules. Other testing is performed ad-hoc by the code authors.
6.4	Testing demonstrated that the computer program does not perform adverse unintended functions.	Yes.	A series of test cases are run prior to release exercising most of the modules. Other testing is performed ad-hoc by the code authors.
6.5	Test Phase activities were performed to assure adherence to requirements, and to assure that the software produces correct results for the test case specified. Acceptable methods for evaluating adequacy of software test case results included: (1) analysis with computer assistance; (2) other validated computer programs; (3) experiments and tests; (4) standard problems with known solutions; (5) confirmed published data and correlations.	Partial	A series of test cases are run prior to release exercising most of the modules. Other testing is performed ad-hoc by the code authors. Significant work has been performed to compare results to experiment. Current suite of test cases (Volume III) supplied with software includes commercial reactor and experimental facility examples. Documentation of requirements is lacking.
6.6	Test Phase documentation includes test procedures or plans and the results of the execution of test cases. The test results documentation demonstrates successful completion of all test cases or the resolution of unsuccessful test cases and provides direct traceability between the test results and specified software requirements.	Partial.	Only partial record of testing is available. It is known that testing was conducted on MELCOR, and it is judged that the final version (1.8.5) performs as intended. However, resolution of unsuccessful cases is not possible to check, nor is traceability between test results and software requirements.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
6.7	Test procedures or plans specify the following, as applicable: (1) Required tests and test sequence, (2) Required range of input parameters, (3) Identification of the stages at which testing is required, (4) Requirements for testing logic branches, (5) Requirements for hardware integration, (6) Anticipated output values, (7) Acceptance criteria, (8) Reports, records, standard formatting, and conventions, (9) Identification of operating environment, support software, software tools or system software, hardware operating system(s) and/or limitations.	Partial.	A series of test cases are run prior to release exercising most of the modules. Other testing is performed ad-hoc by the code authors. No comprehensive detailed record of test procedures and plans was available. It can be inferred that this criterion was partially met. Complete verification was not possible due to lack of documentation.

4.6.2 Sources and Method of Review

SNL personnel were interviewed and documentation listed in Table 1-3 was reviewed.

4.6.3 Software Quality-Related Issues or Concerns

Lack of a test report for MELCOR forces the review to infer test case program results and outcome based on limited information. Volume 3 of the MELCOR 1.8.5 code manual (Gauntt, 2001) contains a portfolio of sample demonstration problems. These problems are a combination of experiment analyses, which illustrate code model performance against data, and full plant analyses showing MELCOR's performance on larger realistic problems. A few of these problems address, at least partially, aerosol transport, which is a key phenomenological area for LPF applications. While these studies promote confidence in the models for LPF applications, the documentation of these tests lack the necessary formality and comprehensiveness to address all components of the testing phase criterion.

4.6.4 Recommendations

A verifiable, written Test Report Document for MELCOR should be part of the written SQA Plan and Procedures for this software. Upgrades to the MELCOR software baseline will require that a Test Case Description and Report be completed. Test cases should include one or more example types that serve to demonstrate adequacy of the MELCOR software for LPF calculations that are representative of applications for DOE safety analysis. The Test Report and test phase documentation should address each of the above table elements.

4.7 Topical Area 7 Assessment: User Instructions

This area corresponds to the requirement entitled User Instructions in Table 3-3 of DOE (2003e).

User instructions for MELCOR have been documented (Gauntt, 2000a; Gauntt, 2000b). Considered along with DOE-specific input preparation guidance in DOE (2003c), there is sufficient information to evaluate compliance to this requirement.

4.7.1 Criterion Specification and Result

Table 4.7-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
7.1	A description of the model is documented.	Yes.	MELCOR models are described sufficiently (Gauntt, 2000a; Gauntt, 2000b).
7.2	User's manual or guide includes approved operating systems (for cases where source code is provided, applicable compilers should be noted).	Yes.	(Gauntt, 2000a; Gauntt, 2000b)
7.3	User's manual or guide includes description of the user's interaction with the software.	Yes.	(Gauntt, 2000a; Gauntt, 2000b)
7.4	User's manual or guide includes a description of any required training necessary to use the software.	Partial.	The MELCOR primer document discusses an approach a new user might take to become familiar with the code.
7.5	User's manual or guide includes input and output specifications.	Yes.	The User's manual (Gauntt, 2000a, Gauntt 2000b)
7.6	User's manual or guide includes a description of software and hardware limitations.	Yes.	The Reference Manual discusses the physics and models.
7.7	User's manual or guide includes a description of user messages initiated as a result of improper input and how the user can respond.	Yes.	The code and manuals provide adequate diagnostics.
7.8	User's manual or guide includes information for obtaining user and maintenance support.	Yes.	The MELCOR website contains email and phone contact information.

4.7.2 Sources and Method of Review

Compliance with this requirement was evaluated by review of documentation listed in Table 1.3. SNL personnel were interviewed in January 2004.

4.7.3 Software Quality-Related Issues or Concerns

User instruction documentation is good. No substantive issues or concerns have surfaced.

4.7.4 Recommendations

Recommendations related to this topical area are as follows:

- A simple training program would be useful. This could take several forms including a training manual, or interactive course. The novice user could be tasked with two to three simple problem types and walked through them with output information and explanation. The current sample case file could take on this function with expansion and concentration on LPF related elements.
- MELCOR limitations should be made more explicit in the User's Guide. Specific attention to limitations should be a focused topic and to the extent practical collected in one location.

4.8 Topical Area 8 Assessment: Acceptance Test

This area corresponds to the requirement entitled Acceptance Test Table 3-3 of DOE (2003e). During this phase of the software development, the software becomes part of a system incorporating applicable software components, hardware, and data, and then is accepted for use. Much of the testing is the burden of the user organization, but the developing organization assumes some responsibility.

4.8.1 Criterion Specification and Result

Table 4.8-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
8.1	To the extent applicable to the developer, acceptance testing includes a comprehensive test in the operating environment(s).	Yes.	Volume III (Gauntt 2001) and the electronic files provided allow the user to run a thorough test of the software. The sample problems should expand to provide one or more LPF specific cases.
8.2	To the extent applicable to the developer, acceptance testing was performed prior to approval of the computer program for use.	Yes.	Sample problem sets are run prior to release and checked. Errors or problems are corrected before release.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
8.3	To the extent applicable to the developer, software validation was performed to ensure that the installed software product satisfies the specified software requirements. The engineering function (i.e., an engineering operation an item is required to perform to meet the component or system design basis) determines the acceptance testing to be performed prior to approval of the computer program for use.	Yes.	While documentation of requirements and comprehensive testing is lacking, the code is checked with a series of problems, and individual module testing is performed during development. Most new major modifications are compared against experiment and all are corrected before release.
8.4	Acceptance testing documentation includes results of the execution of test cases for system installation and integration, user instructions (Refer to Requirement 7 above), and documentation of the acceptance of the software for operational use.	Yes.	Volume III (Gauntt 2001) and the electronic files provided allow the user to run a thorough test of the software. Output for comparison is provided. Instructions are provided for installation.

4.8.2 Sources and Method of Review

Software package for code transmittal and documentation listed in Table 1.3 were reviewed. SNL personnel were interviewed in January 2004.

4.8.3 Software Quality-Related Issues or Concerns

There are no software quality issues or concerns for this requirement.

4.8.4 Recommendations

No recommendations are made for this topical area.

4.9 Topical Area 9 Assessment: Configuration Control

This area corresponds to the requirement entitled Configuration Control in Table 3-3 of (DOE 2003e).

4.9.1 Criterion Specification and Result

Table 4.9-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
9.1	For the developers the methods used to	Yes.	(SNL - 1992b) provides details

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	control, uniquely identify, describe, and document the configuration of each version or update of a computer program (for example, source, object, back-up files) and its related documentation (for example, software design requirements, instructions for computer program use, test plans, and results) are described in implementing procedures.		of required configuration control of the code and its related documentation.
9.2	Implementing procedures meet applicable criteria for configuration identification, change control and configuration status accounting.	Yes.	(SNL-1992b) provides details.

4.9.2 Sources and Method of Review

SNL personnel were interviewed in January 2004. (SNL-1992b) was reviewed and discussed.

4.9.3 Software Quality-Related Issues or Concerns

There are no software quality issues or concerns for this requirement.

4.9.4 Recommendations

No recommendations are made for this topical area.

4.10 Topical Area 10 Assessment: Error Impact

This area corresponds to the requirement entitled Error Impact in Table 3-3 of DOE (2003e).

4.10.1 Criterion Specification and Result

Table 4.10-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
10.1	The problem reporting and corrective action process used by the software developing organization addresses the appropriate requirements of the developing organization's corrective action system, and are documented in implementing procedures.	Yes.	The process used for monitoring errors and user feedback on MELCOR is defined in (SNL-1992b). This was formerly strictly followed. It continues to be followed, but less rigidly than before, in part, because of funding considerations.
10.2	Method(s) for documenting (Error Notification and Corrective Action Report), evaluating, and correcting software problems describe the evaluation process for determining whether a reported problem is an error.	Partial.	Some guidance is given in (SNL-1992b). Judgment is used by the authors to determine the severity of the error. Formal specifications to help with this judgment are lacking.
10.3	Method(s) for documenting (Error Notification and Corrective Action Report), evaluating, and correcting software problems define the responsibilities for disposition of the problem reports, including notification to the originator of the results of the evaluation.	Partial.	Guidance is given in (SNL-1992b) Errors and defects are handled by logging them and including updates in the next release. Notification is lacking formality usually associated with a safety related code. Procedures state notification depends on funding. NRC as the current sponsor and SNL define MELCOR as a research code. The reporting scheme currently conforms to this definition.
10.4	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the error relates to appropriate software engineering elements.	Yes.	Guidance is given in (SNL-1992b).
10.5	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the error impacts past and present use of the computer program	Partial.	Some guidance is given in (SNL-1992b). In practice, this may be accomplished but is not automatic and is left to the judgment of the authors.
10.6	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the corrective action	No.	No information was available to support that this occurs formally. Rather consistency of personnel and experience

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	impacts previous development activities		are used to the extent this is accomplished.
10.7	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the users are notified of the identified error, its impact; and how to avoid the error, pending implementation of corrective actions.	No.	Errors and defects are handled by logging them and including updates in the next release. Notification is lacking formality. Procedures state notification depends on funding. NRC as the current sponsor and SNL define MELCOR as a research code. The reporting scheme conforms to this definition.

4.10.2 Sources and Method of Review

SNL personnel were interviewed in January 2004. SNL has an informal Software Reporting system. The MELCOR website has a link to send an e-mail to MELCOR technical staff. Staff indicated that email is the primary means by which defects are reported. Through the FAQ link on the MELCOR website, users can read about problems other users have reported and see the response of the MELCOR technical staff. The effectiveness or timeliness of this system, however, is difficult to judge. Under the FAQ link, the MELCOR technical staff relays user-reported problems, discuss the causes of error messages, and provide tips to avoid discovered problems until a patch or new version is distributed. As of January 2004, six problems were addressed at the FAQ link. None have been identified as having any significant impact on LPF results.

4.10.3 Software Quality-Related Issues or Concerns

While an informal Software Reporting system process is institutionalized at SNL, its effectiveness can not be established. The authors make concerted effort to record emails they receive, and log the information as it comes in internally. Notification to users of defects on a timely basis, close out with the defect reporter, and formal impact determination are in need of improvement.

4.10.4 Recommendations

As part of the new software baseline for MELCOR, a comprehensive Software Error Notification and Corrective Action process should be provided. Expanded use of the MELCOR website or its equivalent is suggested to provide timely reporting of user issues, errors and defects. It may also provide software news, suggested strategies for resolving software problems, and general communications. Timely, formal user notification of errors or defects should be addressed.

4.11 Training Program Assessment

Current MELCOR training opportunities are limited and not well publicized. Comprehensive training on a more frequent basis would be beneficial.

The Energy Facility Contractors Group (EFCOG) Workshops provide two annual opportunities to give training to the DOE users. The winter session is during the Safety Basis Subgroup meeting and the summer session is organized for the larger Safety Analysis Working Group. Multi-day MELCOR training at these two workshops would potentially reach 300 DOE MELCOR users, managers, regulators, and oversight groups.

In May 2004 the MELCOR Code Application Program (MCAP) group is planning to meet near Washington DC. The first day of this meeting is closed to non-members. Potential exists to add training for MELCOR, both general, or specific to LPF, at the end of this meeting.

Training could result in MELCOR LPF certification/qualification. This level of user proficiency could be measured by demonstrating competency through a written exam and software execution of a set of test cases. Ideally, this could be accomplished through formal course attendance or through a self directed (self-study) process.

4.12 Software Improvements and New Baseline

The minimum remedial program required to yield the new software baseline for MELCOR was discussed earlier as part of Table 1.1. Included are upgrades to software documents that constitute the baseline for software, including:

- Updated Software Quality Assurance Plan
- Software Requirements Document (Specific to LPF)
- Software Design Document (Specific to LPF)
- Test Case Description and Report (Specific to LPF)
- Updated Software Configuration and Control
- Updated Error Notification and Corrective Action Report Procedure, and
- Updated User's Manual.

The SNL procedural guide NP-19 implements an earlier version of Subpart 2.7 to NQA-1, specifically NQA-2a-1990. Application of this procedure was assessed for the SNL MACCS2 code with the result being the minimum set of actions as documented in Bixler (2000) and shown below in Table 4.12-1. Column "SNL NP 19-1 (Bixler)". Application of this procedure to MELCOR can be expected to result in a similar set of actions as specified in the column labeled "Corresponding Recommended Steps from this GAP analysis".

While not exactly matching up with the recommendations proposed in this gap analysis, the SNL proposed program is similar to the requirements outlined in this report. Furthermore, the estimates are based on SNL resources, and as such, are taken as more accurate resource estimates than could be provided otherwise. The overall SQA upgrade program in the SNL program was estimated to require 1.5 full-time equivalent years to complete. The requirements are matched against the requirements earlier, in Table 4.12-1. The overall level of effort, 1.5 FTE-years is rounded up to approximately 2 FTE-years as the final estimate for resource allocation to perform the upgrades required to compensate for MELCOR's

known SQA gaps. This is a very rough estimate based on this comparison, extrapolating from MACCS to MELCOR and considering the differences. It assumes there would not be major defects found as the program is completed and that existing information would be adequate to complete verification and validation of the LPF models. Long term, maintenance funding will be required for activities such as defect reporting, coordinated update testing as NRC makes changes in the future, and minor SQA administrative duties.

Table 4.12-1 — Comparison of SQA Upgrade Steps Discussed in Bixler (2000) with the Approach Discussed in DOE (2003e)

Topic No.	Topic: ASME NQA-1-2000 Requirements	Level B Existing Software (Topic Applied?)	GAP Report Section No.	SNL NP 19-1 Steps (Bixler)	Compliance Steps in this GAP Document, DOE (2003e)
1	Software Classification	Yes	4.1	None	None
2	SQA Procedures/ Plans	Yes	4.2	Create a Primitive Baseline (PB) document to establish the SQA status of the existing code	Update SQA plan
3	Dedication	No[4]	–	–	–
4	Evaluation	No[4]	–	–	–
5	Requirements	Yes	4.3	Write a Software Requirements Document (SRD)	Write a Software Requirements Document (SRD)
6	Design Phase	Yes	4.4	None	Write a Design Document
7	Implementation Phase	Yes	4.5	Create an Implementation Document (ID) to describe the process of generating the executable software modules	Create an Implementation Document (ID) to describe the process of generating the executable software modules

[4] Topic evaluated as not significantly relevant to safety analysis toolbox codes.

8	Testing Phase	Yes	4.6	Establish a Verification and Validation Plan (VVP) based on the SRD; Generate a Validation Document (VD), to measure the performance of the software against the criteria specified in the VVP	Establish a Verification and Validation Plan (VVP) based on the SRD; Generate a Validation Document (VD), to measure the performance of the software against the criteria specified in the VVP
9	User Instruc-tions	Yes	4.7	Update, the User's Manual (UM)	Update, the User's Manual (UM)
10	Accept-ance Test	Yes	4.8	Perform Installation and Checkout (I&C) to verify correct installation on all supported platforms	None (normally done for MELCOR))
11	Opera-tion and Mainten-ance	No[4]	-	—	–
12	Config-uration Control	Yes	4.9	Implement a Software Configuration Control System (CC)	Update Software Configuration Control System (CC)
13	Error Impact	Yes	4.10	Implement a Software Problem Reporting System (SPR)	Update Software Problem Reporting System (SPR)
14	Access Control	No[4]	–	–	–

5.0 Conclusions

The gap analysis for Version 1.8.5 of the MELCOR software, based on a set of requirements and criteria compliant with NQA-1, has been completed. Of the ten primary SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), five requirements are met at an acceptable level, i.e., *Software Classification (1), Implementation Phase (5). User Instruction s(7), Acceptance Test (8),* and *Configuration Control (9).* Five topical quality areas were not met satisfactorily. Improvement actions are recommended for MELCOR to fully meet SQA criteria for the remaining five requirements.

It was determined that the MELCOR code as applied to LPF calculations does meet its intended function for use in supporting documented safety analysis. However, as with all safety-related software, users should be aware of current limitations and capabilities of MELCOR for supporting safety analysis. Informed use of the software can be assisted by the current set of MELCOR reports (refer to Table 1-3), and the code guidance report for DOE safety analysts, *MELCOR Computer Code Application Guidance for Leak Path Factor in Documented Safety Analysis*, (DOE, 2004). Furthermore, while SQA improvement actions are recommended for MELCOR as applied to LPF calculations, no evidence has been found of programming, logic, or other types of software errors in MELCOR that have led to non-conservatisms in nuclear facility operations, or in the identification of facility controls.

By order of priority, it is recommended that MELCOR software improvement actions be taken, especially:

1. Correcting known defects in the SQA process
2. Upgrading existing SQA documentation
3. Providing training on a regular basis, and
4. Revising and developing new software documentation.

A new software baseline set of documents is recommended for MELCOR to demonstrate completion of the revision to software documentation item (above). The list of revised baseline documents includes:

- Updated Software Quality Assurance Plan
- Software Requirements Document (Specific to LPF)
- Software Design Document (Specific to LPF)
- Test Case Description and Report (Specific to LPF)
- Updated Software Configuration and Control
- Updated Error Notification and Corrective Action Report Procedure, and
- Updated User's Manual.

Approximately two full-time equivalent years is conservatively estimated to upgrade MELCOR software to be compliant with NQA-1-based requirements for existing software. Of this level of effort, 1.5 FTE is estimated for the current software owner, Sandia National Laboratories, and roughly, 0.5 FTE is estimated to be required for independent review.

A new version of MELCOR is planned for release in the future. It is recommended that this version be evaluated upon issue relative to the software improvement and baseline recommendations, and the full set of SQA criteria discussed in this report. If this version is found to be satisfactory, it should replace Version 1.8.5 as designated version of the software for the toolbox.

Approximately one FTE-month per year would be needed to maintain a web-based error notification and corrective action process for MELCOR (Section 4.10). However, such a process has not been defined in depth for MELCOR and the other designated toolbox codes.

6.0 Acronyms and Definitions

ACRONYMS:

AEC	Atomic Energy Commission
ALOHA	Areal Locations of Hazardous Atmospheres (designated toolbox software)
ANS	American Nuclear Society
ANSI	American National Standards Institute
ASME	American Society of Mechanical Engineers
CCPS	Center for Chemical Process Safety
CD	Compliance Decision
CFAST	Consolidated Fire and Smoke Transport Model (designated toolbox software)
CFD	Computational Fluid Dynamics
CFR	Code of Federal Regulations
CSARP	Cooperative Severe Accident Research Program
DCF	Dose Conversion Factor
DIR	Defect Investigation Report
DNFSB	Defense Nuclear Facilities Safety Board
DoD	Department of Defense
DOE	Department of Energy
DSA	Documented Safety Analysis
EFCOG	Energy Facility Contractors Group
EH	DOE Office of Environment, Safety and Health
EIA	Electronic Industries Alliance
EM	DOE Office of Environmental Management
EPIcode	Emergency Prediction Information code (designated toolbox software)
EPRI	Electric Power Research Institute
FTE	Full-time equivalent
GENII	Generalized Environmental Radiation Dosimetry Software System - Hanford Dosimetry System (Generation II) (designated toolbox software)
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IP	Implementation Plan
ISO	International Organization for Standardization
MACCS2	MELCOR Accident Consequence Code System 2 (designated toolbox software)
MELCOR	Methods for Estimation of Leakages and Consequences of Releases (designated toolbox software)
NNSA	National Nuclear Security Administration
NRC	Nuclear Regulatory Commission
OCRWM	Office of Civilian Radioactive Waste Management
PSA	Probabilistic Safety Analysis (or Assessment)
QAP	Quality Assurance Program (alternatively, Plan)
RSICC	Radiation Safety Information Computational Center
SNL	Sandia National Laboratories
SQA	Software Quality Assurance
SRS	Savannah River Site
V&V	Verification and Validation
WSRC	Westinghouse Savannah River Company
YMP	Yucca Mountain Project

DEFINITIONS:

The following definitions are taken from the Implementation Plan. References in brackets following definitions indicate the original source, when not the Implementation Plan.

Central Registry — An organization designated to be responsible for the storage, control, and long-term maintenance of the Department's safety analysis "toolbox codes." The central registry may also perform this function for other codes if the Department determines that this is appropriate.

Firmware — The combination of a hardware device and computer instructions and data that reside as read-only software on that device. [IEEE Standard 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology]

Gap Analysis — Evaluation of the Software Quality Assurance attributes of specific computer software against identified criteria.

Nuclear Facility — A reactor or a nonreactor nuclear facility where an activity is conducted for or on behalf of DOE and includes any related area, structure, facility, or activity to the extent necessary to ensure proper implementation of the requirements established by 10 CFR 830. [10 CFR 830]

Safety Analysis and Design Software — Computer software that is not part of a structure, system, or component (SSC) but is used in the safety classification, design, and analysis of nuclear facilities to ensure proper accident analysis of nuclear facilities; proper analysis and design of safety SSCs; and, ensure the proper identification, maintenance, and operation of safety SSCs. [DOE O 414.1B]

Safety Analysis Software Group (SASG) — A group of technical experts formed by the Deputy Secretary in October 2000 in response to Technical Report 25 issued by the Defense Nuclear Facilities Safety Board (DNFSB). This group was responsible for determining the safety analysis and instrument and control (I&C) software needs to be fixed or replaced, establishing plans and cost estimates for remedial work, providing recommendations for permanent storage of the software and coordinating with the Nuclear Regulatory Commission on code assessment as appropriate.

Safety-Class Structures, Systems, and Components (SC SSCs) — SSCs, including portions of process systems, whose preventive and mitigative function is necessary to limit radioactive hazardous material exposure to the public, as determined from the safety analyses. [10 CFR 830]

Safety-Significant Structures, Systems, and Components (SS SSCs) — SSCs which are not designated as safety-class SSCs, but whose preventive or mitigative function is a major contributor to defense in depth and/or worker safety as determined from safety analyses. [10 CFR 830] As a general rule of thumb, SS SSC designations based on worker safety are limited to those systems, structures, or components whose failure is estimated to result in prompt worker fatalities, serious injuries, or significant radiological or chemical exposure to workers. The term, serious injuries, as used in this definition, refers to medical treatment for immediately life-threatening or permanently disabling injuries (e.g., loss of eye, loss of limb). The general rule of thumb cited above is neither an evaluation guideline nor a

quantitative criterion. It represents a lower threshold of concern for which an SS SSC designation may be warranted. Estimates of worker consequences for the purpose of SS SSC designation are not intended to require detailed analytical modeling. Consideration should be based on engineering judgment of possible effects and the potential added value of SS SSC designation. [DOE G 420.1-1]

Safety Software — Includes both safety system software, and safety analysis and design software. [DOE O 414.1B]

Safety Structures, Systems, and Components (SSCs) — The set of safety-class SSCs and safety-significant SSCs for a given facility. [10 CFR 830]

Safety System Software — Computer software and firmware that performs a safety system function as part of a structure, system, or component (SSC) that has been functionally classified as Safety Class (SC) or Safety Significant (SS). This also includes computer software such as human-machine interface software, network interface software, programmable logic controller (PLC) programming language software, and safety management databases that are not part of an SSC but whose operation or malfunction can directly affect SS and SC SSC function. [DOE O 414.1B]

Safety Analysis and Design Software – Computer software that is not part of a structure, system, or component (SSC) but is used in the safety classification, design, and analysis of nuclear facilities to ensure the proper accident analysis of nuclear facilities; the proper analysis and design of safety SSCs; and, the proper identification, maintenance, and operation of safety SSCs. [DOE O 414.1B]

Software — Computer programs, operating systems, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. [IEEE Standard 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology]

Toolbox Codes — A small number of standard computer models (codes) supporting DOE safety analysis, having widespread use, and of appropriate qualification that are maintained, managed, and distributed by a central source. Toolbox codes meet minimum quality assurance criteria. They may be applied to support 10 CFR 830 DSAs provided the application domain and input parameters are valid. In addition to public domain software, commercial or proprietary software may also be considered. In addition to safety analysis software, design codes may also be included if there is a benefit to maintain centralized control of the codes. [modified from DOE N 411.1]

Validation — 1) The process of testing a computer program and evaluating the results to ensure compliance with specified requirements. [ANSI/ANS-10.4-1987]
2) The process of determining the degree to which a model is an accurate representation of the real-world from the perspective of the intended uses of the model. [Department of Defense Directive 5000.59, DoD Modeling and Simulation (M&S) Management]

Verification — 1) The process of evaluating the products of a software development phase to provide assurance that they meet the requirements defined for them by the previous phase. [ANSI/ANS-10.4-1987]
2) The process of determining that a model implementation accurately represents the developer's conceptual description and specifications. [Department of Defense Directive 5000.59, DoD Modeling and Simulation (M&S) Management]

7.0 References

Note: The references listed below may not have been used directly in the gap analysis. However, they were used to provide a context for performing the overall code evaluation.

Bixler, N. (2000). *Proposal to Resolve QA Deficiencies in MACCS2*, Memorandum to D. Chung (DOE/DP), Sandia National Laboratories, Albuquerque, NM (2000).

CFR Code of Federal Regulations (10 CFR 830). 10 CFR 830, Nuclear Safety Management Rule.

DNFSB Defense Nuclear Facilities Safety Board, (2000). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, Technical Report DNFSB/TECH-25, (January 2000).

DNFSB Defense Nuclear Facilities Safety Board, (2002). *Recommendation 2002-1, Quality Assurance for Safety-Related Software* , (September 2002).

DOE, U.S. Department of Energy (2000a). *Appendix A, Evaluation Guideline,* DOE-STD-3009-94, *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports* (January 2000).

DOE, U.S. Department of Energy (2000b). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, DOE Response to TECH-25, Letter and Report, (October 2000).

DOE, U.S. Department of Energy (2002). *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports*, DOE-HDBK-3010-94, Change Notice 2 (April 2002).

DOE, U.S. Department of Energy (2003a). *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).

DOE, U.S. Department of Energy (2003b). *Designation of Initial Safety Analysis Toolbox Codes*, Letter, (March 28, 2003).

DOE, U.S. Department of Energy (2003c). *Assessment Criteria and Guidelines for Determining the Adequacy of Software Used in the Safety Analysis and Design of Defense Nuclear Facilities*, Report, CRAD-4.2.4-1, Rev 0, (August 27 2003).

DOE, U.S. Department of Energy (2003d). *Software Quality Assurance Improvement Plan: Format and Content For Code Guidance Reports*, Revision A (draft), Report, (August 2003).

DOE, U.S. Department of Energy (2003e). *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes*, Revision 1, (November 2003).

DOE, U.S. Department of Energy (2004). MELCOR Computer Code Application Guidance for Leak Path Factor in Documented Safety Analysis, (May 2004).

East, J. M. (1998) and E. P. Hope. *Independent Evaluation of the MACCS2 Software Quality Assurance Program (U)*, WSRC-RP-98-00712, Westinghouse Savannah River Company, Aiken, SC (August 1998).

Gauntt, R. O. (2000a) et al. *MELCOR Computer Code Manuals, Vol. 1: Primer and Users' Guide*, Version 1.8.5, NUREG/CR-6119 Rev. 2, SAND2000-2417/1, May 2000.

Gauntt, R. O. (2000b) et al. *MELCOR Computer Code Manuals, Vol. 2: Reference Manuals*, Version 1.8.5, NUREG/CR-6119 Rev. 2, SAND2000-2417/2, May 2000.

Gauntt, R. O. (2001) et al. *MELCOR Computer Code Manuals, Vol. 3: Demonstration Problems*, Version 1.8.5, NUREG/CR-6119 Rev. 0, SAND2001-0929P, May 2001.

SNL (1986). *Sandia Software Guidelines: Volume 3: Standards, Practices, and Conventions.* Sandia National Laboratories, Albuquerque, NM, SAND85-2346.

SNL (1987). *Sandia Software Guidelines: Volume 1: Software Quality Planning.* Sandia National Laboratories, Albuquerque, NM, SAND85-2344.

SNL (1989). *Sandia Software Guidelines: Volume 5: Tools, Techniques, and Methodologies.* Sandia National Laboratories, Albuquerque, NM, SAND85-2348.

SNL (1992a). *Sandia Software Guidelines: Volume 4: Configuration Management.* Sandia National Laboratories, Albuquerque, NM, SAND85-2347.

SNL (1992b). *Software Quality Assurance Procedures for MELCOR.* Sandia National Laboratories, Albuquerque, NM, Revision 1.2, August 2, 1992.

SNL (1995). *Sandia Software Guidelines: Volume 2: Documentation.* Sandia National Laboratories, Albuquerque, NM, SAND85-2345.

SNL (2001). *5th MELCOR User's Workshop*, Sandia National Laboratories, Bethesda, MD, May 10th – 15th, 2001.

SNL (2003). *Software Requirements*, Revision 10, Nuclear Waste Management Procedure, NP 19-1, Waste Isolation Pilot Plant, (May 2003).

Appendices

Appendix	Subject
A	Software Information Template

APPENDIX A. SOFTWARE INFORMATION TEMPLATE

The following is a condensed version of the information request sent to the MELCOR code developer in October 2003. *(Note: This information is provided to give the reader of this report, an idea of the information requested to complete the gap analysis for MELCOR. Detailed information in response was not filled in. See Section 1.6. Instead, the contacts and the SQA evaluators used the form as a guide for continual discussion throughout the gap analysis for MELCOR.*

Information Form

Development and Maintenance of Designated Safety Analysis Toolbox Codes

The following summary information in Table 2 should be completed to the level that is meaningful – enter N/A if not applicable. See Appendix A for an example of the input to the table prepared for the MELCOR code.

Table 2. Summary Description of Subject Software

Table 2. Summary Description of Subject Software	
Type	**Specific Information**
Code Name	
Version of the Code	
Developing Organization and Sponsor Information	
Auxiliary Codes	
Software Platform/Portability	
Coding and Computer(s)	
Technical Support Point of Contact	
Code Procurement Point of Contact	
Code Package Label/Title	
Contributing Organization(s)	

Table 2. Summary Description of Subject Software	
Type	**Specific Information**
Recommended Documentation - Supplied with Code Transmittal upon Distribution or Otherwise Available	1. 2. 3. 4. 5.
Input Data/Parameter Requirements	
Summary of Output	
Nature of Problem Addressed by Software	
Significant Strengths of Software	
Known Restrictions or Limitations	
Preprocessing (set-up) time for Typical Safety Analysis Calculation	
Execution Time	
Computer Hardware Requirements	
Computer Software Requirements	
Other Versions Available	

Table 2. Summary Description of Subject Software	
Type	Specific Information

Table 3. Point of Contact for Form Completion

Individual(s) completing this information form: Name: Organization: Telephone: Email: Fax:	

1. **Software Quality Assurance Plan**

The software quality assurance plan for your software may be either a standalone document, or embedded in other documents, related procedures, QA assessment reports, test reports, problem reports, corrective actions, supplier control, and training package.

1.a For this software, identify the governing Software Quality Assurance Plan (SQAP)?
[Please submit a PDF of the SQAP, or send hard copy of the SQAP[5]]

1.b What software quality assurance industry standards are met by the SQAP?

1.c What federal agency standards were used, if any, from the sponsoring organization?

1.d Has the SQAP been revised since the current version of the Subject Software was released? If so, what was the impact to the subject software?

1.e Is the SQAP proceduralized in your organization? If so, please list the primary procedures that provide guidance.

Guidance for SQA Plans:

[5] Notify Kevin O'Kula of your intent to send hard copies of requested reports and shipping will be arranged.

Requirement 2 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a)
ASME NQA-1 2000 Section 200
IEEE Standard 730, *IEEE Standard for Software Quality Assurance Plans.*
IEEE Standard 730.1, *IEEE Guide for Software Quality Assurance Planning.*

2. Software Requirements Description

The software requirements description (SRD) should contain functional and performance requirements for the subject software. It may be contained in a standalone document or embedded in another document, and should address functionality, performance, design constraints, attributes and external interfaces.

2.a **For this software, was a software requirements description documented with the software sponsor?** [If available, please submit a PDF of the Software Requirements Description, or include hard copy with transmittal of SQAP]

2.b **If a SRD was not prepared, are there written communications that indicate agreement on requirements for the software? Please list other sources of this information if it is not available in one document.**

Guidance for Software Requirements Documentation:

Requirement 5 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 401
IEEE Standard 830, *Software Requirements Specifications*

3. Software Design Documentation

The software design documentation (SDD) depicts how the software is structured to satisfy the requirements in the software requirements description. It should be defined and maintained to ensure that software will serve its intended function. The SDD for the subject software may be contained in a standalone document or embedded in another document.

The SDD should provide the following:

- Description of the major components of the software design as they relate to the software requirements,
- Technical description of the software with respect to the theoretical basis, mathematical model, control flow, data flow, control logic, and data structure,
- Description of the allowable or prescribed ranges of inputs and outputs,
- Design described in a manner suitable for translating into computer coding, and
- Computer program listings (or suitable references).

3.a **For the subject software, was a software design document prepared, or were its constituents parts covered elsewhere?** [If available, please submit a PDF of the Software Design Document, or include hard copy with transmittal of SQAP]

3.b If the intent of the SDD information is satisfied in other documents, provide the appropriate references (document number, section, and page number).

Guidance for Software Design Documentation:

Requirement 6 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
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*
IEEE Standard 1012, *IEEE Standard for Software Verification and Validation;*
IEEE Standard 1012a, *IEEE Standard for Software Verification and Validation – Supplement to 1012*

4. Software User Documentation

Software User Documentation is necessary to assist the user in installing, operating, managing, and maintaining the software, and to ensure that the software satisfies user requirements. At minimum, the documentation should describe:

- The user's interaction with the software
- Any required training
- Input and output specifications and formats, options
- Software limitations
- Error message identification and description, including suggested corrective actions to be taken to correct those errors, and
- Other essential information for using the software.

4.a **For the subject software, has Software User Documentation been prepared, or are its constituents parts covered elsewhere?** [If available, please submit a PDF of the Software User Documentation, or include a hard copy with transmittal of SQAP]

4.b **If the intent of the Software User Documentation information is satisfied in other documents, provide the appropriate references (document number, section, and page number).**

4.c **Training – How is training offered in correctly running the subject software? Complete the appropriate section in the following:**

Type	Description	Frequency of training
Training Offered to User Groups as Needed		
Training Sessions Offered at Technical Meetings or Workshops		
Training Offered on Web or Through Video Conferencing		
Other Training Modes		
Training Not Provided		

Type	Description	Frequency of training

Guidance for Software User Documentation:

Requirement 9 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 203
IEEE Standard 1063, *IEEE Standard for Software User Documentation*

5. Software Verification &Validation Documentation (Includes Test Reports)

Verification and Validation (*V&V*) documentation should confirm that a software V&V process has been defined, that V&V has been performed, and that related documentation is maintained to ensure that:

(a) The software adequately and correctly performs all intended functions, and
(b) The software does not perform any unintended function.

The software V&V documentation, either as a standalone document or embedded in other documents and should describe:

- The tasks and criteria for verifying the software in each development phase and validating it at completion,
- Specification of the hardware and software configurations pertaining to the software V&V
- Traceability to both software requirements and design
- Results of the V&V activities, including test plans, test results, and reviews (also see 5.b below)
- A summary of the status of the software's completeness
- Assurance that changes to software are subjected to appropriate V&V,
- V&V is complete, and all unintended conditions are dispositioned before software is approved for use, and
- V&V performed by individuals or organizations that are sufficiently independent.

5.a **For the subject software, identify the V&V Documentation that has been prepared.**
[If available, please submit a PDF of the Verification and Validation Documentation, or include a hard copy with transmittal of SQAP]

5.b **If the intent of the V&V Documentation information is satisfied in one or more other documents, provide the appropriate references (document number, section, and page number). For example, a "Test Plan and Results" report, containing a plan for software testing, the test results, and associated reviews may be published separately.**

5.c **Testing of software: What has been used to test the subject software?**

☐ Experimental data or observations
☐ Standalone calculations
☐ Another validated software
☐ Software is based on previously accepted solution technique

Provide any reports or written documentation substantiating the responses above.

Guidance for Software Verification & Validation, and Testing Documentation:

Requirement 6 – *Design Phase* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
Requirement 8 – *Testing Phase* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
Requirement 10 – *Acceptance Test* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 402 (Note: Some aspects of verification may be handled as part of the Design Phase).
ASME NQA-1 2000 Section 404 (Note: Aspects of validation may be handled as part of the Testing Phase).
IEEE Standard 1012, *IEEE Standard for Software Verification and Validation;*
IEEE Standard 1012a, *IEEE Standard for Software Verification and Validation – Supplement to 1012*
IEEE Standard 829, *IEEE Standard for Software Test Documentation.*
IEEE Standard 1008, *Software Unit Testing*

6. Software Configuration Management (SCM)

A process and related documentation for SCM should be defined, maintained, and controlled.

The appropriate documents, such as project procedures related to software change controls, should verify that a software configuration management process exists and is effective.

The following points should be covered in SCM document(s):

- A Software Configuration Management Plan, either in standalone form or embedded in another document,
- Configuration management data such as software source code components, calculational spreadsheets, operational data, run-time libraries, and operating systems,
- A configuration baseline with configuration items that have been placed under configuration control,
- Procedures governing change controls,
- Software change packages and work packages to demonstrate that (1) possible impacts of software modifications are evaluated before changes are made, (2) various software system products are examined for consistency after changes are made, and (3) software is tested according to established standards after changes have been made.

6.a **For the subject software, has a Software Configuration Management Plan been prepared, or are its constituent parts covered elsewhere?** [If available, please submit a PDF of the Software Configuration Management Plan and related procedures, or include hard copies with transmittal of SQAP].

6.b **Identify the process and procedures governing control and distribution of the subject software with users.**

6.c **Do you currently interact with a software distribution organization such as the Radiation Safety Information Computational Center (RSICC)?**

6.d A Central Registry organization, under the management and coordination of the Department of Energy's Office of Environment, Safety and Health (EH), will be responsible for the long-term maintenance and control of the safety analysis toolbox codes for DOE safety analysis applications. Indicate any questions, comments, or concerns on the Central Registry's role and the maintenance of the subject software.

Guidance for Software Configuration Management Plan Documentation:

Requirement 12 *Configuration Control* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 203
IEEE Standard 828, *IEEE Standard for Software Configuration Management Plans.*

7. Software Problem Reporting and Corrective Action

Software problem reporting and corrective action documentation help ensure that a formal procedure for problem reporting and corrective action development for software errors and failures is established, maintained, and controlled.

A Software Error Notification and Corrective Action Report, procedure, or similar documentation, should be implemented to report, track, and resolve problems or issues identified in both software items, and in software development and maintenance processes. Documentation should note specific organizational responsibilities for implementation. Software problems should be promptly reported to affected organizations, along with corrective actions. Corrective actions taken ensure that:

- Problems are identified, evaluated, documented, and, if required, corrected,
- Problems are assessed for impact on past and present applications of the software by the responsible organization,
- Corrections and changes are executed according to established change control procedures, and
- Preventive actions and corrective actions results are provided to affected organizations.

Identify documentation specific to the subject software that controls the error notification and corrective actions. [If available, please submit a PDF of the Error Notification and Corrective Action Report documentation for the subject software (or related procedures). If this is not available, include hard copies with transmittal of SQAP].

7.aProvide examples of problem/error notification to users and the process followed to address the deficiency. Attach files as necessary.

7.bProvide an assessment of known errors or defects in the subject software and the planned action and time frame for correction.

Category of Error or Defect	Corrective Action	Planned schedule for correctic
Major		

Minor		

7.c Identify the process and procedures governing communication of errors/defects related to the subject software with users.

Guidance for Error/Defect Reporting and Corrective Action Documentation:

Requirement 13 – *Error Impact* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 204
IEEE Standard 1063, *IEEE Standard for Software User Documentation*

8. Resource Estimates

If one or more plans, documents, or sets of procedures identified in parts one (1) through seven (7) do not exist, please provide estimates of the resources (full-time equivalent (40-hour) weeks, FTE-weeks) and the duration (months) needed to meet the specific SQA requirement.

Enter estimate in Table 4 only if specific document has not been prepared, or requires revision.

Table 4. Resource and Schedule for SQA Documentation

Plan/Document/Procedure	Resource Estimate (FTE-weeks)	Duration of Activity (months)
1. Software Quality Assurance Plan		
2. Software Requirements Document		
3. Software Design Document		
4. Test Case Description and Report		
5. Software Configuration and Control		
6. Error Notification and Corrective Action Report		
7. User's Instructions (User's Manual)		
8. Other SQA Documentation		

Comments or Questions:

9. Software Upgrades

Describe modifications planned for the subject software.

Technical Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

User Interface Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Software Engineering Improvements

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Other Planned Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Thank you for your input to the SQA upgrade process. Your experience and insights are critical towards successfully resolving the issues identified in DNFSB Recommendation 2002-1.

SEPARATION

PAGE

Defense Nuclear Facilities Safety Board Recommendation 2002-1
Software Quality Assurance Improvement Plan
Commitment 4.2.1.3:

Software Quality Assurance Improvement Plan: EPIcode Gap Analysis

Final Report

U.S. Department of Energy
Office of Environment, Safety and Health
1000 Independence Ave., S.W.
Washington, DC 20585-2040

May 2004

INTENTIONALLY BLANK

FOREWORD

This report documents the outcome of an evaluation of the Software Quality Assurance (SQA) attributes of the chemical source term and atmospheric dispersion computer code, EPIcode, relative to established requirements. This evaluation, a "gap analysis", is performed to meet commitment 4.2.1.3 of the Department of Energy's Implementation Plan to resolve SQA issues identified in the Defense Nuclear Facilities Safety Board Recommendation 2002-1.

Suggestions for corrections or improvements to this document should be addressed to –

Chip Lagdon
EH-31/GTN
U.S. Department of Energy
Washington, D.C. 20585-2040
Phone (301) 903-4218
Email: chip.lagdon@ch.doe.gov

INTENTIONALLY BLANK

REVISION STATUS

Page/Section	Revision	Change
1. Entire Document	1. Interim Report	1. Original Issue
2. Entire Document	2. Final Report, May 3, 2004	2. Updated all sections per review comments.

INTENTIONALLY BLANK

CONTENTS

Section			Page
FOREWORD			**III**
REVISION STATUS			**V**
EXECUTIVE SUMMARY			**XIII**
1.0	**INTRODUCTION**		**1-1**
	1.1	BACKGROUND: OVERVIEW OF DESIGNATED TOOLBOX SOFTWARE IN THE CONTEXT OF 10 CFR 830	1-1
	1.2	EVALUATION OF TOOLBOX CODES	1-2
	1.3	USES OF THE GAP ANALYSIS	1-2
	1.4	SCOPE	1-2
	1.5	PURPOSE	1-3
	1.6	METHODOLOGY FOR GAP ANALYSIS	1-3
	1.7	SUMMARY DESCRIPTION OF SOFTWARE BEING REVIEWED	1-5
2.0	**ASSESSMENT SUMMARY RESULTS**		**2-1**
	2.1	CRITERIA MET	2-1
	2.2	EXCEPTIONS TO REQUIREMENTS	2-1
	2.3	AREAS NEEDING IMPROVEMENT	2-2
	2.4	CONCLUSION REGARDING CODES ABILITY TO MEET INTENDED FUNCTION	2-2
3.0	**LESSONS LEARNED**		**3-3**
4.0	**DETAILED RESULTS OF THE ASSESSMENT PROCESS**		**4-3**
	4.1	TOPICAL AREA 1 ASSESSMENT: SOFTWARE CLASSIFICATION	4-4
		4.1.1 Criterion Specification and Result	*4-4*
		4.1.2 Sources and Method of Review	*4-5*
		4.1.3 Software Quality-Related Issues or Concerns	*4-5*
		4.1.4 Recommendations	*4-5*
	4.2	TOPICAL AREA 2 ASSESSMENT: SQA PROCEDURES AND PLANS	4-5
		4.2.1 Criterion Specification and Result	*4-6*
		4.2.2 Sources and Method of Review	*4-6*
		4.2.3 Software Quality-Related Issues or Concerns	*4-6*
		4.2.4 Recommendations	*4-6*
	4.3	TOPICAL AREA 3 ASSESSMENT: REQUIREMENTS PHASE	4-7
		4.3.1 Criterion Specification and Result	*4-7*
		4.3.2 Sources and Method of Review	*4-8*
		4.3.3 Software Quality-Related Issues or Concerns	*4-8*
		4.3.4 Recommendations	*4-8*
	4.4	TOPICAL AREA 4 ASSESSMENT: DESIGN PHASE	4-9
		4.4.1 Criterion Specification and Result	*4-9*
		4.4.2 Sources and Method of Review	*4-11*
		4.4.3 Software Quality-Related Issues or Concerns	*4-11*
		4.4.4 Recommendations	*4-11*
	4.5	TOPICAL AREA 5 ASSESSMENT: IMPLEMENTATION PHASE	4-12

4.5.1 Criterion Specification and Result *4-12*
4.5.2 Sources and Method of Review *4-12*
4.5.3 Software Quality-Related Issues or Concerns *4-13*
4.5.4 Recommendations *4-13*
4.6 TOPICAL AREA 6 ASSESSMENT: TESTING PHASE 4-13
4.6.1 Criterion Specification and Result *4-13*
4.6.2 Sources and Method of Review *4-15*
4.6.3 Software Quality-Related Issues or Concerns *4-15*
4.6.4 Recommendations *4-15*
4.7 TOPICAL AREA 7 ASSESSMENT: USER INSTRUCTIONS 4-15
4.7.1 Criterion Specification and Result *4-16*
4.7.2 Sources and Method of Review *4-17*
4.7.3 Software Quality-Related Issues or Concerns *4-17*
4.7.4 Recommendations *4-17*
4.8 TOPICAL AREA 8 ASSESSMENT: ACCEPTANCE TEST 4-17
4.8.1 Criterion Specification and Result *4-17*
4.8.2 Sources and Method of Review *4-18*
4.8.3 Software Quality-Related Issues or Concerns *4-18*
4.8.4 Recommendations *4-19*
4.9 TOPICAL AREA 9 ASSESSMENT: CONFIGURATION CONTROL 4-19
4.9.1 Criterion Specification and Result *4-19*
4.9.2 Sources and Method of Review *4-19*
4.9.3 Software Quality-Related Issues or Concerns *4-19*
4.9.4 Recommendations *4-20*
4.10 TOPICAL AREA 10 ASSESSMENT: ERROR IMPACT 4-20
4.10.1 Criterion Specification and Result *4-20*
4.10.2 Sources and Method of Review *4-23*
4.10.3 Software Quality-Related Issues or Concerns *4-23*
4.10.4 Recommendations *4-23*
4.11 TRAINING PROGRAM ASSESSMENT 4-23
4.12 SOFTWARE IMPROVEMENTS 4-23

5.0 CONCLUSION **5-1**

6.0 ACRONYMS AND DEFINITIONS **6-1**

7.0 REFERENCES **7-1**

APPENDIX A. — SOFTWARE INFORMATION TEMPLATE A-1

TABLES

	Page
Table 1-1 — Plan for SQA Evaluation of Existing Safety Analysis Software	1-3
Table 1-2 — Summary Description of EPIcode Software	1-6
Table 1-3 — Software Documentation Reviewed for EPIcode	1-9
Table 2-1 — Summary of Important Exceptions, Reasoning, and Suggested Remediation	2-1
Table 2-2 — Summary of Important Recommendations for EPIcode	2-2
Table 4-0-1— Cross-Reference of Requirements with Subsection and Entry from DOE (2003e)	4-3
Table 4.1-1 — Subset of Criteria for Software Classification Topic and Results	4-4
Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results	4-6
Table 4.3-1 — Subset of Criteria for Requirements Phase Topic and Results	4-7
Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results	4-9
Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results	4-12
Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results	4-13
Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results	4-16
Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results	4-18
Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results	4-19
Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results	4-20

INTENTIONALLY BLANK

FIGURES

None

INTENTIONALLY BLANK

Software Quality Assurance Improvement Plan: EPIcode Gap Analysis

EXECUTIVE SUMMARY

The Defense Nuclear Facilities Safety Board (DNFSB) issued Recommendation 2002-1 on *Quality Assurance for Safety-Related Software* in September 2002 (DNFSB 2002). The Recommendation identified a number of quality assurance issues for software used in the 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 high-use, Software Quality Assurance (SQA)-compliant safety analysis codes is one of the major improvement actions discussed in the *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities*. A DOE safety analysis toolbox would contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed and maintained for DOE-broad safety basis applications.

The EPIcode 7.0 software for chemical source term and atmospheric dispersion and consequence analysis, is one of the codes designated for the toolbox. To determine the actions needed to bring the EPIcode 7.0 software into compliance with the SQA qualification criteria, and develop an estimate of the resources required to perform the upgrade, the Implementation Plan has committed to sponsoring a code-specific gap analysis document. The gap analysis evaluates the software quality assurance attributes of EPIcode 7.0 against identified criteria.

The balance of this document provides the outcome of the EPIcode gap analysis compliant with NQA-1-based requirements. Of the ten SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), two requirements are met at acceptable level, i.e., *Classification* (1) and *User Instructions* (7). Improvement actions are recommended for EPIcode to fully meet the remaining eight requirements. This evaluation outcome is deemed acceptable because: (1) EPIcode 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 EPIcode is limited to those analytic applications for which the software is intended.

Suggested remedial actions for this software would warrant upgrading software documents. The complete list of revised 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 Manual.

It is estimated that a concentrated program to upgrade the SQA pedigree of EPIcode to be compliant with the ten criteria discussed here would require fourteen to sixteen full-time equivalent (FTE)-months.

It was determined that the EPIcode 7.0 does meet its intended function for use in supporting documented safety analysis. However, as with all safety-related software, users should be aware of current limitations and capabilities of the software for supporting safety analysis. Informed use of the code can be assisted by appropriate use of current EPIcode documentation and the EPIcode guidance report for DOE safety analysts, *EPIcode Computer Code Application Guidance for Documented Safety Analysis*, (DOE, 2004). Furthermore, while SQA improvement actions are recommended for EPIcode, no evidence has been found of programming, logic, or other types of software errors in EPIcode 7.0 that have led to non-conservatisms in nuclear facility operations, or in the identification of facility controls.

INTENTIONALLY BLANK

1.0 Introduction

This document reports on the results of a gap analysis for Version 7.0 of the EPIcode computer code. The intent of the gap analysis is to determine the actions needed to bring the designated software into compliance with established Software Quality Assurance (SQA) criteria. A secondary aspect of this report is to develop an estimate of the level of effort required to upgrade each code based on the gap analysis results.

1.1 Background: Overview of Designated Toolbox Software in the Context of 10 CFR 830

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* (DNFSB, 2000). 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.

While progress was made in resolving several of the issues raised in TECH-25, the DNFSB issued Recommendation 2002-1 on *Quality Assurance for Safety-Related Software* in September 2002. 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, the quality of the software used to design or develop safety-related controls, and the proficiency of personnel using the software. The Recommendation identified a number of quality assurance issues for software used in the 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 high-use, SQA-compliant safety analysis codes is one of the major commitments contained in the February 28, 2003 *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities* (IP). In time, the DOE safety analysis toolbox will contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed and maintained for DOE-broad safety basis applications.

Six computer codes, including ALOHA (chemical release dispersion/consequence analysis), CFAST (fire analysis), EPIcode (chemical release dispersion/consequence analysis), GENII (radiological dispersion/consequence analysis), MACCS2 (radiological dispersion/consequence analysis), and MELCOR (leak path factor analysis), were designated by DOE for the toolbox (DOE/EH, 2003). It is found that this software provides generally recognized and acceptable approaches for modeling source term and consequence phenomenology, and can be applied as appropriate to support accident analysis in Documented Safety Analyses (DSAs).

As one of the designated toolbox codes, EPIcode Version 7.0 is likely to require some degree of quality assurance improvement before meeting current SQA standards. The analysis of this document evaluates EPIcode Version 7.0 relative to current software quality assurance criteria. It assesses the extent of the deficiencies, or gaps, to provide DOE and the software developer the extent to which minimum upgrades are needed. The overall assessment is therefore termed a "gap" analysis.

1.2 Evaluation of Toolbox Codes

The quality assurance criteria identified in later sections of this report are defined as the set of established requirements, or basis, by which to evaluate each designated toolbox code. This evaluation process, a gap analysis, is commitment 4.2.1.3 in the IP:

> Perform a SQA evaluation to the toolbox codes to determine the actions needed to bring the codes into compliance with the SQA qualification criteria, and develop a schedule with milestones to upgrade each code based on the SQA evaluation results.

This process is a prerequisite step for software improvement. It will allow DOE to determine the current limitations and vulnerabilities of each code as well as help define and prioritize the steps required for improvement.

Ideally, each toolbox code owner will provide complete information on the SQA programs, processes, and procedures used to develop their software. However, the gap analysis itself will be performed by a SQA evaluator. The SQA evaluator is independent of the code developer, but knowledgeable in the use of the software for accident analysis applications and current software development standards.

1.3 Uses of the Gap Analysis

The gap analysis provides key information to DOE, code developers, and code users.

DOE obtains the following benefits:

- Estimate of the resources required to perform modifications to designated toolbox codes
- Basis for schedule and prioritization to upgrade each designated toolbox code.

Each code developer is provided:

- Information on areas where software quality assurance improvements are needed to comply with industry SQA standards and practices
- Specific areas for improvement to guide development of new versions of the software.

DOE safety analysts and code users benefit from:

- Improved awareness of the strengths, limits, and vulnerable areas of each computer code
- Recommendations for code use in safety analysis application areas.

1.4 Scope

This analysis is applicable to the EPIcode, one of the six designated toolbox codes for safety analysis. While EPIcode is the subject of the current report, other safety analysis software considered for the toolbox in the future may be evaluated with the same process applied here. The template outlined here is applicable for any analytical software as long as the primary criteria are ASME NQA-1, 10 CFR 830, and related DOE directives discussed in DOE (2003e).

1.5 Purpose

The purpose of this report is to document the gap analysis performed on the EPIcode as part of DOE's implementation plan on SQA improvements.

1.6 Methodology for Gap Analysis

The gap analysis for EPIcode is based on the plan and criteria described in *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes* (DOE 2003e). The overall methodology for the gap analysis is summarized in Table 1-1. The gap analysis reported here utilizes ten of the fourteen topical areas listed in DOE (2003e) related to software quality assurance to assess the quality of the EPIcode 7.0 computer code. The ten areas are those particularly applicable to the software development, specifically: (1) Software Classification, (2) SQA Procedures/Plans, (5) Requirements Phase, (6) Design Phase, (7) Implementation Phase, (8) Testing Phase, (9) User Instructions, (10) Acceptance Test, (12) Configuration Control, and (13) Error Impact. Each area, or requirement, is assessed individually in Section 4. Each area or requirement is assessed individually in Section 4.

Requirements 3 (Dedication), 4 (Evaluation), 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 evaluating the areas listed above. Requirement 11 (Operation and Maintenance) is only partially applicable to software development, and is interpreted to be applicable mostly to the software user organization.

An information template was transmitted to the Safety Analysis Software Developers on 20 October 2003 to provide basic information as input to the gap analysis process (O'Kula, 2003). The core section of the template is attached as Appendix A to the present report. It is noted that the written response provided by the EPIcode software developer to the information template was incomplete.

Table 1-1 — Plan for SQA Evaluation of Existing Safety Analysis Software[1]

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).

[1] Originally documented as Table 2-2 in DOE (2003e).

Phase	Procedure
	c. Identify documents that are acceptable from SQA perspective. Note inadequate documents as appropriate.

Table 1-1 — Plan for SQA Evaluation of Existing Safety Analysis Software (continued)

Phase	Procedure
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.
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.

1.7 Summary Description of Software Being Reviewed

The gap analysis was performed on version 7.0 of the EPIcode® (note: EPIcode® is a registered trademark of Homann Associates, Inc.). EPIcode was developed by Homann Associates, Inc., which maintains and upgrades the code. The code is commercially available from Homann Associates, Inc. The technical contact for EPIcode is the code author, Steven Homann (www.epicode.com, or epicode@aol.com).

EPIcode performs calculations for source terms and downwind concentrations. Source term calculations determine the rate at which the chemical material is released to the atmosphere, release height, release duration, and the form and properties of the chemical upon release. The analyst specifies the chemical and then either specifies the chemical source term rate or provides EPIcode with the necessary information and data to calculate a steady evaporation rate when the scenario involves a spill of a chemical liquid. Releases may be elevated either through discharge from a stack or as a result of plume rise from buoyancy or momentum effects. The EPIcode considers the chemical cloud emission to be neutrally buoyant and applies standard Gaussian puff and plume models as appropriate. In addition to the source term and downwind concentration calculations, EPIcode supports the use of concentration limits

for the purpose of consequence assessment (e.g., assessment of human health risks from contaminant plume exposure). When available, data for Immediately Dangerous to Life or Health (IDLH), Emergency Response Planning Guidelines (ERPGs), Department of Energy Temporary Emergency Exposure Limits (TEELs), and EPA Acute Exposure Guideline Limits (AEGLs) have been incorporated into the chemical library of EPIcode.

A brief summary of EPIcode that was supplied code developer is summarized in Table 1-2.

Table 1-2 — Summary Description of EPIcode Software

Type	Specific Information
Code Name	EPIcode®
Version of the Code	Version 7.0
Developing Organization and Sponsor Information	Homann Associates, Inc.
Auxiliary Codes	N/A
Software Platform/Portability	Microsoft™ Visual Basic Professional 6.0, PC-based
Coding and Computer(s)	Microsoft™ Visual Basic Professional 6.0, PC-based 80486 or Pentium processor Windows 95/98/00/NT/XP OS
Technical Support Point of Contact	Homann Associates, Inc. (510) 490-6379 epicode@aol.com www.epicode.com
Code Procurement Point of Contact	Homann Associates, Inc. (510) 490-6379 epicode@aol.com www.epicode.com
Code Package Label/Title	EPIcode 7.0, single CD
Contributing Organization(s)	N/A
Recommended Documentation - Supplied with Code Transmittal upon Distribution or Otherwise Available	EPIcode documentation and user manual are components of EPIcode 7.0 onboard runtime library. Users access this information via a command button or the F1 key.

Table 1-2 -- Summary Description of EPIcode Software (Continued)

Type	Specific Information
Input Data/Parameter Requirements	Source Term substance: via name, CAS number, DOT Number, TEEL database name (rev 19). Source Term: Total release rate or total release (g/s, g, etc.) Airborne Fraction (AF) .The fraction of the total quantity of material that remains airborne. Deposition velocity (cm/sec). Effective release height (m). Explosive Release Modules: High Explosive (pounds TNT equivalent). Fuel Fire Module: Volume of Fuel (gallons), Burn duration (minutes), Heat emission rate (calories/second). Radius of fire zone (m). Optional Source Term Geometry: Horizontal Dimension (meters), Vertical Dimension (meters), Height (meters). Wind Speed (m/s) at input reference height. Wind Direction (compass degrees) for geographical mapping overlay Stability Class (A-G) Receptor Height (meters). Inversion Layer Height (meters) Washout Coefficient (1/second), for washout plume depletion and ground deposition.
Summary of Output	Results from EPIcode atmospheric release calculations can be displayed or printed in tabular form or as graphic plots showing the downwind centerline concentration or concentration contours. All files can be archived. EPIcode contours can also be displayed on any .bmp image, e.g., satellite maps, map photos, etc. Off-axis locations can also be included in the tabular output.
Nature of Problem Addressed by Software	EPIcode has been specially developed to provide emergency response personnel, emergency planners, and health and safety professionals with a software tool to aid them in evaluating the atmospheric release of toxic substances.

Table 1-2 — Summary Description of EPIcode Software (Continued)

Type	Specific Information
Significant Strengths of Software	EPIcode is completely menu-driven and easy to use. EPIcode uses the same algorithms and methodologies outlined in EPA document titled "Technical Guidance for Hazards Analysis -Emergency Planning for Extremely Hazardous Substances," U.S. Environmental Protection Agency, Federal Emergency Management Agency, and U.S. Department of Transportation, December 1987. EPIcode output always contains all of the input assumptions, and the calculated radii of the vulnerable zones are in exact agreement with the above EPA document. EPIcode contains a library of over 2,000 chemical substances along with the associated exposure levels accepted by various professional organizations and regulatory agencies. These include all of the current American Industrial Hygiene Association Emergency Response Planning Guidelines (ERPGs), Department of Energy Temporary Emergency Exposure Limits (TEELs), and EPA Acute Exposure Guideline Limits (AEGLs). The EPIcode Library also contains information on substances listed in the Threshold Limit Values for Chemical Substances and Physical Agents and Biological Exposure Indices published by the American Conference of Governmental Industrial Hygienists. IDLH (Immediately Dangerous to Life or Health) data are also included when available. Virtual source terms are used to more accurately model the initial distribution of material associated with explosions or fires.
Known Restrictions or Limitations	The atmospheric model included in the code does not model the impact of terrain effects on atmospheric dispersion. A single wind direction and input height is assumed.
Preprocessing (set-up) time for Typical Safety Analysis Calculation	Few minutes or less
Execution Time	Less than 5 seconds
Computer Hardware Requirements	Any PC running Microsoft™ Windows 95/98/00/NT/XP OS (Fully operational on Apple™ computers running Windows 95/98 emulator software)
Computer Software Requirements	Microsoft™ Windows 95/98/00/NT/XP OS
Other Versions Available	N/A

Table 1-2 — Summary Description of EPIcode Software (Continued)

Type	Specific Information
Individual(s) completing this information form: Name: Organization: Telephone: Email: Fax:	Steven Homann Homann Associates, Inc. Voice: (510) 490-6379 Email: epicode@aol.com Fax: (510) 490-6379 Web: www.epicode.com

The set of documents reviewed as part of the gap analysis are listed in Table 1-3.

Table 1-3 — Software Documentation Reviewed for EPIcode

No.	Reference
1.	*EPIcode Version 7.0 User Documentation* (EPIcode, 2003) {Online Help distributed with software package}
2.	*Technical Guidance for Hazards Analysis: Emergency Planning for Extremely Hazardous Substances* (EPA, 1987) {Source of algorithms and methodologies that are used in EPIcode}
3.	*Risk Management Program Guidance for Offsite Consequences* (EPA, 1999) {Source of updated evaporation model (use of 0.67 for mass transfer coefficient instead of 0.24) that is cited in Ref. 2 above (EPA, 1987)}
4.	*EPIcode User's Guide, Version 6.0* (Homann, 1996) {User documentation for earlier version, which documents more sample problems than current versions cited in Ref. 1}

2.0 Assessment Summary Results

2.1 Criteria Met

Of the ten general topical quality areas assessed in the gap analysis, two satisfactorily met the criteria. The analysis found that the EPIcode SQA program, in general, met criteria for *Software Classification* and *User Instructions*, Requirements 1 and 7, respectively. The remaining eight topical quality areas were judged either not wholly compliant with the SQA criteria, and/or lacked documentation to confirm compliance. The eight areas that should be addressed for improvement actions are listed in Section 2.2 (Exceptions to Requirements). Details on the evaluation process relative to the requirements and the criteria applied, are found in Section 4.

2.2 Exceptions to Requirements

Exceptions to criteria found for EPIcode 7.0 are listed below in Table 2-1. The requirement is given, the reason the requirement was not met is provided, and action(s) are listed to correct the exceptions. The ten criteria evaluated are those predominantly executed by the software developer. However, it is noted that criteria for SQA Procedures/Plan, Testing, Acceptance Test, Configuration Control, and Error Notification also have requirements for the organization implementing the software. These criteria were assessed in the present evaluation only from the code developer perspective.

Table 2-1 — Summary of Important Exceptions, Reasoning, and Suggested Remediation

No.	Criterion	Reason Not Met	Remedial action(s)
1.	SQA Procedures/Plans (Section 4.2)	SQA Plans and Procedures were not available for the gap analysis.	SQA Plans and Procedures should be developed and made available for review.
2.	Requirements Phase (Section 4.3)	A Software Requirements Document does not exist for review. Thus, it was necessary to infer requirements from draft model description and user guidance documents.	A Software Requirements Document should be prepared and made available for review.
3.	Design Phase (Section 4.4)	A Software Design Document does not exist for review. Thus, it was necessary to infer the intent of the design from draft model description and user guidance documents.	A Software Design Document should be prepared and made available for review.
4	Implementation Phase (Section 4.5)	Documentation to support the implementation is lacking.	A verifiable, written set of SQA plans and procedures including implementation, test case descriptions, and associated criteria related to

No.	Criterion	Reason Not Met	Remedial action(s)
			design should be made available.
5.	Testing Phase (Section 4.6)	A Software Testing Report Document does not exist for review.	A Software Testing Report Document should be prepared and made available for review.
6	Acceptance Test (Section 4.8)	A verifiable, written set of SQA plans and procedures, which would include acceptance testing documentation, is lacking.	Documented acceptance testing should be developed..
7.	Configuration Control (Section 4.9)	A Configuration and Control Document does not exist for review.	A Configuration and Control Document should be prepared and made available for review.
8.	Error Notification (Section 4.10)	An Error Notification and Corrective Action Report do not exist for review.	While a Software Problem Reporting system is apparently in place, written documentation should be provided to the Central Registry for verification of its effectiveness.

2.3 Areas Needing Improvement

The gap analysis identified a few improvements that could be made related to the code.. The recommended upgrades are listed in Table 2-2. These recommended upgrades for EPIcode focus on adding technical capabilities to broaden the use of EPIcode for DSA-type applications and reducing conservatism in the results.

Table 2-2 — Summary of Important Recommendations for EPIcode

No.	Recommendation
1.	Add capability to model dense gas behavior or provide a warning when the release scenario has conditions that might lead to dense gas type of atmospheric transport and dispersion.
2.	Add capability to read from a file of hourly meteorological data over a one-year period, calculate consequences for each hourly entry, and output the 50^{th} and 95^{th} percentile results.
3.	Add capability to use surface roughness input to adjust the rural vertical dispersion coefficient when the input value is greater than 3 cm and less than 100 cm.

2.4 Conclusion Regarding Codes Ability to Meet Intended Function

The EPIcode 7.0 software was evaluated to determine if the software in its current state meets the intended function in a safety analysis context as assessed in this gap analysis. When the code is run for

the intended applications as detailed in the code guidance document, EPIcode *Computer Code Application Guidance for Documented Safety Analysis*, (DOE 2004), it is judged that it will meet its intended function.

3.0 Lessons Learned

Additional opportunities and venues should be sought for training and user qualification on safety analysis software. This is a long-term recommendation for EPIcode and other designated software for the DOE toolbox.

4.0 Detailed Results of the Assessment Process

Ten topical areas or requirements are presented in the assessment as listed in Table 4.0-1.
In the tables that follow, criteria and recommendations are labeled as (1.x, 2.x, ...10.x) with the first value (1., 2., ...) corresponding to the topical area and the second value (x), the sequential table order.

Table 4-0-1— Cross-Reference of Requirements with Subsection and Entry from DOE (2003e)

Subsection (This Report)	Corresponding Entry Table 3-3 from DOE (2003e)	Requirement	ASME NQA-1 2000 Section/Consensus Standards
4.1	1	Software Classification	ASME NQA-1 2000 Section 200
4.2	2	SQA Procedures/Plans	ASME NQA-1 2000 Section 200; IEEE Std. 730, *IEEE Standard for Software Quality Assurance Plans*
4.3	5	Requirements Phase	ASME NQA-1 2000 Section 401; IEEE Standard 830, *Software Requirements Specifications*
4.4	6	Design Phase	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*
4.5	7	Implementation Phase	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*
4.6	8	Testing Phase	ASME NQA-1 2000 Section 404; IEEE Std. 829, *IEEE Standard for Software Test Documentation;* IEEE Standard 1008, *Software Unit*

			Testing
4.7	9	User Instructions	ASME NQA-1 2000 Section 203; IEEE Standard 1063, *IEEE Standard for Software User Documentation*
4.8	10	Acceptance Test	ASME NQA-1 2000 Section 404; IEEE Std. 829, *IEEE Standard for Software Test Documentation;* IEEE Standard 1008, *Software Unit Testing*
4.9	12	Configuration Control	ASME NQA-1 2000 Section 405; ASME NQA-1 2000 Section 406
4.10	13	Error Notification	ASME NQA-1 2000 Section 203

4.1 Topical Area 1 Assessment: Software Classification

This area corresponds to the requirement entitled Software Classification in Table 3-3 of (DOE 2003e).

4.1.1 Criterion Specification and Result

Table 4.1-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Sufficient documentation is provided with software transmittal to make an informed determination of the classification of the software. A user of the EPIcode software for safety analysis applications would be expected to interpret the information on the software in light of the requirements for atmospheric dispersion and consequence analysis discussed in Appendix A to DOE-STD-3009-94 to decide on an appropriate safety classification. For most organizations, the safety class or safety significant classification, or Level B in the classification hierarchy discussed in DOE (2003e), would be selected.

Table 4.1-1 — Subset of Criteria for Software Classification Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
1.1	The code developer must provide sufficient information to allow the user to make an informed decision on the classification of the software.	Yes.	It is concluded that sufficient information is provided with the documentation that is transmitted with the software for the user to make an informed determination of the classification of the software. For most DSA applications, the safety class or safety significant classification, or Level B in the classification hierarchy discussed in DOE (2003e), would be selected,

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			which by definition relate to 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 ➢ Whose results are used to make decisions that could result in death or serious injury or are part of the evaluation in accident analyses.

4.1.2 Sources and Method of Review

Documentation supplied or referenced with the software package and the software developer's partial response to the software information template shown in Appendix A were used as the basis for response to this requirement.

4.1.3 Software Quality-Related Issues or Concerns

There are no SQA issues or concerns relative to this requirement.

4.1.4 Recommendations

No recommendations are provided at this time.

4.2 Topical Area 2 Assessment: SQA Procedures and Plans

This area corresponds to the requirement entitled SQA Procedures and Plans in Table 3-3 of (DOE 2003e).

From the limited information received from the software developer, formal, published SQA procedures and plans were not developed. While it is possible that most elements of a compliant SQA program were

followed in the development of EPIcode 7.0, the lack of written documentation prevents an independent evaluator from making a definitive confirmation.

4.2.1 Criterion Specification and Result

Table 4.2-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
2.1	Procedures/plans for SQA (**SQA Plan**) have identified organizations responsible for performing work; independent reviews, etc.	No.	It is recommended that a SQA plan be developed to provide a framework for configuration control, code maintenance, and support of future upgrades.
2.2	Procedures/plans for SQA (**SQA Plan**) have identified software engineering methods.	No.	See Criterion 2.1 summary remarks.
2.3	Procedures/plans for SQA (**SQA Plan**) have identified documentation to be required as part of program.	No.	See Criterion 2.1 summary remarks.
2.4	Procedures/plans for SQA (**SQA Plan**) have identified standards, conventions, techniques, and/or methodologies which shall be used to guide the software development, methods to ensure compliance with the same.	No.	See Criterion 2.1 summary remarks.
2.5	Procedures/plans for SQA (**SQA Plan**) have identified software reviews and schedule.	No.	See Criterion 2.1 summary remarks.
2.6	Procedures/plans for SQA (**SQA Plan**) have identified methods for error reporting and corrective actions.	No.	See Criterion 2.1 summary remarks.

4.2.2 Sources and Method of Review

Documentation supplied or referenced with the software package and the software developer's partial response to the software information template shown in Appendix A were used as the basis for response to this requirement.

4.2.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures for EPIcode should be addressed.

4.2.4 Recommendations

Recommendations related to this topical area are provided as follows:

- It is recommended that a SQA plan be developed to provide a framework for configuration control, code maintenance, and support of future upgrades.

4.3 Topical Area 3 Assessment: Requirements Phase

This area corresponds to the requirement entitled Requirements Phase in Table 3-3 of (DOE 2003e).

4.3.1 Criterion Specification and Result

Table 4.3-1 lists the subset of criteria reviewed for this topical area and summarizes the findings

Table 4.3-1 — Subset of Criteria for Requirements Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
3.1	Software requirements for the subject software have been established.	Yes.	Implicitly fulfilled. The EPIcode program was developed to provide emergency response personnel and emergency planners with a software tool to evaluate downwind concentrations from the atmospheric release of toxic substances. Specifically, the online user's documentation states that EPIcode was designed to produce calculated radii of the vulnerable zones that are in exact agreement with the EPA document, "Technical Guidance for Hazards Analysis -Emergency Planning for Extremely Hazardous Substances" (EPA, 1987).
3.2	Software requirements are specified, documented, reviewed and approved.	No.	A verifiable, written set of SQA plans and procedures, which would include software requirements, is lacking for EPIcode.
3.3	Requirements define the functions to be performed by the software and provide detail and information necessary to design the software.	Yes.	EPIcode strictly follows the well-established Gaussian model. EPIcode uses no "black-box" techniques. All algorithms are presented and fully referenced in the onboard Software User Documentation. EPIcode uses the same algorithms and methodologies outlined in

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			EPA document titled "Technical Guidance for Hazards Analysis - Emergency Planning for Extremely Hazardous Substances," U.S. Environmental Protection Agency, Federal Emergency Management Agency, and U.S. Department of Transportation, December 1987.
3.4	A **Software Requirements Document**, or equivalent defines requirements for functionality, performance, design inputs, design constraints, installation considerations, operating systems (if applicable), and external interfaces necessary to design the software.	Partial.	As stated above, the online user's documentation implicitly states requirements. The user's documentation also addresses, at least partially, installation, operating systems and design inputs.
3.5	Acceptance criteria are established in the software requirements documentation for each of the identified requirements.	Partial.	According to the online user's documentation, "EPIcode output always contains all of the input assumptions, and the calculated radii of the vulnerable zones are in exact agreement with the EPA document. This demonstrates correct implementation of the basic Gaussian algorithms contained in the EPA document."

Additional Detail The Gaussian model is the basic workhorse for atmospheric dispersion calculations and has found its way into most governmental guidebooks. The Gaussian model has also been used and accepted by the Environmental Protection Agency (EPA, 1978). The adequacy of this model for making initial dispersion estimates or worst-case safety analyses has been tested and verified for many years.

4.3.2 Sources and Method of Review

Documentation supplied or referenced with the software package and the software developer's partial response to the software information template shown in Appendix A were used as the basis for response to this requirement.

4.3.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which would include written software requirements, for EPIcode should be addressed.

4.3.4 Recommendations

Recommendations related to this topical area are provided as follows:

- Documented software requirements will be needed for EPIcode to meet all prerequisites for the DOE toolbox.

4.4 Topical Area 4 Assessment: Design Phase

This area corresponds to the requirement entitled Design Phase in Table 3-3 of (DOE 2003e).

4.4.1 Criterion Specification and Result

Table 4.4-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.1	The software design was developed, documented, reviewed and controlled.	Uncertain.	Because SQA plans and procedures from the software developer are not available, a thorough evaluation was not possible.
4.2	Code developer(s) 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.	Partial.	Design may be inferred from final software product, but design document was not made available for review.
4.3	The following design should be present and documented: specification of interfaces, overall structure (control and data flow) and the reduction of the overall structure into physical solutions (algorithms, equations, control logic, and data structures).	Uncertain.	See Criterion 4.1 summary remarks.
4.4	The following design should be present and documented: 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.	Uncertain.	See Criterion 4.1 summary remarks.
4.5	The following design should be present and documented: evidence of measures to mitigate the consequences of software design problems. These potential problems include external and internal abnormal conditions and	Not applicable to non-process, instrumentation and control software.	None.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	events that can affect the computer program.		
4.6	A Software Design Document, or equivalent, is available and contains a description of the major components of the software design as they relate to the software requirements.	No.	A verifiable, written set of SQA plans and procedures, which would include software design documentation, is lacking for EPIcode.
4.7	A Software Design Document, or equivalent, is available and contains 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.	No.	See Criterion 4.6 summary remarks.
4.8	A Software Design Document, or equivalent, is available and contains a description of the allowable or prescribed ranges for inputs and outputs.	Yes.	The EPIcode user documentation contains this information.
4.9	A Software Design Document, or equivalent, is available and contains the design described in a manner that can be translated into code.	No.	See Criterion 4.6 summary remarks.
4.10	A Software Design Document, or equivalent, is available and contains 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.	No.	See Criterion 4.6 summary remarks.
4.11	The organization responsible for the design identified and documented the particular verification methods to be used and assured that an Independent Review was performed and documented. This review evaluated the technical adequacy of the design approach; assured internal completeness, consistency, clarity, and correctness of the software design; and verified that the software design is traceable to the requirements.	Uncertain.	While some elements of this criterion may have been met informally, there is no written documentation that allows confirmation.
4.12	The organization responsible for the design assured that the test results adequately demonstrated the	Uncertain.	See Criterion 4.1 summary remarks.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	requirements were met.		
4.13	The Independent Review was performed by competent individual(s) other than those who developed and documented the original design, but who may have been from the same organization.	Uncertain.	While some elements of this criterion may have been met informally, there is no written documentation that allows confirmation.
4.14	The results of the Independent Review are documented with the identification of the verifier indicated.	Uncertain.	See Criterion 4.1 summary remarks.
4.15	If review alone was not adequate to determine if requirements are met, alternate calculations were used, or tests were developed and integrated into the appropriate activities of the software development cycle.	Uncertain.	See Criterion 4.1 summary remarks.
4.16	Software design documentation was completed prior to finalizing the Independent Review.	No.	See Criterion 4.6 summary remarks.
4.17	The extent of the Independent Review and the methods chosen are shown to be a function of: ➢ The importance to safety, ➢ The complexity of the software, ➢ The degree of standardization, and ➢ The similarity with previously proven software.	Uncertain.	See Criterion 4.1 summary remarks.

4.4.2 Sources and Method of Review

Documentation supplied or referenced with the software package and the software developer's partial response to the software information template shown in Appendix A were used as the basis for response to this requirement.

4.4.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which would include software design documentation, for EPIcode should be addressed.

4.4.4 Recommendations

Recommendations related to this topical area are provided as follows:

- Documented software design will be needed for EPIcode to meet all prerequisites for the DOE toolbox.

4.5 Topical Area 5 Assessment: Implementation Phase

This area corresponds to the requirement entitled Implementation Phase in Table 3-3 of (DOE 2003e).

4.5.1 Criterion Specification and Result

Table 4.5-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
5.1	The implementation process resulted in software products such as computer program listings and instructions for computer program use.	Partial.	Elements of this criterion may be inferred from documentation and the final software product, however, the implementation process has not been formally documented..
5.2	Implemented software was analyzed to identify and correct errors.	Uncertain.	Because SQA plans and procedures from the software developer are not available, a thorough evaluation was not possible.
5.3	The source code finalized during verification (this phase) was placed under configuration control.	Uncertain.	See Criterion 5.2 summary remarks.
5.4	Documentation during verification included a copy of the software, test case description and associated criteria that are traceable to the software requirements and design documentation.	No.	A verifiable, written set of SQA plans and procedures, which would include test case descriptions as well as software requirements and design documentation, is lacking for EPIcode.

4.5.2 Sources and Method of Review

Documentation supplied or referenced with the software package and the software developer's partial response to the software information template shown in Appendix A were used as the basis for response to this requirement.

4.5.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which would include test case descriptions as well as software requirements and design documentation, for EPIcode should be addressed.

4.5.4 Recommendations

Recommendations related to this topical area are provided as follows:

- A documented implementation process will be needed for EPIcode to meet all prerequisites for the DOE toolbox.

4.6 Topical Area 6 Assessment: Testing Phase

This area corresponds to the requirement entitled Testing Phase in Table 3-3 of (DOE 2003e).

4.6.1 Criterion Specification and Result

Table 4.6-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
6.1	The software was validated by executing test cases.	Yes.	EPIcode uses the same algorithms and methodologies outlined in EPA document titled "Technical Guidance for Hazards Analysis -Emergency Planning for Extremely Hazardous Substances," U.S. Environmental Protection Agency, Federal Emergency Management Agency, and U.S. Department of Transportation, December 1987. According to the code developer, EPIcode output always contains all of the input assumptions, and the calculated radii of the vulnerable zones are in exact agreement with the EPA document. This demonstrates correct implementation of the basic Gaussian algorithms contained in the EPA document.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
6.2	Testing demonstrated the capability of the software to produce valid results for test cases encompassing the range of permitted usage defined by the program documentation. Such activities provide evidence to ensure that the software adequately and correctly performed all intended functions.	Partial.	The EPIcode user's guide contains 15 example case studies that show how EPIcode can be applied to a wide range of chemical accident scenarios. In nearly half of these examples, the EPIcode results are compared against field measurements or the output of other computer codes. Documentation is lacking, however, to confirm all aspects of this requirement.
6.3	Testing demonstrated that the computer program properly handles abnormal conditions and events as well as credible failures.	Uncertain.	Because SQA plans and procedures from the software developer are not available, a thorough evaluation was not possible.
6.4	Testing demonstrated that the computer program does not perform adverse unintended functions.	Uncertain.	See Criterion 6.3 summary remarks.
6.5	Test Phase activities were performed to assure adherence to requirements, and to assure that the software produces correct results for the test case specified. Acceptable methods for evaluating adequacy of software test case results included: (1) analysis with computer assistance; (2) other validated computer programs; (3) experiments and tests; (4) standard problems with known solutions; (5) confirmed published data and correlations.	Partial.	See Criterion 6.1 summary remarks.
6.6	Test Phase documentation includes test procedures or plans and the results of the execution of test cases. The test results documentation demonstrates successful completion of all test cases or the resolution of unsuccessful test cases and provides direct traceability between the test results and specified software requirements.	No.	A verifiable, written set of SQA plans and procedures, which would include test phase documentation, is lacking for EPIcode.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
6.7	Test procedures or plans specify the following, as applicable: (1) required tests and test sequence, (2) required range of input parameters, (3) identification of the stages at which testing is required, (4) requirements for testing logic branches, (5) requirements for hardware integration, (6) anticipated output values, (7) acceptance criteria, (8) reports, records, standard formatting, and conventions, (9) identification of operating environment, support software, software tools or system software, hardware operating system(s) and/or limitations.	No.	See Criterion 6.6 summary remarks.

4.6.2 Sources and Method of Review

Documentation supplied or referenced with the software package and the software developer's partial response to the software information template shown in Appendix A were used as the basis for response to this requirement.

4.6.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which includes test reports, for EPIcode should be addressed.

4.6.4 Recommendations

Recommendations related to this topical area are provided as follows:

- It is recommended that benchmark comparisons and validation cases be formally documented (current documentation is in the form of sample case illustrations in the user's manual for the previous version of the code).
- It is recommended that formal test report documentation be established for future upgrades to the code.

4.7 Topical Area 7 Assessment: User Instructions

This area corresponds to the requirement entitled User Instructions in Table 3-3 of (DOE 2003e).

4.7.1 Criterion Specification and Result

Table 4.7-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
7.1	A description of the model is documented and made available to users.	Yes.	EPIcode strictly follows the well-established Gaussian model. EPIcode uses no "black-box" techniques. All algorithms are presented and fully referenced in the onboard Software User Documentation.
7.2	User's manual or guide describes software and hardware limitations and identifies includes approved operating systems (for cases where source code is provided, applicable compilers should be noted).	Yes.	(EPIcode, 2003)
7.3	User's manual or guide includes description of the user's interaction with the software.	Yes.	(EPIcode, 2003)
7.4	User's manual or guide includes a description of any required training necessary to use the software.	Not Applicable.	The user's manual does not state the need for any required general training. Formal training, while recommended, is not required.
7.5	User's manual or guide includes input and output specifications.	Yes.	(EPIcode, 2003)
7.6	User's manual or guide includes a description of user messages initiated as a result of improper input and how the user can respond.	Partial.	The user's documentation content is too brief on potential user-induced software problems. Common errors and warning messages could be included with suggested solutions. For some parameters, EPIcode will only allow values within a certain range that is identified in the dialog box that prompts the user to enter input. If the user attempts to input data outside the range, EPIcode will set the value to either the minimum or maximum value of the allowable range as appropriate for the attempted input. It is recommended that a warning message be given when the release

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			scenario has conditions that might lead to dense gas type of atmospheric transport and dispersion.
7.7	User's manual or guide includes information for obtaining user and maintenance support.	Yes.	(EPIcode, 2003)

4.7.2 Sources and Method of Review

Documentation supplied or referenced with the software package and the software developer's partial response to the software information template shown in Appendix A were used as the basis for response to this requirement.

4.7.3 Software Quality-Related Issues or Concerns

User instruction documentation is good. No substantive issues or concerns have surfaced.

4.7.4 Recommendations

Recommendations related to this topical area are as follows:

- The user's documentation content is too brief on potential user-induced software problems. Common errors and warning messages could be included with suggested solutions. Additionally, it is recommended that a warning message be given when the release scenario has conditions that might lead to dense gas type of atmospheric transport and dispersion.

4.8 Topical Area 8 Assessment: Acceptance Test

This area corresponds to the requirement entitled Acceptance Test Table 3-3 of (DOE 2003e). During this phase of the software development, the software becomes part of a system incorporating applicable software components, hardware, and data and is accepted for use. Much of this testing is the burden of the user organization, but the developing organization shoulders some responsibility.

4.8.1 Criterion Specification and Result

Table 4.8-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
8.1	To the extent applicable to the developer, acceptance testing includes a comprehensive test in the operating environment(s).	Uncertain.	A verifiable, written set of SQA plans and procedures, which would include acceptance testing documentation, is lacking for EPIcode.
8.2	To the extent applicable to the developer acceptance testing was performed prior to approval of the computer program for use.	Uncertain.	See Criterion 8.1 summary remarks.
8.3	The acceptance testing comprehensively evaluates software performance against specified software requirements. To the extent applicable to the developer software validation was performed to ensure that the installed software product satisfies the specified software requirements.	Yes.	EPIcode has an automatic QC check to ensure correct installation and operation of the software. Selection of this option automatically runs all of the EPIcode Release Examples/Case Studies (see onboard Documentation), to verify correct EPIcode operation. Each Example is executed with all parameters/defaults set to the exact values stated in the documentation. The resulting output is compared with the documented results. This ensures that EPIcode has been installed and is operating correctly.
8.4	Acceptance testing documentation includes results of the execution of test cases for system installation and integration, user instructions (Refer to Requirement 7 above), and documentation of the acceptance of the software for operational use.	Yes.	See above.

4.8.2 Sources and Method of Review

Documentation supplied or referenced with the software package and the software developer's partial response to the software information template shown in Appendix A were used as the basis for response to this requirement.

4.8.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which include acceptance testing documentation for EPIcode should be addressed.

4.8.4 Recommendations

Recommendations related to this topical area are provided as follows:

- A documented implementation process will be needed for EPIcode to meet all prerequisites for the DOE toolbox.

4.9 Topical Area 9 Assessment: Configuration Control

This area corresponds to the requirement entitled Configuration Control in Table 3-3 of (DOE 2003e).

4.9.1 Criterion Specification and Result

Table 4.9-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
9.1	For the developer, the methods used to control, uniquely identify, describe, and document the configuration of each version or update of a computer program (for example, source, object, back-up files) and its related documentation (for example, software design requirements, instructions for computer program use, test plans, and results) are described in implementing procedures.	Uncertain.	Because a written set of SQA plans and procedures, which would include configuration control procedures, is lacking for EPIcode, a thorough evaluation was not possible.
9.2	Implementing procedures meet applicable criteria for configuration identification, change control and configuration status accounting.	Uncertain.	See Criterion 9.1 summary remarks.

4.9.2 Sources and Method of Review

Documentation supplied or referenced with the software package and the software developer's partial response to the software information template shown in Appendix A were used as the basis for response to this requirement.

4.9.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which include configuration control documentation, for EPIcode should be addressed.

4.9.4 Recommendations

Recommendations related to this topical area are provided as follows:

- A documented configuration control process will be needed for EPIcode to meet all prerequisites for the DOE toolbox.

4.10 Topical Area 10 Assessment: Error Impact

This area corresponds to the requirement entitled Error Impact in Table 3-3 of (DOE 2003e).

4.10.1 Criterion Specification and Result

Table 4.10-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results

Criterion Number	Criterion Specification	Compliant?	Summary Remarks
10.1	The developing organization's problem reporting and corrective action process addresses the appropriate requirements of its corrective action system and is documented in implementing procedures.	Partial.	Homann Associates, Inc. controls the error notification and corrective actions process. No written confirmation of a documented process.
10.2	The process for evaluating, and documenting whether a reported problem is an error is documented and implemented.	Partial.	No written confirmation of a documented process. Only given an example of the process as it relates to a recent incident and corrective action: Revised EPA Evaporation model in EPIcode. Homann Associates was notified by LLNL NARAC that the EPA Evaporation model had been revised. Homann Associates reviewed/revised the Evaporation model per EPA document "Risk Management Program Guidance for Offsite Consequence Analysis," United States Environmental Protection Agency, EPA 550-B-99-009, April 1999. Appendix D – Technical Background, pg. D-2.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			The mass transfer coefficient of water is now assumed to be 0.67; The value of 0.67 is based on the Donald MacKay and Ronald S. Matsugu,

Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results (Continued)

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			"Evaporation Rates of Liquid Hydrocarbon Spills on Land and Water," Canadian Journal of Chemical Engineering, August 1973, p. 434. The value of the factor that includes conversion factors, mass coefficient for water, and the molecular weight of water to the one-third power, originally 0.106, is now 0.284. The net result is an evaporation rate that is 2.68 times greater than previous EPIcode versions.
10.3	The process for disposition of the problem reports, including notification to the originator of the results of the evaluation, is documented and implemented.	Uncertain.	Because SQA plans and procedures from the software developer are not available, a thorough evaluation was not possible.
10.4	A documented process provides guidance on determining how identified errors relate to appropriate software engineering elements and is implemented.	Uncertain.	See Criterion 10.3 summary remarks.
10.5	The process is documented and implemented for determining how an error impacts past and present use of the computer program.	Uncertain.	See Criterion 10.3 summary remarks.
10.6	The process is documented and implemented for determining how an error and resulting corrective action impacts previous development activities.	Uncertain.	See Criterion 10.3 summary remarks.
10.7	The process is documented and implemented describing how the users are notified of an identified error, its impact; and how to avoid the error, pending implementation of corrective actions.	Uncertain.	See Criterion 10.3 summary remarks.

4.10.2 Sources and Method of Review

Documentation supplied or referenced with the software package and the software developer's partial response to the software information template shown in Appendix A were used as the basis for response to this requirement.

4.10.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which includes error notification and corrective action report, for EPIcode should be addressed.

4.10.4 Recommendations

Recommendations related to this topical area are provided as follows:

- A documented error notification and corrective action process will be needed for EPIcode to meet all prerequisites for the DOE toolbox.

4.11 Training Program Assessment

The software developer's does not have a published training program available for review. It is suggested that training on EPIcode be given at the Energy Facility Contractors Group (EFCOG) conferences. The winter session is during the Safety Basis Subgroup meeting and the summer session is the larger Safety Analysis Working Group, and historically has included training workshops.

4.12 Software Improvements

The EPIcode software was recently upgraded with the issuance of Version 7.0 in September of 2003. EPIcode Version 7.1 is currently in alpha test. This version contains new chemical warfare and biological warfare features. It allows the user to select new output options included time integrated concentration and inhaled dose. A dense gas warning feature is added. A dense gas capability is in development. Additional documentation has been added, inclusive of case studies and validation examples.

It is estimated that a concentrated program to upgrade the SQA pedigree of EPIcode to be compliant with the ten criteria discussed here would require fourteen to sixteen full-time equivalent (FTE)-months. Technical review of the chemical databases associated with this software is assumed to have been performed, and is not included in the level-of-effort estimate.

5.0 Conclusion

The gap analysis for Version 7.0 of the EPIcode software, based on a set of requirements and criteria compliant with NQA-1, has been completed. Of the ten SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), two requirements are met at acceptable level, i.e., *Classification* (1) and *User Instructions* (7). Improvement actions are recommended for EPIcode to fully meet the remaining eight requirements. This evaluation outcome is deemed acceptable because: (1) EPIcode 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 EPIcode is limited to those analytic applications for which the software is intended.

Suggested remedial actions for this software would warrant upgrading software documents. The complete list of revised 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 Manual.

Overall, it was determined that the EPIcode 7.0 does meet its intended function for use in supporting documented safety analysis. However, as with all safety-related software, users should be aware of current limitations and capabilities of the software for supporting safety analysis. Informed use of the code can be assisted by appropriate use of current EPIcode documentation and the EPIcode guidance report for DOE safety analysts, *EPIcode Computer Code Application Guidance for Documented Safety Analysis*, (DOE, 2004). Furthermore, while SQA improvement actions are recommended for EPIcode, no evidence has been found of programming, logic, or other types of software errors in EPIcode 7.0 that have led to non-conservatisms in nuclear facility operations, or in the identification of facility controls.

Recommendations are given in Section 2.3 of this document for upgrading the capabilities of EPIcode, focusing on added technical capabilities to broaden the use of EPIcode for DSA-type applications and reducing conservatism in the results.

6.0 Acronyms and Definitions

ACRONYMS:

ASME	American Society of Mechanical Engineers
CD	Compliance Decision
CFR	Code of Federal Regulations
DNFSB	Defense Nuclear Facilities Safety Board
DOE	Department of Energy
DSA	Documented Safety Analysis
EFCOG	Energy Facility Contractors Group
EH	DOE Office of Environment, Safety and Health
EM	DOE Office of Environmental Management
IEEE	Institute of Electrical and Electronics Engineers
IP	Implementation Plan
QAP	Quality Assurance Program (alternatively, Plan)
SQA	Software Quality Assurance
V&V	Verification and Validation
WSRC	Westinghouse Savannah River Company

DEFINITIONS:

The following definitions are taken from the Implementation Plan. References in brackets following definitions indicate the original source, when not the Implementation Plan.

Acceptance Testing — [NQA-1] The process of exercising or evaluating a system or system component by manual or automated means to ensure that it satisfies the specified requirements and to identify differences between expected and actual results in the operating environment.

Central Registry — An organization designated to be responsible for the storage, control, and long-term maintenance of the Department's safety analysis "toolbox codes." The central registry may also perform this function for other codes if the Department determines that this is appropriate.

Classification (Level of Software) — Determination of the level of software quality assurance associated with a computer code commensurate with the importance of the software application. For the toolbox codes, classification level is determined as described in Appendix A of: "Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes".

Computer Code — A set of instructions that can be interpreted and acted upon by a programmable digital computer (also referred to as a module or a computer program).

Configuration Item — A collection of hardware or software elements treated as a unit for the purpose of configuration control. [NQA-1]

Configuration Management —The process that controls the activities, and interfaces, among design, construction, procurement, training, licensing, operations, and maintenance to ensure that the configuration of the facility is established, approved and maintained. (Software specific): The process of identifying and defining the configuration items in a system (i.e., software and hardware), controlling the release and change of these items throughout the system's life cycle, and recording and reporting the status of configuration items and change requests. [NQA-1]

Data Library — A data file for use with an executable code that is created and maintained by the controlling organization and is not intended for modification by the user.

Dedication (of Software) — The evaluation of software not developed under utilizing organization existing QA plans and procedures (or not developed under NQA-1 standards). The evaluation determines and asserts the software's compliance with NQA-1 quality standards and its readiness for use in specific applications. (Typically applies to commercially available software.) The utilizing organization reviews the intended software application sufficiently to determine the critical functions that provide evidence of the software's suitability for use. Once the critical functions have been established, methods are defined to verify critical function adequacy and provide verifiable acceptance criteria. Acceptable dedication methods are implemented and required documentation is prepared.

Design Requirements — Description of the methodology, assumptions, functional requirements, and technical requirements for a software system.

Discrepancy — The failure of software to perform according to its documentation.

Error — A condition deviating from an established base line, including deviations from the current approved computer program and its baseline requirements. [NQA-1]

Executable Code — The user form of a computer code. For programs written in a compilable programming language, the compiled and loaded program. For programs written in an interpretable programming language, the source code.

Firmware — The combination of a hardware device and computer instructions and data that reside as read-only software on that device. [IEEE Standard 610.12-1990]

Gap Analysis — Evaluation of the Software Quality Assurance attributes of specific computer software against identified criteria.

Independent Verification and Validation (IV&V) — Verification and validation performed by an organization that is technically, managerially, and financially independent of the development organization.

Nuclear Facility — A reactor or a nonreactor nuclear facility where an activity is conducted for or on behalf of DOE and includes any related area, structure, facility, or activity to the extent necessary to ensure proper implementation of the requirements established by 10 CFR 830. [10 CFR 830]

Object Code — A computer code in its compiled form. This applies only to programs written in a compilable programming language.

Operating Environment — A collection of software, firmware, and hardware elements that provide for the execution of computer programs. [NQA-1]

Safety Analysis and Design Software — Computer software that is not part of a structure, system, or component (SSC) but is used in the safety classification, design, and analysis of nuclear facilities to ensure proper accident analysis of nuclear facilities; proper analysis and design of safety SSCs; and proper identification, maintenance, and operation of safety SSCs.

Safety-Class Structures, Systems, and Components (SC SSCs) — SSCs, including portions of process systems, whose preventive and mitigative function is necessary to limit radioactive hazardous material exposure to the public, as determined from the safety analyses. [10 CFR 830]

Safety-Significant Structures, Systems, and Components (SS SSCs) — SSCs which are not designated as safety-class SSCs, but whose preventive or mitigative function is a major contributor to defense in depth and/or worker safety as determined from safety analyses. [10 CFR 830] As a general rule of thumb, SS SSC designations based on worker safety are limited to those systems, structures, or components whose failure is estimated to result in prompt worker fatalities, serious injuries, or significant radiological or chemical exposure to workers. The term serious injuries, as used in this definition, refers to medical treatment for immediately life-threatening or permanently disabling injuries (e.g., loss of eye, loss of limb). The general rule of thumb cited above is neither an evaluation guideline nor a quantitative criterion. It represents a lower threshold of concern for which an SS SSC designation may be warranted. Estimates of worker consequences for the purpose of SS

SSC designation are not intended to require detailed analytical modeling. Consideration should be based on engineering judgment of possible effects and the potential added value of SS SSC designation. [DOE G 420.1-1]

Safety Software — Includes both safety system software and safety analysis and design software.

Safety Structures, Systems, and Components (SSCs) — The set of safety-class SSCs and safety-significant SSCs for a given facility. [10 CFR 830]

Safety System Software — Computer software and firmware that performs a safety system function as part of a structure, system, or component (SSC) that has been functionally classified as Safety Class (SC) or Safety Significant (SS). This also includes computer software such as human-machine interface software, network interface software, programmable logic controller (PLC) programming language software, and safety management databases that are not part of an SSC but whose operation or malfunction can directly affect SS and SC SSC function.

Software — Computer programs, operating systems, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. [IEEE Std. 610.12-1990]

Software Design Verification —The process of determining if the product of the software design activity fulfills the software design requirements. [NQA-1]

Software Engineering — The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software; also: the study of these applications. [NQA-1]

Source Code — A computer code in its originally coded form, typically in text file format. For programs written in a compilable programming language, the uncompiled program.

System Software —Software designed to enable the operation and maintenance of a computer system and its associated computer programs. [NQA-1]

Test Case —A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement. [NQA-1]

Test Case Input — Input data for a test case used to verify a modification to a module or a data library.

Test Plan (Procedure) —A document that describes the approach to be followed for testing a system or component. Typical contents identify the items to be tested, tasks to be performed, and responsibilities for the testing activities. [NQA-1]

Testing —An element of verification for the determination of the capability of an item to meet specified requirements by subjecting the item to a set of physical, chemical, environmental, or operating conditions. [NQA-1]

Testing (Software) —The process of

(a) Operating a system (i.e., software and hardware) or system component under specified conditions;

(b) Observing and recording the results; and

(c) Making an evaluation of some aspect of the system (i.e., software and hardware) or system component; in order to verify that it satisfies specified requirements and to identify errors. [NQA-1]

Toolbox Codes — A small number of standard computer models (codes) supporting DOE safety analysis, having widespread use, and of appropriate qualification that are maintained, managed, and distributed by a central source. Toolbox codes meet minimum quality assurance criteria. They may be applied to support 10 CFR 830 DSAs provided the application domain and input parameters are valid. In addition to public domain software, commercial or proprietary software may also be considered. In addition to safety analysis software, design codes may also be included if there is a benefit to maintain centralized control of the codes [modified from DOE N 411.1].

User Manual — A document that presents the information necessary to employ a system or component to obtain desired results. Typically described are system or component capabilities, limitations, options, permitted inputs, expected outputs, possible error messages, and special instructions. Note: A user manual is distinguished from an operator manual when a distinction is made between those who operate a computer system (mounting tapes, etc.) and those who use the system for its intended purpose. Syn: User Guide. [IEEE 610-12]

Validation – 1. The process of testing a computer program and evaluating the results to ensure compliance with specified requirements [ANSI/ANS-10.4-1987].
2.The process of determining the degree to which a model is an accurate representation of the real-world from the perspective of the intended uses of the model [Department of Defense Directive 5000.59, *DoD Modeling and Simulation (M&S) Management*].

Verification – 1. The process of evaluating the products of a software development phase to provide assurance that they meet the requirements defined for them by the previous phase [ANSI/ANS-10.4-1987].
2. The process of determining that a model implementation accurately represents the developer's conceptual description and specifications [Department of Defense Directive 5000.59, *DoD Modeling and Simulation (M&S) Management*].

7.0 References

CFR, Code of Federal Regulations (10 CFR 830). 10 CFR 830, Nuclear Safety Management Rule.

DNFSB, Defense Nuclear Facilities Safety Board, (2000). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, Technical Report DNFSB/TECH-25, (January 2000).

DNFSB, Defense Nuclear Facilities Safety Board, (2002). *Recommendation 2002-1, Quality Assurance for Safety-Related Software* , (September 2002).

DOE, U.S. Department of Energy (2000a). *Appendix A, Evaluation Guideline,* DOE-STD-3009-94, *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports* (January 2000).

DOE, U.S. Department of Energy (2000b). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, DOE Response to TECH-25, Letter and Report, (October 2000).

DOE, U.S. Department of Energy (2002). *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports*, DOE-HDBK-3010-94, Change Notice 2 (April 2002).

DOE, U.S. Department of Energy (2003a). *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).

DOE, U.S. Department of Energy (2003b). *Designation of Initial Safety Analysis Toolbox Codes*, Letter, (March 28, 2003).

DOE, U.S. Department of Energy (2003c). *Assessment Criteria and Guidelines for Determining the Adequacy of Software Used in the Safety Analysis and Design of Defense Nuclear Facilities*, Report, CRAD-4.2.4-1, Rev 0, (August 27 2003).

DOE, U.S. Department of Energy (2003d). *Software Quality assurance Improvement Plan: Format and Content For Code Guidance Reports*, Revision A (draft), Report, (August 2003).

DOE, U.S. Department of Energy (2003e). *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes*, Revision 1, (November 2003).

DOE, U.S. Department of Energy (2004). EPIcode Computer Code Application Guidance for Documented Safety Analysis, (May 2004).

EPA, U.S. Environmental Protection Agency (1987). *Technical Guidance for Hazards Analysis: Emergency Planning for Extremely Hazardous Substances*, U.S. Environmental Protection Agency, Federal Emergency Management Agency, U.S. Department of Transportation (December 1987).

EPA, U.S. Environmental Protection Agency (1999). *Risk Management Program Guidance for Offsite Consequences*, EPA550-B-99-009, Appendix D – Technical Background (April, 1999).

EPIcode (2003). *EPIcode Version 7.0 User Documentation*, Online Help distributed with software package, Homann Associates, Inc. (September 2003).

S. G. Homann (1996), *EPIcode User's Guide, Version 6.0*, Homann Associates, Inc.

Appendices

Appendix	Subject
A	Software Information Template

APPENDIX A.— SOFTWARE INFORMATION TEMPLATE

Information Form

Development and Maintenance of Designated Safety Analysis Toolbox Codes

The following summary information in Table 2 should be completed to the level that is meaningful - enter N/A if not applicable. *(Note: This information is provided to give the reader of this Gap report, an idea of the information requested to complete the Gap analysis for EPIcode. Detailed information in response was not filled in. See Section 1.6. Instead, the contacts and the Gap authors used the form as a guide for continual discussion throughout the Gap analysis for EPIcode.)*

Table 2. Summary Description of Subject Software

Table 2. Summary Description of Subject Software	
Type	**Specific Information**
Code Name	
Version of the Code	
Developing Organization and Sponsor Information	
Auxiliary Codes	
Software Platform/Portability	
Coding and Computer(s)	
Technical Support Point of Contact	
Code Procurement Point of Contact	
Code Package Label/Title	
Contributing Organization(s)	

Table 2. Summary Description of Subject Software	
Type	**Specific Information**
Recommended Documentation - Supplied with Code Transmittal upon Distribution or Otherwise Available	1. 2. 3. 4. 5.
Input Data/Parameter Requirements	
Summary of Output	
Nature of Problem Addressed by Software	
Significant Strengths of Software	
Known Restrictions or Limitations	
Preprocessing (set-up) time for Typical Safety Analysis Calculation	
Execution Time	
Computer Hardware Requirements	
Computer Software Requirements	
Other Versions Available	

Table 2. Summary Description of Subject Software	
Type	Specific Information

Table 3. Point of Contact for Form Completion

Individual(s) completing this information form: Name: Organization: Telephone: Email: Fax:	

1. Software Quality Assurance Plan

The software quality assurance plan for your software may be either a standalone document, or embedded in other documents, related procedures, QA assessment reports, test reports, problem reports, corrective actions, supplier control, and training package.

1.a For this software, identify the governing Software Quality Assurance Plan (SQAP)? [Please submit a PDF of the SQAP, or send hard copy of the SQAP[2]]

1.b What software quality assurance industry standards are met by the SQAP?

1.c What federal agency standards were used, if any, from the sponsoring organization?

1.d Has the SQAP been revised since the current version of the Subject Software was released? If so, what was the impact to the subject software?

1.e Is the SQAP proceduralized in your organization? If so, please list the primary procedures that provide guidance.

Guidance for SQA Plans:

Requirement 2 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a)
ASME NQA-1 2000 Section 200

[2] Notify Kevin O'Kula of your intent to send hard copies of requested reports and shipping will be arranged.

IEEE Standard 730, *IEEE Standard for Software Quality Assurance Plans.*
IEEE Standard 730.1, *IEEE Guide for Software Quality Assurance Planning.*

2. Software Requirements Description

The software requirements description (SRD) should contain functional and performance requirements for the subject software. It may be contained in a standalone document or embedded in another document, and should address functionality, performance, design constraints, attributes and external interfaces.

2.a **For this software, was a software requirements description documented with the software sponsor?** [If available, please submit a PDF of the Software Requirements Description, or include hard copy with transmittal of SQAP]

2.b **If a SRD was not prepared, are there written communications that indicate agreement on requirements for the software? Please list other sources of this information if it is not available in one document.**

Guidance for Software Requirements Documentation:

Requirement 5 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 401
IEEE Standard 830, *Software Requirements Specifications*

3. Software Design Documentation

The software design documentation (SDD) depicts how the software is structured to satisfy the requirements in the software requirements description. It should be defined and maintained to ensure that software will serve its intended function. The SDD for the subject software may be contained in a standalone document or embedded in another document.

The SDD should provide the following:

- Description of the major components of the software design as they relate to the software requirements,
- Technical description of the software with respect to the theoretical basis, mathematical model, control flow, data flow, control logic, and data structure,
- Description of the allowable or prescribed ranges of inputs and outputs,
- Design described in a manner suitable for translating into computer coding, and
- Computer program listings (or suitable references).

3.a For the subject software, was a software design document prepared, or were its constituents parts covered elsewhere? [If available, please submit a PDF of the Software Design Document, or include hard copy with transmittal of SQAP]

3.b If the intent of the SDD information is satisfied in other documents, provide the appropriate references (document number, section, and page number).

Guidance for Software Design Documentation:

Requirement 6 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
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*
IEEE Standard 1012, *IEEE Standard for Software Verification and Validation;*
IEEE Standard 1012a, *IEEE Standard for Software Verification and Validation - Supplement to 1012*

4. Software User Documentation

Software User Documentation is necessary to assist the user in installing, operating, managing, and maintaining the software, and to ensure that the software satisfies user requirements. At minimum, the documentation should describe:

- The user's interaction with the software
- Any required training
- Input and output specifications and formats, options
- Software limitations
- Error message identification and description, including suggested corrective actions to be taken to correct those errors, and
- Other essential information for using the software.

4.a For the subject software, has Software User Documentation been prepared, or are its constituents parts covered elsewhere? [If available, please submit a PDF of the Software User Documentation, or include a hard copy with transmittal of SQAP]

4.b If the intent of the Software User Documentation information is satisfied in other documents, provide the appropriate references (document number, section, and page number).

4.c Training – How is training offered in correctly running the subject software? Complete the appropriate section in the following:

Type	Description	Frequency of training
Training Offered to User Groups as Needed		
Training Sessions Offered at Technical Meetings or Workshops		
Training Offered on Web or Through Video Conferencing		
Other Training Modes		
Training Not Provided		

Guidance for Software User Documentation:

Requirement 9 - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 203
IEEE Standard 1063, *IEEE Standard for Software User Documentation*

5. Software Verification &Validation Documentation (Includes Test Reports)

Verification and Validation (*V&V*) documentation should confirm that a software V&V process has been defined, that V&V has been performed, and that related documentation is maintained to ensure that:

(a) The software adequately and correctly performs all intended functions, and
(b) The software does not perform any unintended function.

The software V&V documentation, either as a standalone document or embedded in other documents and should describe:

- The tasks and criteria for verifying the software in each development phase and validating it at completion,
  - Specification of the hardware and software configurations pertaining to the software V&V
  - Traceability to both software requirements and design
  - Results of the V&V activities, including test plans, test results, and reviews (also see 5.b below)
  - A summary of the status of the software's completeness
  - Assurance that changes to software are subjected to appropriate V&V,
- V&V is complete, and all unintended conditions are dispositioned before software is approved for use, and
- V&V performed by individuals or organizations that are sufficiently independent.

5.a **For the subject software, identify the V&V Documentation that has been prepared.** [If available, please submit a PDF of the Verification and Validation Documentation, or include a hard copy with transmittal of SQAP]

5.b **If the intent of the V&V Documentation information is satisfied in one or more other documents, provide the appropriate references (document number, section, and page number). For example, a "Test Plan and Results" report, containing a plan for software testing, the test results, and associated reviews may be published separately.**

5.c **Testing of software: What has been used to test the subject software?**

☐ Experimental data or observations
☐ Standalone calculations
☐ Another validated software
☐ Software is based on previously accepted solution technique

Provide any reports or written documentation substantiating the responses above.

Guidance for Software Verification & Validation, and Testing Documentation:

Requirement 6 – *Design Phase* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
Requirement 8 – *Testing Phase* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))

Requirement 10 – *Acceptance Test* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 402 (Note: Some aspects of verification may be handled as part of the Design Phase).
ASME NQA-1 2000 Section 404 (Note: Aspects of validation may be handled as part of the Testing Phase).
IEEE Standard 1012, *IEEE Standard for Software Verification and Validation;*
IEEE Standard 1012a, *IEEE Standard for Software Verification and Validation - Supplement to 1012*
IEEE Standard 829, *IEEE Standard for Software Test Documentation.*
IEEE Standard 1008, *Software Unit Testing*

6. Software Configuration Management (SCM)

A process and related documentation for SCM should be defined, maintained, and controlled.

The appropriate documents, such as project procedures related to software change controls, should verify that a software configuration management process exists and is effective.

The following points should be covered in SCM document(s):

- A Software Configuration Management Plan, either in standalone form or embedded in another document,
- Configuration management data such as software source code components, calculational spreadsheets, operational data, run-time libraries, and operating systems,
- A configuration baseline with configuration items that have been placed under configuration control,
- Procedures governing change controls,
- Software change packages and work packages to demonstrate that (1) possible impacts of software modifications are evaluated before changes are made, (2) various software system products are examined for consistency after changes are made, and (3) software is tested according to established standards after changes have been made.

6.a **For the subject software, has a Software Configuration Management Plan been prepared, or are its constituent parts covered elsewhere?** [If available, please submit a PDF of the Software Configuration Management Plan and related procedures, or include hard copies with transmittal of SQAP].

6.b **Identify the process and procedures governing control and distribution of the subject software with users.**

6.c **Do you currently interact with a software distribution organization such as the Radiation Safety Information Computational Center (RSICC)?**

6.d A Central Registry organization, under the management and coordination of the Department of Energy's Office of Environment, Safety and Health (EH), will be responsible for the long-term maintenance and control of the safety analysis toolbox codes for DOE safety analysis applications. Indicate any questions, comments, or concerns on the Central Registry's role and the maintenance of the subject software.

Guidance for Software Configuration Management Plan Documentation:

Requirement 12 – *Configuration Control* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 203
IEEE Standard 828, *IEEE Standard for Software Configuration Management Plans.*

7. Software Problem Reporting and Corrective Action

Software problem reporting and corrective action documentation help ensure that a formal procedure for problem reporting and corrective action development for software errors and failures is established, maintained, and controlled.

A Software Error Notification and Corrective Action Report, procedure, or similar documentation, should be implemented to report, track, and resolve problems or issues identified in both software items, and in software development and maintenance processes. Documentation should note specific organizational responsibilities for implementation. Software problems should be promptly reported to affected organizations, along with corrective actions. Corrective actions taken ensure that:

- Problems are identified, evaluated, documented, and, if required, corrected,
- Problems are assessed for impact on past and present applications of the software by the responsible organization,
- Corrections and changes are executed according to established change control procedures, and
- Preventive actions and corrective actions results are provided to affected organizations.

Identify documentation specific to the subject software that controls the error notification and corrective actions. [If available, please submit a PDF of the Error Notification and Corrective Action Report documentation for the subject software (or related procedures). If this is not available, include hard copies with transmittal of SQAP].

7.aProvide examples of problem/error notification to users and the process followed to address the deficiency. Attach files as necessary.

7.bProvide an assessment of known errors or defects in the subject software and the planned action and time frame for correction.

Category of Error or Defect	Corrective Action	Planned schedule for corre
Major		
Minor		

7.cIdentify the process and procedures governing communication of errors/defects related to the subject software with users.

Guidance for Error/Defect Reporting and Corrective Action Documentation:

Requirement 13 – *Error Impact* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 204
IEEE Standard 1063, *IEEE Standard for Software User Documentation*

8. Resource Estimates

If one or more plans, documents, or sets of procedures identified in parts one (1) through seven (7) do not exist, please provide estimates of the resources (full-time equivalent (40-hour) weeks, FTE-weeks) and the duration (months) needed to meet the specific SQA requirement.

Enter estimate in Table 4 only if specific document has not been prepared, or requires revision.

Table 4. Resource and Schedule for SQA Documentation

Plan/Document/Procedure	Resource Estimate (FTE-weeks)	Duration of Activity (months)
1. Software Quality Assurance Plan		
2. Software Requirements Document		
3. Software Design Document		
4. Test Case Description and Report		
5. Software Configuration and Control		
6. Error Notification and Corrective Action Report		
7. User's Instructions (User's Manual)		
8. Other SQA Documentation		

Comments or Questions:

9. Software Upgrades

Describe modifications planned for the subject software.

Technical Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

User Interface Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Software Engineering Improvements

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Other Planned Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Thank you for your input to the SQA upgrade process. Your experience and insights are critical towards successfully resolving the issues identified in DNFSB Recommendation 2002-1.

REFERENCES

CFR Code of Federal Regulations (CFR). 10 CFR 830, Nuclear Safety Management Rule.

DNFSB Defense Nuclear Facilities Safety Board (2000). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, Technical Report DNFSB/TECH-25, (January 2000).

DNFSB Defense Nuclear Facilities Safety Board (2002). *Recommendation 2002-1, Quality Assurance for Safety-Related Software* , (September 2002).

DOE, U.S. Department of Energy (2000a). *Appendix A, Evaluation Guideline,* DOE-STD-3009-94, *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports* (January 2000).

DOE, U.S. Department of Energy (2002). *Selection of Computer Codes for DOE Safety Analysis Applications* (August 2002).

DOE, U.S. Department of Energy (2003). *Implementation Plan for Defense Nuclear Facilities Safety Board Recommendation 2002-1: Quality Assurance for Safety Software at Department of Energy Nuclear Facilities*, Letter (March 13, 2003); Report (February 28, 2003).

DOE, U.S. Department of Energy (2003a). *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes,* Interim Report, (September 2003).

DOE/EH, U.S. Department of Energy Office of Environment, Safety and Health (2003), *Designation of Initial Safety Analysis Toolbox Codes*, Letter, (March 28, 2003).

SEPARATION

PAGE

Defense Nuclear Facilities Safety Board Recommendation 2002-1
Software Quality Assurance Improvement Plan
Commitment 4.2.1.3:

Software Quality Assurance Improvement Plan: CFAST Gap Analysis

Final Report

U.S. Department of Energy
Office of Environment, Safety and Health
1000 Independence Ave., S.W.
Washington, DC 20585-2040

May 2004

INTENTIONALLY BLANK

FOREWORD

This report documents the outcome of an evaluation of the Software Quality Assurance (SQA) attributes of the CFAST computer code for accident analysis applications, relative to established requirements. This evaluation, a "gap analysis," is performed to meet commitment 4.2.1.3 of the Department of Energy's Implementation Plan to resolve SQA issues identified in the Defense Nuclear Facilities Safety Board Recommendation 2002-1.

Suggestions for corrections or improvements to this document should be addressed to –

Chip Lagdon
EH-31/GTN
U.S. Department of Energy
Washington, D.C. 20585-2040
Phone (301) 903-4218
Email: chip.lagdon@eh.doe.gov

INTENTIONALLY BLANK

REVISION STATUS

Page/Section	Revision	Change
1. Entire Document	1. Interim Report	1. Original Issue
2. Entire Document	2. Final Report, May 3, 2004	2. Updated all sections per review comments. Changed reference from CFAST 5.0.1 to CFAST 5.1.

INTENTIONALLY BLANK

CONTENTS

Section Pag

FOREWORD iii

REVISION STATUS v

CONTENTS vii

TABLES ix

EXECUTIVE SUMMARY xi

1.0 Introduction 1-1

1.1 Background: Overview of Designated Toolbox Software in the Context of 10 CFR 830 1-1
1.2 Evaluation of Toolbox Codes 1-2
1.3 Uses of the Gap Analysis 1-2
1.4 Scope 1-3
1.5 Purpose 1-3
1.6 Methodology for Gap Analysis 1-3
1.7 Summary Description of Software Being Reviewed 1-5

2.0 Assessment Summary Results 2-1

2.1 Criteria Met 2-1
2.2 Exceptions to Requirements 2-1
2.3 Other Areas Needing Improvement 2-2
2.4 CFAST Issues Cited in TECH-25 and Recommended Approaches for Resolutions 2-3
2.5 Conclusion Regarding Software's Ability to Meet Intended Function 2-4

3.0 Lessons Learned 3-1

4.0 Detailed Results of the Assessment Process 4-1

4.1 Topical Area 1 Assessment: Software Classification 4-1
4.1.1 Criterion Specification and Result 4-1
4.1.2 Sources and Method of Review 4-2
4.1.3 Software Quality-Related Issues or Concerns 4-2
4.1.4 Recommendations 4-2
4.2 Topical Area 2 Assessment: SQA Procedures and Plans 4-2
4.2.1 Criterion Specification and Result 4-3
4.2.2 Sources and Method of Review 4-3
4.2.3 Software Quality-Related Issues or Concerns 4-3
4.2.4 Recommendations 4-3
4.3 Topical Area 3 Assessment: Requirements Phase 4-3
4.3.1 Criterion Specification and Result 4-4
4.3.2 Sources and Method of Review 4-4
4.3.3 Software Quality-Related Issues or Concerns 4-4
4.3.4 Recommendations 4-4
4.4 Topical Area 4 Assessment: Design Phase 4-4
4.4.1 Criterion Specification and Result 4-5

4.4.2 Sources and Method of Review 4-7
4.4.3 Software Quality-Related Issues or Concerns 4-7
4.4.4 Recommendations 4-7
4.5 Topical Area 5 Assessment: Implementation Phase 4-7
4.5.1 Criterion Specification and Result 4-8
4.5.2 Sources and Method of Review 4-8
4.5.3 Software Quality-Related Issues or Concerns 4-8
4.5.4 Recommendations 4-8
4.6 Topical Area 6 Assessment: Testing Phase 4-8
4.6.1 Criterion Specification and Result 4-9
4.6.2 Sources and Method of Review 4-10
4.6.3 Software Quality-Related Issues or Concerns 4-10
4.6.4 Recommendations 4-10
4.7 Topical Area 7 Assessment: User Instructions 4-10
4.7.1 Criterion Specification and Result 4-10
4.7.2 Sources and Method of Review 4-11
4.7.3 Software Quality-Related Issues or Concerns 4-12
4.7.4 Recommendations 4-12
4.8 Topical Area 8 Assessment: Acceptance Test 4-12
4.8.1 Criterion Specification and Result 4-12
4.8.2 Sources and Method of Review 4-13
4.8.3 Software Quality-Related Issues or Concerns 4-13
4.8.4 Recommendations 4-13
4.9 Topical Area 9 Assessment: Configuration Control 4-14
4.9.1 Criterion Specification and Result 4-14
4.9.2 Sources and Method of Review 4-14
4.9.3 Software Quality-Related Issues or Concerns 4-14
4.9.4 Recommendations 4-14
4.10 Topical Area 10 Assessment: Error Impact 4-15
4.10.1 Criterion Specification and Result 4-15
4.10.2 Sources and Method of Review 4-16
4.10.3 Software Quality-Related Issues or Concerns 4-16
4.10.4 Recommendations 4-16
4.11 Training Program Assessment 4-16
4.12 Software Improvements 4-16

5.0 Conclusions **5-1**

6.0 Acronyms and Definitions **6-1**

7.0 References **7-1**

APPENDIX A. — SOFTWARE INFORMATION TEMPLATE A-1

APPENDIX B. — SFPE TRAINING CLASS DESCRIPTIONS B-1

APPENDIX C. — CFAST REVISION NOTICE C-1

TABLES

Page

Table 1-1. – Plan for SQA Evaluation of Existing Safety Analysis Software 1-3

Table 1-2 – Summary Description of CFAST Software 1-6

Table 1-3 — Software Documentation Reviewed for CFAST 1-7

Table 2-1 — Summary of Important Exceptions, Reasoning, and Suggested Remediation 2-2

Table 2-2 — Summary of Recommendations for CFAST 2-3

Table 3-1 — Lessons Learned 3-1

Table 4.0-1 — Cross-Reference of Requirements with Subsection and Entry from (DOE, 2003d) 4-1

Table 4.1-1 — Subset of Criteria for Software Classification Topic and Results 4-2

Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results 4-3

Table 4.3-1 — Subset of Criteria for Dedication Topic and Results 4-4

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results 4-5

Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results 4-8

Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results 4-9

Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results 4-11

Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results 4-13

Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results 4-14

Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results 4-15

INTENTIONALLY BLANK

Software Quality Assurance Improvement Plan: CFAST Gap Analysis

EXECUTIVE SUMMARY

The Defense Nuclear Facilities Safety Board (DNFSB) issued Recommendation 2002-1 on *Quality Assurance for Safety-Related Software* in September 2002 (DNFSB 2002). The Recommendation identified a number of quality assurance issues for software used in the 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 high-use, Software Quality Assurance (SQA)-compliant safety analysis codes is one of the major improvement actions discussed in the *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities*. A DOE safety analysis toolbox would contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed and maintained for DOE-broad safety basis applications.

The fire modeling software *Consolidated Model of Fire Growth and Smoke Transport* (CFAST), both versions 3.1.7 and 5.1, is one of the codes designated for the toolbox. To determine the actions needed to bring the CFAST software into compliance with the SQA qualification criteria, and develop an estimate of the resources required to perform the upgrade, the Implementation Plan has committed to sponsoring a code-specific gap analysis document. The gap analysis evaluates the software quality assurance attributes of CFAST against identified criteria.

The balance of this document provides the outcome of the CFAST gap analysis compliant with NQA-1-based requirements as contained in U.S. Department of Energy, *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes*, (DOE, 2003d). It was determined that CFAST does meet its intended function for use in supporting documented safety analysis. However, as with all safety-related software, users should be aware of current limitations and capabilities of CFAST for supporting safety analysis. Informed use of the software can be assisted by the current set of CFAST reports (See Table 1-1.), and the code guidance report for DOE safety analysts, *The CFAST Computer Code Application Guidance for Documented Safety Analysis* (DOE, **Error! Reference source not found.**). Furthermore, while SQA improvement actions are recommended for both versions of CFAST, no evidence has been found of software-induced errors that have led to non-conservatisms in nuclear facility operations or in the identification of facility controls no evidence has been found of programming, logic, or other types of software errors in CFAST that have led to non-conservatisms in nuclear facility operations, or in the identification of facility controls.

Of the ten primary SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), two requirements are met at acceptable level, i.e., *Classification* (1) and *Configuration Control* (9). Five requirements are partially met: *Implementation Phase* (5), *Testing Phase* (6), *User Instructions* (7), *Acceptance Test* (8), and *Error Notification and Corrective Action* (10). Three requirements are not met *SQA Procedures and Plans(2), Requirements Phase(3), and Design Phase(4)*. Improvement actions are recommended for CFAST to fully meet eight of the requirements. This evaluation outcome is deemed acceptable because: (1) CFAST 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 CFAST is limited to those analytic applications for which the software is intended.

By order of priority, it is recommended that CFAST software improvement actions be taken, especially:

1. Revising software documentation and user instructions to provide a comprehensive description of the software output (Section 4.7).
2. Establishing an acceptance test protocol to be used to assure that the installed version of CFAST is working properly when software is installed on a new computer system (Section 4.8)
3. Defining the minimum training necessary to use the software and offering the training on a regular basis (Section 4.7)
4. Implementing a formal error notification and corrective action process (Section 4.10).

Performing these four primary actions should satisfactorily improve the SQA compliance status of CFAST relative to the primary evaluation criteria cited in this report.

It is recommended that the most significant SQA shortcomings be addressed initially, including error reporting, user training and user instructions. It is estimated that approximately 0.5 full-time equivalent year (FTE) would be required to address these three SQA areas. An additional several FTE-months is estimated for completing improvement actions recommended in the five partially compliant areas.

It is recommended that CFAST user training for DOE safety analysis applications be conducted formally on, at minimum, an annual basis. Prerequisites for, and core knowledge needed by, the user prior to initiating CFAST applications should be documented by the code developer.

Approximately one FTE-month per year would be needed to maintain a web-based error notification and corrective action process for CFAST (Section 4.10). However, such a process has not been defined in depth for CFAST and the other designated toolbox codes.

INTENTIONALLY BLANK

1.0 Introduction

This document reports the results of a gap analysis for versions 3.1.7 and 5.1 of the CFAST computer code. The intent of the gap analysis is to determine the actions needed to bring the specific software into compliance with established Software Quality Assurance (SQA) criteria. A secondary aspect of this report is to develop an estimate of the level of effort required to upgrade each code based on the gap analysis results.

1.1 Background: Overview of Designated Toolbox Software in the Context of 10 CFR 830

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* (DNFSB, 2000). 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.

While progress was made in resolving several of the issues raised in TECH-25, the DNFSB issued Recommendation 2002-1 on *Quality Assurance for Safety-Related Software* in September 2002. 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, the quality of the software used to design or develop safety-related controls, and the proficiency of personnel using the software. The Recommendation identified a number of quality assurance issues for software used in the 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 high-use, SQA-compliant safety analysis codes is one of the major commitments contained in the March 2003 *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities* (IP). In time, the DOE safety analysis toolbox will contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed and maintained for DOE-broad safety basis applications.

Six computer codes, including ALOHA (chemical release dispersion/consequence analysis), CFAST (fire analysis), EPIcode (chemical release dispersion/consequence analysis), GENII (radiological dispersion/consequence analysis), MACCS2 (radiological dispersion/consequence analysis), and MELCOR (leak path factor analysis), were designated by DOE for the toolbox (DOE, 2003b). It is found that this software provides generally recognized and acceptable approaches for modeling source term and consequence phenomenology, and can be applied as appropriate to support accident analysis in Documented Safety Analyses (DSAs).

As one of the designated toolbox codes, CFAST versions 3.1.7 and 5.1, will require some degree of quality assurance improvement before meeting current DOE SQA standards. The analysis documented herein is an evaluation of CFAST relative to current DOE software quality assurance criteria. It assesses the extent of the deficiencies, or gaps, to provide DOE and the software developer the extent to which minimum upgrades are needed. The overall assessment is therefore termed a "gap" analysis.

1.2 Evaluation of Toolbox Codes

The quality assurance criteria identified in later sections of this report are defined as the set of established requirements, or bases, by which to evaluate each designated toolbox code. This gap analysis evaluation, is commitment 4.2.1.3 in the IP:

> Perform a SQA evaluation to the toolbox codes to determine the actions needed to bring the codes into compliance with the SQA qualification criteria, and develop a schedule with milestones to upgrade each code based on the SQA evaluation results.

This process is a prerequisite step for software improvement. It allowed DOE to determine the current limitations and vulnerabilities of each code as well as help define and prioritize the steps required for improvement.

Early in the SQA evaluation program, it was anticipated that each toolbox code owner would provide input information on the SQA programs, processes, and procedures used to develop their software. However, most of the designated toolbox software, including CFAST, was developed without complete conformance to software quality standards. Furthermore, many of the software developer organizations cannot confirm that key processes were followed. Therefore, most of the SQA evaluation has been preceded with reconstructing software development processes based on anecdotal evidence and limited, supporting documentation.

For independence reasons, the gap analysis is performed by a SQA evaluator, not affiliated with the CFAST development program. While independent of the code developer, the SQA evaluators responsible for CFAST are knowledgeable in the use of the software for accident analysis applications, and understand current software development standards.

1.3 Uses of the Gap Analysis

The gap analysis provides key information to DOE, code developers, and code users.

DOE obtains the following benefits:

- Estimates of the resources required to perform modifications to designated toolbox codes
- Basis for schedule and prioritization to upgrade each designated toolbox code.

Each code developer is provided:

- Information on areas where software quality assurance improvements are needed to comply with industry SQA standards and practices
- Specific areas for improvement to guide development of new versions of the software.

DOE safety analysts and code users benefit from:

- Improved awareness of the strengths, limits, and vulnerable areas of each computer code
- Recommendations for code use in safety analysis application areas.

1.4 Scope

The gap analysis is applicable to the CFAST code, one of the six designated toolbox codes for safety analysis. While CFAST is the subject of the current report, other safety analysis software considered for the toolbox in the future may be evaluated with the same process applied here. The template outlined in this document is applicable for any analytical software as long as the primary criteria are ASME NQA-1, 10 CFR 830, and related DOE directives discussed in DOE (2003d).

1.5 Purpose

The purpose of this report is to document the gap analysis performed on the CFAST code as part of DOE's implementation plan on SQA improvements.

1.6 Methodology for Gap Analysis

The gap analysis for CFAST was based on the plan and criteria described in Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes (**Error! Reference source not found.**). The overall methodology used for the gap analysis is summarized in Table 1-1. The gap analysis utilized ten of the fourteen topical areas listed in **Error! Reference source not found.**, related to software quality assurance to assess the quality of the CFAST software. The ten areas are those particularly applicable to the software development, specifically: (1) Software Classification, (2) SQA Procedures/Plans, (5) Requirements Phase, (6) Design Phase, (7) Implementation Phase, (8) Testing Phase, (9) User Instructions, (10) Acceptance Test, (12) Configuration Control, and (13) Error Impact. Each area, or requirement, is assessed individually in Section 4.

Requirements 3 (Dedication), 4 (Evaluation), 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 evaluating the areas listed above. Requirement 11 (Operation and Maintenance) is only partially applicable to software development, and is interpreted to be applicable mostly to the software user organization.

Table 1-1. – Plan for SQA Evaluation of Existing Safety Analysis Software[1]

Phase	Procedure
1. Prerequisites	a. Determine whether sufficient information is provided by the software developer 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.

[1] From Table 2-2 in DOE (DOE 2003c).

Phase	Procedure
	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 Procedure, 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 Procedure, 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.
6. Training	a. Identify current training programs provided by developer. b. Determine applicability of training for DOE facility safety analysis.

Phase	Procedure
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.

An information template was transmitted to the Safety Analysis Software Developers on 20 October 2003 to provide basic information as input to the gap analysis process. The core section of the template is attached as Appendix A to the present report. NIST has provided a positive response to this request. Information gleaned from this request is included in the preparation of this report, Section 4.0.

1.7 Summary Description of Software Being Reviewed

The gap analysis was performed on both versions 3.1.7 and 5.1 of the CFAST code. CFAST was initially developed in 1990 and (http://cfast.nist.gov/versionhistory.html) was written in FORTRAN. This software is maintained by the National Institute of Standards and Technology and is in widespread use in the fire protection industry to evaluate the safety of exiting buildings, perform post-fire reconstructions and to evaluate performance based designs. Since the issuance of DOE-STD-3009-94 for nuclear facility accident analysis, CFAST has been used for DOE applications primarily as a tool for establishing compartment temperature profiles and target temperature predictions. The output of CFAST is used to support decision-making on control selection in nuclear facilities, specifically identification of safety structures. systems, and components (SSCs).

CFAST is a fire "model used to calculate the evolving distribution of smoke, fire gases and temperature throughout a constructed facility during a fire. In CFAST, each compartment is divided into two layers. [Models based on this simplification are referred to as zone models in the fire protection industry.] The modeling equations used in CFAST take the mathematical form of an initial value problem for a system of ordinary differential equations (ODE). These equations are derived using the conservation of mass, the conservation of energy (equivalently the first law of thermodynamics), the ideal gas law and relations for density and internal energy. These equations predict as functions of time quantities such as pressure, layer heights and temperatures given the accumulation of mass and enthalpy in the two layers. The CFAST model then consists of a set of ODEs to compute the environment in each compartment and a collection of algorithms to compute the mass and enthalpy source terms required by the ODEs." (DOE, 2004, U.S. Department of Energy (2004). *The CFAST Computer Code Application Guidance for Documented Safety Analysis,* (May 2004).

Jones, 2003)

A brief summary of CFAST is contained in Table 1-2.

The set of documents reviewed as part of this gap analysis are listed in Table 1-3. All of this material is available at the NIST website www.cfast.nist.gov.

Table 1-2 – Summary Description of CFAST Software

Type	Specific Information
Code Name	*Consolidated Model of Fire Growth and Smoke Transport* (CFAST),
Versions of the Code	Versions 3.1.7 and 5.1
Developing Organization and Sponsor	National Institute of Standards and Technology 100 Bureau Drive, MS 8883, Gaithersburg, MD 20899
Auxiliary Codes	FAST: Graphical User Interface that supports CFAST 3.1.7 CPLOT: Post-processor for use with CFAST history files
Software Platform/ Portability	PC (Windows 95 and later), IRIX (6.3)
Coding and Computer	FORTRAN, C
Technical Support	Walter W. Jones National Institute of Standards and Technology 301.975.6887 wwj@nist.gov
Code Procurement Point of Contact	Freeware available from: http://cfast.nist.gov/
Documentation Supplied with Code Transmittal	See Table 1–3.
Nature of Problem Addressed by Software	Fire growth and smoke spread
Significant Strengths of Software	Very fast; it has been verified and validated.
Known Restrictions or Limitations	Cannot calculate deflagration or detonation scenarios.
Preprocessing (set-up) time for Typical Safety Analysis Calculation	Problem dependent. Simple calculations take only a few minutes to set up and run
Execution Time	Run time will vary with the computer platform and the complexity of the model. Six compartment cases run faster than real time with a 2.6 GHz processor.
Computer Hardware Requirements	Disk space for version 5.1 is about 5 MB and requires about 10 MB of memory for large cases. History files (*.HI) can be up to 10 MB for complex cases.
Computer Software Requirements	The GUI uses Microsoft Office .ocx dialog boxes.
Contributing Organization(s)	Naval Research Laboratory, Nuclear Regulatory Commission, Concrete Masonry Institute

Table 1-3 — Software Documentation Reviewed for CFAST

No.	Reference purpose	Reference
1.	Users Guide for versions 3.1.7 and 5.1	Peacock, R. D., Paul A. Reneke, Walter W. Jones, Richard W. Bukowski, and Glenn P. Forney. 2000. *A User's Guide for FAST: Engineering Tools for Estimating Fire Growth and Smoke Transport.* Gaithersburg: MD. National Institute of Standards and Technology. (January) NIST Special Publication 921, 2000 edition (Peacock, 2000).
2.	Technical reference for version 3.1.7	Peacock, R. D., Paul A. Reneke, Walter W. Jones, Rebecca M. Portier, and Glenn P. Forney. 1993. *CFAST, the Consolidated Model of Fire Growth and Smoke Transport.* Gaithersburg: MD. National Institute of Standards and Technology. (February) NIST Technical Note 1299 (Peacock, 1993).
3.	Technical reference for version 5.1	Jones, Walter W., Glenn P. Forney, Richard D. Peacock and Paul A. Reneke. 2003. *A Technical Reference for CFAST: An Engineering Tool for Estimating Fire and Smoke Transport.* Gaithersburg: MD. National Institute of Standards and Technology. (April) NIST TN 1431 (DOE, 2004, U.S. Department of Energy (2004). *The CFAST Computer Code Application Guidance for Documented Safety Analysis,* (May 2004). Jones, 2003).

2.0 Assessment Summary Results

2.1 Criteria Met

Of the ten general topical quality areas assessed in the gap analysis, two satisfactorily met the criteria. The analysis found that the CFAST SQA program, in general, met criteria for *Software Classification* and *Configuration Control*, Requirements 1 and 9, respectively. Eight topical quality areas were not met satisfactorily. The major areas for improvement are covered below in Section 2.2 (Exceptions to Requirements). The majority of these areas for improvement actions are expected because CFAST was developed before the DOE SQA requirements. Detail on the evaluation process relative to the requirements, and the criteria applied, are found in Section 4.

2.2 Exceptions to Requirements

Some of the more important exceptions to criteria found for CFAST are listed in Table 2-1. The requirement is given, the reason the requirement was not met is provided, and remedial action(s) are listed to correct the exceptions. The ten criteria evaluated are those predominantly executed by the software developer. However, it is noted that criteria for SQA Procedures/Plan, Testing, Acceptance Test, Configuration Control, and Error Notification also have requirements for the organization implementing the software. These criteria were assessed in the present evaluation only from the code developer perspective. The most significant exceptions are:

- The CFAST Users Manual does not provide a comprehensive description of the software output (Section 4.7).
- A description of the training necessary to use the software is not available (Section 4.7)
- An acceptance test protocol to be used to assure that the installed version of CFAST is working properly is not documented (Section 4.8)
- There is no formal error notification and corrective action process (Section 4.10).

Table 2-1 — Summary of Important Exceptions, Reasoning, and Suggested Remediation

No.	Criterion	Reason Not Met	Remedial Action(s)
1.	SQA Procedures/ Plans (Section 4.2)	SQA Plan and Procedures for CFAST were not prepared.	Develop a backfit plan and procedures.
2.	Requirements Phase (Section 4.3)	Requirements phase documentation for CFAST is not complete.	Develop backfit documentation.
3.	Design Phase (Section 4.4)	Design phase documentation for CFAST was not complete.	Develop backfit documentation.
4.	Implementation Phase (Section 4.5)	Implementation phase documentation for CFAST was not complete.	Develop backfit documentation.
5.	Testing Phase (Section 4.6)	NIST has recently prepared a verification and validation report for CFAST. The report was not readily available to be included in this final report.	Contact NIST to obtain the presently available documentation, review this documentation and develop an action plan.
6.	User Instructions (Section 4.7)	The user's manual does not list approved operating systems, a description of training necessary to use the software, a comprehensive description of the software outputs, a description of software and hardware limitations and a description on user messages.	Develop a training description with input from NIST. Work with NIST to establish a comprehensive description of CFAST outputs.
7.	Acceptance Test (Section 4.8)	An Acceptance Test protocol is not available. There is no known formal procedure to assure that an installed version of CFAST is working properly.	Work with NIST to document the existing Acceptance Test protocol.
8.	Error Impact (Section 4.10)	There is no formal Error Notification and Corrective Action Report process for CFAST. A version history is maintained on the CFAST web site that describes software updates.	DOE should establish a formal Error Notification and Correction Action Report process for CFAST.

These exceptions are the most significant since they can directly affect the successful use of CFAST. All of the CFAST gap analysis recommendations are summarized in Table 2-1.

2.3 Other Areas Needing Improvement

The Graphical User Interface to support version 5.1 needs to be released. The presently available version is considered an alpha release and has limited capabilities.

CFAST does not explicitly calculate leak path factors (LPFs). It appears that it should be capable of this function, however instructions to accomplish this are not provided. Since fire is often a dominant risk in nuclear facilities, a software that could estimate LPFs would be very beneficial.

2.4 CFAST Issues Cited in TECH-25 and Recommended Approaches for Resolutions

One technical issue was noted in TECH-25 that explicitly related CFAST software. This section discusses the issue and recommended disposition.

TECH-25 noted, "no formal SQA plan was documented for this code **[Error! Reference source not found.]**. Some validation documentation is referenced. The SQA/V&V status of this code is not commensurate with current industry standards." Completion of this gap analysis and the development of an action plan will address this comment.

Table 2-2 — Summary of Recommendations for CFAST

No.	Type*	Recommendation
2.1	OI	Work with NIST to establish a backfit SQA plan and procedures for CFAST.
3.1	OI	Work with NIST to establish backfit Requirements Phase documentation for CFAST.
4.1	OI	Work with NIST to establish backfit Design Phase documentation for CFAST.
5.1	OI	Work with NIST to establish backfit Implementation Phase documentation for CFAST.
6.1	OI	Contact NIST to obtain a copy of the verification and validation report.
6.2	OI	Review recently prepared verification and validation report when it becomes available and establish a plan to identify gaps as appropriate.
7.1	UI	The user's manual should be updated to reflect the minimum operating system requirements.
7.2	PI	DOE should establish the minimum qualification for personnel who are expected to prepare safety analyses using CFAST. (Two levels of qualification may be appropriate. The lower tier would be to operate the software and produce results, the higher tier would be to interpret the results.)
7.3	UI	A description of output files should be prepared and included in the user's manual.
7.4	UI	Sample problems that include the input data files, output data files and a discussion of the results should be provided.
7.5	UI	The user's manual should be updated to include a description of software and hardware limitations.
8.1	OI	Work with NIST to document the existing acceptance tests and their use.
9.1	OI	Contact NIST to obtain a copy of the NIST internal report documenting the version update process.
9.2	OI	Review the existing NIST report documenting the version update process when it becomes available and establish a plan to identify gaps as appropriate.
10.1	OI	Establish an Error Impact Management Process plan.
12.1	UI	Support the development of a GUI for CFAST 5.1 by contributing to CFAST users groups.
12.2	TM	Fund NIST to modify CFAST to establish LPF values utilizing the contaminate term (CT keyword).

*OI – Open Item in gap analysis, PI – DOE Procedure Improvement, UI – User Interface Enhancements, TM – Technical Model Upgrade

2.5 Conclusion Regarding Software's Ability to Meet Intended Function

The CFAST code was evaluated to determine if the software, in its current state, meets the intended function in a safety analysis context as assessed in this gap analysis. When the code is run for the intended applications as detailed in the code guidance document, *The CFAST Computer Code Application Guidance for Documented Safety Analysis*, (**Error! Reference source not found.**), it is judged that it will meet the intended function. Current software concerns and issues can be avoided by understanding CFAST limitations and capabilities, and applying the software in the appropriate types of scenarios for which precedents have been identified.

The software can be applied for modeling those types of scenarios where precedents exist, and there is confidence that alternative analysis or experimental data would adequately confirm the code predictions.

Confidence in CFAST to meet its intended function is expected to increase as new benchmarking problems are completed (NRC, 2002).

3.0 Lessons Learned

Table 3-1 provides a summary of the lessons learned during the performance of the CFAST gap analysis.

Table 3-1 — Lessons Learned

No.	Lesson
1.	Use of NQA-1 or other SQA criteria could not be fully verified. It is known that significant effort has been expended in demonstrating the ability of CFAST to successfully predict fire behavior, however the documentation supporting this is not readily available.
2.	Non-DOE sponsored software that is used to support safety analysis is unlikely to explicitly meet the requirements of ASME NQA-1. To demonstrate compliance with Quality Assurance criteria in Subpart A to 10 CFR 830 (Nuclear Safety Management) will require resources beyond that applied for public-domain codes such as CFAST. A backfit approach to address the quality assurance requirements associated with the use of such software should be considered.
3.	Additional opportunities and venues should be sought for training and user qualification on safety analysis software. This is a long-term deficiency that needs to be addressed for CFAST and other designated software for the DOE toolbox.

4.0 Detailed Results of the Assessment Process

Ten topical areas, or requirements are presented in the assessment as listed in Table 4.0-1. Training and Software Improvements (resource estimate) sections follow the ten topical areas.

In the tables that follow, criteria and recommendations are labeled as (1.x, 2.x, ...10.x) with the first value (1., 2., ...) corresponding to the topical area and the second value (x), the sequential table order.

Table 4.0-1 — Cross-Reference of Requirements with Subsection and Entry from (DOE 2003e Error! Reference source not found.**)**

Subsection (This Report)	Corresponding Entry Table 3-2 from **Error! Reference source not found.**	Requirement
4.1	1	Software Classification
4.2	2	SQA Procedures/Plans
4.3	5	Requirements Phase
4.4	6	Design Phase
4.5	7	Implementation Phase
4.6	8	Testing Phase
4.7	9	User Instructions
4.8	10	Acceptance Test
4.9	12	Configuration Control
4.10	13	Error Impact [Notification]

4.1 Topical Area 1 Assessment: Software Classification

This area corresponds to the requirement entitled Software Classification in Table 3-3 of (**Error! Reference source not found.**).

4.1.1 Criterion Specification and Result

Error! Reference source not found. Sufficient documentation is provided at the NIST sponsored CFAST website, http://fast.nist.gov/, to make an informed determination of the classification of the software. A user of the CFAST software for safety analysis applications would be expected to interpret the information on the software in light of the requirements for consequence analysis discussed in Appendix A to DOE-STD-3009-94 to decide on an appropriate safety classification. For most organizations, the safety class or safety significant classification, or Level B in the classification hierarchy discussed in (**Error! Reference source not found.**), would be selected, which by definition relates to 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
- 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 4.1-1 — Subset of Criteria for Software Classification Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
1.1	The code developer must provide sufficient information to allow the user to make an informed decision on the classification of the software.	Yes	It is concluded that sufficient information is provided at the NIST sponsored CFAST/FAST website, http://fast.nist.gov/, for the user to make an informed determination of the classification of the software.

4.1.2 Sources and Method of Review

Documentation provided at the NIST sponsored CFAST website, http://fast.nist.gov/, was used as the basis for establishing the responses for this requirement.

4.1.3 Software Quality-Related Issues or Concerns

There are no SQA issues or concerns relative to this requirement.

4.1.4 Recommendations

This requirement is met. No recommendations are required at this time to improve compliance with the requirement.

4.2 Topical Area 2 Assessment: SQA Procedures and Plans

This area corresponds to the requirement entitled SQA Procedures / Plans in Table 3-3 of (**Error! Reference source not found.**).

4.2.1 Criterion Specification and Result

Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
2.1	Procedures/plans for SQA have identified organizations responsible for performing work; independent reviews, etc.	No	A verifiable, written set of SQA plans and procedures is lacking for CFAST. When CFAST was developed, such plans were not required.
2.2	Procedures/plans for SQA have identified software engineering methods.	No	See Criterion 2.1 summary remarks.
2.3	Procedures/plans for SQA have identified documentation to be required as part of program.	No	See Criterion 2.1 summary remarks.
2.4	Procedures/plans for SQA have identified standards, conventions, techniques, and/or methodologies, which shall be used to guide the software development, methods to ensure compliance with the same.	No	See Criterion 2.1 summary remarks.
2.5	Procedures/plans for SQA have identified software reviews and schedule.	No	See Criterion 2.1 summary remarks.
2.6	Procedures/plans for SQA have identified methods for error reporting and corrective actions.	No	See Criterion 2.1 summary remarks.

4.2.2 Sources and Method of Review

Documentation provided at the NIST sponsored CFAST website, http://fast.nist.gov/, supplemented with informal communications, was used as the basis for establishing the responses for this requirement.

4.2.3 Software Quality-Related Issues or Concerns

The unavailability of a verifiable, written set of SQA plan and procedures for CFAST should be addressed.

4.2.4 Recommendations

The criteria are not met. Thus, the requirement is not met. Recommendations related to this topical area are:

Recommendation 2.1 — Work with NIST to establish a backfit SQA plan and procedures for CFAST.

4.3 Topical Area 3 Assessment: Requirements Phase

This area corresponds to the requirement entitled Requirements Phase in Table 3-3 of (**Error! Reference source not found.**).

4.3.1 *Criterion Specification and Result*

Table 4.3-1 — Subset of Criteria for Dedication Topic and Results

Criterion Number	Criterion Specification	Compliance	Summary Remarks
3.1	Software requirements for the subject software have been established.	Partial	See Summary Remark to 3.2.
3.2	Software requirements are specified, documented, reviewed and approved.	Partial	Improvements to CFAST are commonly developed using task orders. Most of this documentation is not generally available.
3.3	Requirements define the functions to be performed by the software and provide detail and information necessary to design the software.	Partial	See Summary Remark to 3.2.
3.4	A Software Requirements Document, or equivalent defines requirements for functionality, performance, design inputs, design constraints, installation considerations, operating systems (if applicable), and external interfaces necessary to design the software.	No	
3.5	Acceptance criteria are established in the software requirements documentation for each of the identified requirements.	No	

4.3.2 *Sources and Method of Review*

Documentation provided at the NIST sponsored CFAST website, http://fast.nist.gov/, supplemented with informal communications, was used as the basis for establishing the responses for this requirement.

4.3.3 *Software Quality-Related Issues or Concerns*

The unavailability of a written description of the Requirements Phase for CFAST should be addressed.

4.3.4 *Recommendations*

The criteria are not or partially met. Thus, the requirement is not met. Recommendations related to this topical area are:

Recommendation 3.1 — Work with NIST to establish backfit Requirements Phase documentation for CFAST.

4.4 Topical Area 4 Assessment: Design Phase

This area corresponds to the requirement entitled Design Phase in Table 3.3 of (**Error! Reference source not found.**).

4.4.1 Criterion Specification and Result

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.1	The software design was developed, documented, reviewed and controlled.	Uncertain	
4.2	Code developer(s) 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.	Uncertain	
4.3	The following design should be present and documented: specification of interfaces, overall structure (control and data flow) and the reduction of the overall structure into physical solutions (algorithms, equations, control logic, and data structures).	Uncertain	
4.4	The following design should be present and documented: 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.	Uncertain	
4.5	The following design should be present and documented: evidence of measures to mitigate the consequences of software design problems. These potential problems include external and internal abnormal conditions and events that can affect the computer program.	Uncertain	
4.6	A Software Design Document, or equivalent, is available and contains a description of the major components of the software design as they relate to the software requirements.	No	
4.7	A Software Design Document, or equivalent, is available and contains 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.	No	

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.8	A Software Design Document, or equivalent, is available and contains a description of the allowable or prescribed ranges for inputs and outputs.	Partial	The limitations for many parameters are not fully described. Use of the software requires a working knowledge in fire modeling and severity analysis to judge if the inputs and output information is logical.
4.9	A Software Design Document, or equivalent, is available and contains the design described in a manner that can be translated into code.	No	
4.10	A Software Design Document, or equivalent, is available and contains 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.	No	
4.11	The organization responsible for the design identified and documented the particular verification methods to be used and assured that an Independent Review was performed and documented. This review evaluated the technical adequacy of the design approach; assured internal completeness, consistency, clarity, and correctness of the software design; and verified that the software design is traceable to the requirements.	No	While some elements of this criterion may have been met informally per discussions with the software developer, there is no written documentation that allows confirmation.
4.12	The organization responsible for the design assured that the test results adequately demonstrated that the requirements were met.	Uncertain	
4.13	The Independent Review was performed by competent individual(s) other than those who developed and documented the original design, but who may have been from the same organization.	Uncertain	
4.14	The results of the Independent Review are documented with the identification of the verifier indicated.	Uncertain	
4.15	If review alone was not adequate to determine if requirements are met, alternate calculations were used, or tests were developed and integrated into the appropriate activities of the software development cycle.	Uncertain	
4.16	Software design documentation was completed prior to finalizing the Independent Review.	Uncertain	

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.17	The extent of the Independent Review and the methods chosen are shown to be a function of: ➢ The importance to safety, ➢ The complexity of the software, ➢ The degree of standardization, and ➢ The similarity with previously proven software.	Uncertain	

4.4.2 Sources and Method of Review

Documentation provided at the NIST sponsored CFAST website, http://fast.nist.gov/, supplemented with informal communications, was used as the basis for establishing the responses for this requirement.

4.4.3 Software Quality-Related Issues or Concerns

The unavailability of a written description of the Requirements Phase for CFAST should be addressed.

4.4.4 Recommendations

Recommendations related to this topical area are:

Recommendation 4.1 — Work with NIST to establish a backfit Design Phase documentation for CFAST.

4.5 Topical Area 5 Assessment: Implementation Phase

This area corresponds to the requirement entitled Implementation Phase in Table 3-3 of (**Error! Reference source not found.**).

4.5.1 *Criterion Specification and Result*

Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
5.1	The implementation process resulted in software products such as computer program listings and instructions for computer program use.	Uncertain	Because SQA plans and procedures from the software developer are not available, a thorough evaluation was not possible.
5.2	Implemented software was analyzed to identify and correct errors.	Yes	Output between different versions of the code were compared using the software COMPARE (Alvord, 1995)
5.3	The source code finalized during verification (this phase) was placed under configuration control.	Yes	A copy of the current source code is controlled by the NIST CFAST Subject Matter Expert.
5.4	Documentation during verification included a copy of the software, test case description and associated criteria that are traceable to the software requirements and design documentation.	No	

4.5.2 *Sources and Method of Review*

Documentation provided at the NIST sponsored CFAST website, http://fast.nist.gov/, supplemented with informal communications, was used as the basis for establishing the responses for this requirement.

4.5.3 *Software Quality-Related Issues or Concerns*

The unavailability of a written description of the Implementation Phase for CFAST should be addressed.

4.5.4 *Recommendations*

The criteria are partially met. Thus, the requirement is not met. Recommendations related to this topical area are:

Recommendation 5.1 — Work with NIST to establish backfit Implementation Phase documentation for CFAST.

4.6 Topical Area 6 Assessment: Testing Phase

This area corresponds to the requirement entitled Testing Phase in Table 3-3 of (**Error! Reference source not found.**).

NIST is about to publish *Verification and Validation of CFAST, a Model for Fire Growth and Smoke Transport*, (NIST IR 7080 - 2004). This report was not available to be reviewed as part of this gap analysis.

4.6.1 Criterion Specification and Result

Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
6.1	The software was validated by executing test cases.	Yes	
6.2	Testing demonstrated the capability of the software to produce valid results for test cases encompassing the range of permitted usage defined by the program documentation. Such activities ensured that the software adequately and correctly performed all intended functions.	Uncertain	
6.3	Testing demonstrated that the compute program properly handles abnormal conditions and events as well as credible failures	Uncertain	
6.4	Testing demonstrated that the computer program does not perform adverse unintended functions.	Uncertain	
6.5	Test Phase activities were performed to assure adherence to requirements, and to assure that the software produces correct results for the test case specified. Acceptable methods for evaluating adequacy of software test case results included: (1) analysis with computer assistance; (2) other validated computer programs; (3) experiments and tests; (4) standard problems with known solutions; (5) confirmed published data and correlations.	Uncertain	
6.6	Test Phase documentation includes test procedures or plans and the results of the execution of test cases. The test results documentation demonstrates successful completion of all test cases or the resolution of unsuccessful test cases and provides direct traceability between the test results and specified software requirements.	Uncertain	
6.7	Test procedures or plans specify the following, as applicable: required tests and test sequence, required range of input parameters, identification of the stages at which testing is required, requirements for testing logic branches, requirements for hardware integration, anticipated output values, acceptance criteria, reports, records, standard formatting, and conventions, identification of operating environment, support software, software tools or system software, hardware operating system(s) and/or limitations.	Uncertain	

4.6.2 Sources and Method of Review

Documentation provided at the NIST sponsored CFAST website, http://fast.nist.gov/, supplemented by informal communications, was used as the basis for establishing the responses for this requirement.

4.6.3 Software Quality-Related Issues or Concerns

NIST has recently documented a verification and validation of CFAST in an internal report. When it becomes available, the conclusions in the NIST report should be included in the gap analysis.

4.6.4 Recommendations

The criteria are partially met. Thus, the requirement is not met. Recommendations related to this topical area are:

Recommendation 6.1 — Contact NIST to obtain a copy of the verification and validation report.

Recommendation 6.2 — Review recently prepared verification and validation report when it becomes available and establish a plan to identify gaps as appropriate.

4.7 Topical Area 7 Assessment: User Instructions

This area corresponds to the requirement entitled User Instructions in Table 3-3 of (**Error! Reference source not found.**).

4.7.1 Criterion Specification and Result

Table 4.7-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
7.1	A description of the model is documented.	Yes	(DOE, 2004, U.S. Department of Energy (2004). *The CFAST Computer Code Application Guidance for Documented Safety Analysis,* (May 2004). Jones, 2003, Peacock, 1993, Peacock, 2000)
7.2	User's manual or guide includes approved operating systems (for cases where source code is provided, applicable compilers should be noted).	No	Approved operating systems are not established in the users documentation.
7.3	User's manual or guide includes description of the user's interaction with the software.	Yes	(Peacock, 2000)
7.4	User's manual or guide includes a description of any required training necessary to use the software.	No	
7.5	User's manual or guide includes input and output specifications.	Partially	(DOE, 2004, U.S. Department of Energy (2004). *The CFAST Computer Code Application Guidance for Documented Safety Analysis,* (May 2004). Jones, 2003, Peacock, 1993, Peacock, 2000) See Additional Details.
7.6	User's manual or guide includes a description of software and hardware limitations.	No	
7.7	User's manual or guide includes a description of user messages initiated as a result of improper input and how the user can respond.	No	
7.8	User's manual or guide includes information for obtaining user and maintenance support.	Yes	CFAST website contains an e-mail address to request assistance.

Additional Detail

Criterion 7.5. — Three different output files provide numerical output. These include the history file (*.HI), a comma delineated file (*.csv) and a text file (*.txt). The history file is accessed by the routine CPlot, which is executed from the DOS command prompt. This program is described in Appendix C of (Peacock, 2000). The methods to produce output in the other two formats is also described in (Peacock, 2000), however explicit descriptions for all of the available output information is not published.

4.7.2 Sources and Method of Review

There are two current technical references that describe the algorithms and assumptions used in CFAST. There are (Peacock, 1993) and (DOE, 2004, U.S. Department of Energy (2004). *The CFAST Computer Code Application Guidance for Documented Safety Analysis,* (May 2004).

Jones, 2003), which cover CFAST 3.1.7 and CFAST 5.1 respectively. There is one user's guide for both versions, (Peacock, 2000). These documents are available at the NIST sponsored web site

http://cfast.nist.gov/ and were used as the basis for response to this requirement. Informal communications with NIST personnel provided additional information.

4.7.3 Software Quality-Related Issues or Concerns

As identified above, the description of the output files is limited. This can readily be addressed by preparing a description of each file type. In addition, NIST does not provide complete sample problems. While there are sample input data files provided with the initial installation, the output associated with these files are not available. An update to the user's guide is about to be published. This update has not been evaluated as part of this gap analysis.

4.7.4 Recommendations

The criteria are not met. Thus, the requirement is not met. Recommendations related to this topical area are provided as follows:

Recommendation 7.1 – The user's manual should be updated to reflect the minimum operating system requirements.

Recommendation 7.2 – DOE should establish the minimum qualification for personnel who are expected to prepare safety analyses using CFAST. (Two levels of qualification may be appropriate. The lower tier would be to operate the software and produce results, the higher tier would be to interpret the results.)

Recommendation 7.3 - A description of output files should be prepared and included in the user's manual.

Recommendation 7.4 - Sample problems that include the input data files, output data files and a discussion of the results should be provided.

Recommendation 7.5 – The user's manual should be updated to include a description of software and hardware limitations.

4.8 Topical Area 8 Assessment: Acceptance Test

This area corresponds to the requirement entitled Acceptance Test in Table 3-3 of (**Error! Reference source not found.**).

4.8.1 Criterion Specification and Result

Table 4.8-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
8.1	To the extent applicable to the developer, acceptance testing includes a comprehensive test in the operating environment(s).	No	CFAST is provided with a series of input data files that can be executed to establish if CFAST was installed successfully. Formal user instructions explaining the purpose of these files are not available.
8.2	To the extent applicable to the developer, acceptance testing was performed prior to approval of the computer program for use.	Yes	
8.3	To the extent applicable to the developer, software validation was performed to ensure that the installed software product satisfies the specified software requirements. The engineering function (i.e., an engineering operation an item is required to perform to meet the component or system design basis) determines the acceptance testing to be performed prior to approval of the computer program for use.	Yes	
8.4	Acceptance testing documentation includes results of the execution of test cases for system installation and integration, user instructions (Refer to Requirement 7 above), and documentation of the acceptance of the software for operational use.	No	

4.8.2 Sources and Method of Review

Documentation provided at the NIST sponsored CFAST website, http://fast.nist.gov/, supplemented with informal communications, was used as the basis for establishing the responses for this requirement.

4.8.3 Software Quality-Related Issues or Concerns

As identified above, there is no publicly available acceptance testing protocol associated with CFAST. In addition, there is description of the output files is limited. This can readily be addressed by preparing a description of each file type. In addition, NIST does not provide complete sample problems. While there are sample input data files provided with the initial installation, the output associated with these files are not available.

4.8.4 Recommendations

The criteria are partially met. Thus, the requirement is not met. Recommendations related to this topical area are:

Recommendation 8.1 — Work with NIST to document the existing acceptance tests and their use.

4.9 Topical Area 9 Assessment: Configuration Control

This area corresponds to the requirement entitled Configuration Control in Table 3-3 of (**Error! Reference source not found.**).

A NIST Internal Report (IR) has been prepared detailing the version update process.

4.9.1 Criterion Specification and Result

Table 4.9-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
9.1	For the developers the methods used to control, uniquely identify, describe, and document the configuration of each version or update of a computer program (for example, source, object, back-up files) and its related documentation (for example, software design requirements, instructions for computer program use, test plans, and results) are described in implementing procedures.	Yes	CFAST is labeled and documented for release as Version 3.1.7 and 5.1. A NIST IR has been prepared detailing the version update process.
9.2	Implementing procedures meet applicable criteria for configuration identification, change control and configuration status accounting.	Yes	NIST IR has not been reviewed, however it is assumed to be adequate.

4.9.2 Sources and Method of Review

Documentation provided at the NIST sponsored CFAST website, http://fast.nist.gov/, supplemented with informal communications, was used as the basis for establishing the responses for this requirement.

4.9.3 Software Quality-Related Issues or Concerns

There is no publicly available description of the configuration control process that is in place for CFAST.

4.9.4 Recommendations

This requirement is met, however recommendations are provided to ensure that a comprehensive documentation package can be compiled. Recommendations related to this topical area are:

Recommendation 9.1 — Contact NIST to obtain a copy of the NIST internal report documenting the version update process.

Recommendation 9.2 — Review the existing NIST report documenting the version update process when it becomes available and establish a plan to identify gaps as appropriate.

4.10 Topical Area 10 Assessment: Error Impact

This area corresponds to the requirement entitled Error Impact in Table 3-3 of (**Error! Reference source not found.**).

This section is based on informal communications with the software developer.

4.10.1 Criterion Specification and Result

Table 4.10-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
10.1	The developing organization's problem reporting and corrective action process addresses the appropriate requirements of its corrective action system and is documented in implementing procedures.	No	NIST does not maintain a formal error notification system, however, NIST does gather comments, question, error reports and fix them as needed. See Additional Detail.
10.2	The process for evaluating, and documenting whether a reported problem is an error is documented and implemented.	No	See Criterion 10.1 summary remarks.
10.3	The process for disposition of the problem reports, including notification to the originator of the results of the evaluation, is documented and implemented.	No	See Criterion 10.1 summary remarks.
10.4	A documented process provides guidance on determining how identified errors relate to appropriate software engineering elements and is implemented.	No	See Criterion 10.1 summary remarks.
10.5	The process is documented and implemented for determining how an error impacts past and present use of the computer program.	No	See Criterion 10.1 summary remarks.
10.6	The process is documented and implemented for determining how an error and resulting corrective action impacts previous development activities.	No	See Criterion 10.1 summary remarks.
10.7	The process is documented and implemented describing how the users are notified of an identified error, its impact; and how to avoid the error, pending implementation of corrective actions.	Partial	A version history maintained on the CFAST web site.

Additional Detail

Criterion 10.1 — The NIST web site www.cfast.nist.gov contains a statement "If you need information or help with features not covered here or in the Technical Reference or User's Guide, please send the request to cfast@nist.gov". This address, and its linked companion, "inquiries@fire.gov," serve as a collection point for CFAST user feedback. Any errors that might be identified through this process are prioritized and addressed by the NIST staff.

4.10.2 Sources and Method of Review

Documentation provided at the NIST sponsored CFAST website, http://fast.nist.gov/, supplemented with informal communications, was used as the basis for establishing the responses for this requirement.

On March 1, 2004, CFAST 5.1 was issued. This version corrected a discrepancy between the software calculation and the technical reference manual. The discrepancy was identified through the normal information exchanges between NIST staff and CFAST users. (See Appendix C.)

4.10.3 Software Quality-Related Issues or Concerns

There is no formal error reporting or notification system for CFAST. However, the issuance of CFAST 5.1 demonstrates that the NIST error management program fulfills the intent of criterion 10.1 through 10.5 and criterion 10.7. The weakness in the error impact management process resides with how facilities with existing analyses are notified to initiate a corrective action addressing the error (criterion 10.6).

4.10.4 Recommendations

The criteria are considered to be partially met based on the effectiveness of the recent CFAST update (5.1). Thus, the requirement is not met. Recommendations related to this topical area are provided as follows:

Recommendation 10.1 — Establish an Error Impact Management Process plan.

4.11 Training Program Assessment

NIST does not offer user training for CFAST, however the Society of Fire Protection Engineers (SFPE) has offered such training. While the course is not currently scheduled, SFPE will bring the course to clients when requested. A description of this training is presented in Appendix B. Training has also been offered through Worchester Polytechnical Institute. The link for information on this class is: http://www.wpi.edu/Academics/Depts/Fire/Courses/FP570/CFAST%20slides_files/frame.htm.

4.12 Software Improvements

A graphical user interface for version 5 is being developed to be compatible with Windows XP. The CFAST web site provides access to an Alpha version (0.9a).

Seneca College in Ontario, Canada hosts a discussion forum for CFAST and FDS. The web address presenting information on this forum is at http://fireforum.senecac.on.ca.

CFAST has the capability to track contaminate migration explicitly. If CFAST is modified it will be possible to use this feature to support Leak Path Factor (LPF) analysis.

Recommendation 12.1 — Support the development of a GUI for CFAST 5.1 by contributing to CFAST users groups.

Recommendation 12.2 — Fund NIST to modify CFAST to establish LPF values utilizing the contaminate term (CT keyword).

5.0 Conclusions

The gap analysis for Versions 3.1.7 and 5.1 of the CFAST software, based on a set of requirements and criteria compliant with NQA-1, has been completed. Of the ten primary SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), two requirements are met at acceptable level: *Classification* (1) and *Configuration Control* (9). Five requirements are considered to be partially met: *Implementation Phase* (5), *Testing Phase* (6), *User Instructions* (7), *Acceptance Test* (8), and *Error Notification and Corrective Action* (10). Three requirements are not met *SQA Procedures and Plans (2), Requirements Phase (3), and Design Phase (4)*. Improvement actions are recommended for CFAST to fully meet eight of the requirements. This evaluation outcome is deemed acceptable because: (1) CFAST 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 CFAST is limited to those analytic applications for which the software is intended.

For requirement 10, *Error Impact*, NIST has demonstrated an Error Management Process that successfully evaluates and corrects significant identified errors. The only significant shortcomings in the process based on the criteria stated in Table 4.10-1 are a lack of formality and a notification mechanism that results in a corrective action by operating facilities that have used output from a CFAST analysis in the development of a DSA (Criterion 10.6).

It was determined that CFAST code does meet its intended function for use in supporting documented safety analysis. However, as with all safety-related software, users should be aware of current limitations and capabilities of CFAST for supporting safety analysis. Informed use of the software can be assisted by the current set of CFAST reports (refer to Table 1-3), and the code guidance report for DOE safety analysts, *CFAST Computer Code Application Guidance for Documented Safety Analysis*, (DOE, 2004). Furthermore, while SQA improvement actions are recommended for CFAST, no evidence has been found of programming, logic, or other types of software errors in CFAST that have led to non-conservatisms in nuclear facility operations or in the identification of facility controls.

By order of priority, it is recommended that CFAST software improvement actions be taken, especially:

- Revising software documentation and user instructions to provide a comprehensive description of the software output (Section 4.7).
- Establishing an acceptance test protocol to be used to assure that the installed version of CFAST is working properly when software is installed on a new computer system (Section 4.8)
- Defining the minimum training necessary to use the software and offering the training on a regular basis (Section 4.7)
- Implementing a formal error notification and corrective action process (Section 4.10).

Performing these four primary actions should satisfactorily improve the SQA compliance status of CFAST relative to the evaluation requirements cited in this report.

It is estimated that approximately 0.5 full-time equivalent year (FTE) would be required to fulfill the first three SQA recommendations described in Section 2.2, including

- The CFAST Users Manual does not provide a comprehensive description of the software output (Section 4.7).
- A description of the training necessary to use the software is not available (Section 4.7)

- An acceptance test protocol to be used to assure that the installed version of CFAST is working properly is not documented (Section 4.8).

Several more FTE-months are estimated to address other non-compliant areas discussed in Sections 4.1 through 4.10.

Approximately one FTE-month per year would be needed to maintain a web-based error notification and corrective action process for CFAST (Section 4.10). However, such a process has not been defined in depth for CFAST and the other designated toolbox codes.

6.0 Acronyms and Definitions

Acronyms

ALOHA	Areal Locations of Hazardous Atmospheres (designated toolbox software)
ANS	American Nuclear Society
ANSI	American National Standards Institute
ASME	American Society of Mechanical Engineers
CFAST	Consolidated Fire and Smoke Transport Model (designated toolbox software)
CFR	Code of Federal Regulations
DNFSB	Defense Nuclear Facilities Safety Board
DoD	Department of Defense
DOE	Department of Energy
DSA	Documented Safety Analysis
EPIcode	Emergency Prediction Information code (designated toolbox software)
GENII	Generalized Environmental Radiation Dosimetry Software System - Hanford Dosimetry System (Generation II) (designated toolbox software)
IEEE	Institute of Electrical and Electronics Engineers
LPF	Leak Path Factor
MACCS2	MELCOR Accident Consequence Code System 2 (designated toolbox software)
MELCOR	Methods for Estimation of Leakages and Consequences of Releases (designated toolbox software)
NIST	National Institute of Standards and Technology
NRC	Nuclear Regulatory Commission
ODE	Ordinary Differential Equation
RSICC	Radiation Safety Information Computational Center
SFPE	Society of Fire Protection Engineers
SSC	Safety Analysis and Design Software
SQA	Software Quality Assurance
SQAP	Software Quality Assurance Plan

Definitions

The following definitions are taken from the Implementation Plan. References in brackets following definitions indicate the original source, when not the Implementation Plan.

Acceptance Testing — The process of exercising or evaluating a system or system component by manual or automated means to ensure that it satisfies the specified requirements and to identify differences between expected and actual results in the operating environment. [NQA-1]

Central Registry — An organization designated to be responsible for the storage, control, and long-term maintenance of the Department's safety analysis "toolbox codes." The central registry may also perform this function for other codes if the Department determines that this is appropriate.

Configuration Management —The process that controls the activities, and interfaces, among design, construction, procurement, training, licensing, operations, and maintenance to ensure that the configuration of the facility is established, approved and maintained. (Software specific): The process of identifying and defining the configuration items in a system (i.e., software and hardware), controlling the release and change of these items throughout the system's life cycle, and recording and reporting the status of configuration items and change requests. [NQA-1]

Design Requirements — Description of the methodology, assumptions, functional requirements, and technical requirements for a software system.

Error —A condition deviating from an established base line, including deviations from the current approved computer program and its baseline requirements. [NQA-1]

Firmware — The combination of a hardware device and computer instructions and data that reside as read-only software on that device. [IEEE Standard 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology]

Gap Analysis — Evaluation of the Software Quality Assurance attributes of specific computer software against identified criteria.

Independent Verification and Validation — Verification and validation performed by an organization that is technically, managerially, and financially independent of the development organization.

Nuclear Facility — A reactor or a nonreactor nuclear facility where an activity is conducted for or on behalf of DOE and includes any related area, structure, facility, or activity to the extent necessary to ensure proper implementation of the requirements established by 10 CFR 830. [10 CFR 830]

Operating Environment — A collection of software, firmware, and hardware elements that provide for the execution of computer programs. [NQA-1]

Safety Analysis and Design Software – Computer software that is not part of a structure, system, or component (SSC) but is used in the safety classification, design, and analysis of nuclear

facilities to ensure the proper accident analysis of nuclear facilities; the proper analysis and design of safety SSCs; and, the proper identification, maintenance, and operation of safety SSCs. [DOE O 414.1B]

Safety Software — Includes both safety system software, and safety analysis and design software. [DOE O 414.1B]

Safety System Software — Computer software and firmware that performs a safety system function as part of a structure, system, or component (SSC) that has been functionally classified as Safety Class (SC) or Safety Significant (SS). This also includes computer software such as human-machine interface software, network interface software, programmable logic controller (PLC) programming language software, and safety management databases that are not part of an SSC but whose operation or malfunction can directly affect SS and SC SSC function. [DOE O 414.1B]

Software — Computer programs, operating systems, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. [IEEE Standard 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology]

Software Engineering — The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software; also: the study of these applications. [NQA-1]

Source Code — A computer code in its originally coded form, typically in text file format. For programs written in a compilable programming language, the uncompiled program.

System Software —Software designed to enable the operation and maintenance of a computer system and its associated computer programs. [NQA-1]

Test Plan (Procedure) —A document that describes the approach to be followed for testing a system or component. Typical contents identify the items to be tested, tasks to be performed, and responsibilities for the testing activities. [NQA-1]

Testing --An element of verification for the determination of the capability of an item to meet specified requirements by subjecting the item to a set of physical, chemical, environmental, or operating conditions. [NQA-1]

Toolbox Codes — A small number of standard computer models (codes) supporting DOE safety analysis, having widespread use, and of appropriate qualification that are maintained, managed, and distributed by a central source. Toolbox codes meet minimum quality assurance criteria. They may be applied to support 10 CFR 830 DSAs provided the application domain and input parameters are valid. In addition to public domain software, commercial or proprietary software may also be considered. In addition to safety analysis software, design codes may also be included if there is a benefit to maintain centralized control of the codes [modified from DOE N 411.1].

User Manual — A document that presents the information necessary to employ a system or component to obtain desired results. Typically described are system or component capabilities, limitations, options, permitted inputs, expected outputs, possible error messages, and special instructions. Note: A user manual is distinguished from an operator manual when a distinction is made between those who operate a computer system (mounting tapes, etc.)

and those who use the system for its intended purpose. Synonym: User Guide. [IEEE 610-12]

Validation – 1. The process of testing a computer program and evaluating the results to ensure compliance with specified requirements [ANSI/ANS-10.4-1987].
2.The process of determining the degree to which a model is an accurate representation of the real-world from the perspective of the intended uses of the model [Department of Defense Directive 5000.59, *DoD Modeling and Simulation (M&S) Management*]

Verification – 1. The process of evaluating the products of a software development phase to provide assurance that they meet the requirements defined for them by the previous phase [ANSI/ANS-10.4-1987].
2.The process of determining that a model implementation accurately represents the developer's conceptual description and specifications [Department of Defense Directive 5000.59, *DoD Modeling and Simulation (M&S) Management*].

7.0 References

Alvord, 1995, *A CFAST Output Comparison Method and its use in Comparing Different CFAST Versions*, Gaithersburg: MD. National Institute of Standards and Technology, NISTIR 5705 (August).

CFR, 2001, *Nuclear Safety Management*, Washington, DC: Department of Energy. (1 January) 10 CFR 830. 2001.

DNFSB, 2000, Defense Nuclear Facilities Safety Board, (2000). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, Technical Report DNFSB/TECH-25, (January).

DNFSB, 2002, Defense Nuclear Facilities Safety Board, (2002). *Recommendation 2002-1, Quality Assurance for Safety-Related Software*. Washington, DC: Defense Nuclear Facilities Safety Board. (September).

DOE, 2002, U.S. Department of Energy (2002). *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports*, DOE-STD-3009-94, Change Notice 2 (April).

DOE, 2003a, U.S. Department of Energy (2003). *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).

DOE, 2003b, U.S. Department of Energy (2003). *Designation of Initial Safety Analysis Toolbox Codes*, Letter, (March 28).

DOE, 2003c, *Software Quality Assurance Improvement Plan: Format and Content for Code Guidance Reports*. (2003). Washington, DC: US Department of Energy (August).

DOE, 2003d, U.S. Department of Energy (2003). *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes*, (November 2003).

DOE, 2004, U.S. Department of Energy (2004). *The CFAST Computer Code Application Guidance for Documented Safety Analysis*, (May 2004).

Jones, 2003, Jones, Walter W., Glenn P. Forney, Richard D. Peacock and Paul A. Reneke. (2003). *A Technical Reference for CFAST: An Engineering Tool for Estimating Fire and Smoke Transport*. Gaithersburg: MD. National Institute of Standards and Technology. (April) NIST TN 1431.

NRC, 2002, *Evaluation of Fire Models for Nuclear Power Plant Applications: Cable Tray Fires*. 2002. Washington, DC: US Nuclear Regulatory Commission, Office of Nuclear Regulatory Research, Division of Risk Analysis and Applications. NUREG-1758.

Peacock, 1993, Peacock, R. D., Paul A. Reneke, Walter W. Jones, Rebecca M. Portier, and Glenn P. Forney. (1993). *CFAST, the Consolidated Model of Fire Growth and Smoke Transport*. Gaithersburg: MD. National Institute of Standards and Technology. (February) NIST Technical Note 1299.

Peacock, 2000, Peacock, R. D., Paul A. Reneke, Walter W. Jones, Richard W. Bukowski, and Glenn P. Forney. (2000). *A User's Guide for FAST: Engineering Tools for Estimating Fire Growth*

and Smoke Transport. Gaithersburg: MD. National Institute of Standards and Technology. (January) NIST Special Publication 921, 2000 edition.

Appendices

Appendix	Subject
A	Software Information Template
B	SFPE Training Class Descriptions
C	CFAST Revision Notice

APPENDIX A.— SOFTWARE INFORMATION TEMPLATE

The following is a condensed version of the information request sent to the CFAST code developer in October 2003. *(Note: This information is provided to give the reader of this Gap report, an idea of the information requested to complete the Gap analysis for CFAST. Detailed information in response was not filled in. See Section 1.6. Instead, the contacts and the Gap authors used the form as a guide for continual discussion throughout the Gap analysis for CFAST.*

Information Form

Development and Maintenance of Designated Safety Analysis Toolbox Codes

The following summary information in Table 2 should be completed to the level that is meaningful – enter N/A if not applicable. See Appendix A for an example of the input to the table prepared for the MACCS2 code.

Table 2. Summary Description of Subject Software

Table 2. Summary Description of Subject Software	
Type	**Specific Information**
Code Name	
Version of the Code	
Developing Organization and Sponsor Information	
Auxiliary Codes	
Software Platform/Portability	
Coding and Computer(s)	
Technical Support Point of Contact	
Code Procurement Point of Contact	
Code Package Label/Title	

Table 2. Summary Description of Subject Software	
Type	**Specific Information**
Contributing Organization(s)	
Recommended Documentation - Supplied with Code Transmittal upon Distribution or Otherwise Available	1. 2. 3. 4. 5.
Input Data/Parameter Requirements	
Summary of Output	
Nature of Problem Addressed by Software	
Significant Strengths of Software	
Known Restrictions or Limitations	
Preprocessing (set-up) time for Typical Safety Analysis Calculation	
Execution Time	
Computer Hardware Requirements	
Computer Software Requirements	
Other Versions Available	

Table 3. Point of Contact for Form Completion

Individual(s) completing this information form: Name: Organization: Telephone: Email: Fax:	

1. Software Quality Assurance Plan

The software quality assurance plan for your software may be either a standalone document, or embedded in other documents, related procedures, QA assessment reports, test reports, problem reports, corrective actions, supplier control, and training package.

1.a For this software, identify the governing Software Quality Assurance Plan (SQAP)?
[Please submit a PDF of the SQAP, or send hard copy of the SQAP[1]]

1.b What software quality assurance industry standards are met by the SQAP?

1.c What federal agency standards were used, if any, from the sponsoring organization?

1.d Has the SQAP been revised since the current version of the Subject Software was released? If so, what was the impact to the subject software?

1.e Is the SQAP proceduralized in your organization? If so, please list the primary procedures that provide guidance.

Guidance for SQA Plans:

Requirement 2 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a)

[1] Notify Kevin O'Kula of your intent to send hard copies of requested reports and shipping will be arranged.

ASME NQA-1 2000 Section 200
IEEE Standard 730, *IEEE Standard for Software Quality Assurance Plans.*
IEEE Standard 730.1, *IEEE Guide for Software Quality Assurance Planning.*

2. Software Requirements Description

The software requirements description (SRD) should contain functional and performance requirements for the subject software. It may be contained in a standalone document or embedded in another document, and should address functionality, performance, design constraints, attributes and external interfaces.

2.a For this software, was a software requirements description documented with the software sponsor? [If available, please submit a PDF of the Software Requirements Description, or include hard copy with transmittal of SQAP]

2.b If a SRD was not prepared, are there written communications that indicate agreement on requirements for the software? Please list other sources of this information if it is not available in one document.

Guidance for Software Requirements Documentation:

Requirement 5 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 401
IEEE Standard 830, *Software Requirements Specifications*

3. Software Design Documentation

The software design documentation (SDD) depicts how the software is structured to satisfy the requirements in the software requirements description. It should be defined and maintained to ensure that software will serve its intended function. The SDD for the subject software may be contained in a standalone document or embedded in another document.

The SDD should provide the following:

- Description of the major components of the software design as they relate to the software requirements,
- Technical description of the software with respect to the theoretical basis, mathematical model, control flow, data flow, control logic, and data structure,
- Description of the allowable or prescribed ranges of inputs and outputs,
- Design described in a manner suitable for translating into computer coding, and

- Computer program listings (or suitable references).

3.a **For the subject software, was a software design document prepared, or were its constituents parts covered elsewhere?** [If available, please submit a PDF of the Software Design Document, or include hard copy with transmittal of SQAP]

3.b **If the intent of the SDD information is satisfied in other documents, provide the appropriate references (document number, section, and page number).**

Guidance for Software Design Documentation:

Requirement 6 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
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*
IEEE Standard 1012, *IEEE Standard for Software Verification and Validation;*
IEEE Standard 1012a, *IEEE Standard for Software Verification and Validation – Supplement to 1012*

4. Software User Documentation

Software User Documentation is necessary to assist the user in installing, operating, managing, and maintaining the software, and to ensure that the software satisfies user requirements. At minimum, the documentation should describe:

- The user's interaction with the software
- Any required training
- Input and output specifications and formats, options
- Software limitations
- Error message identification and description, including suggested corrective actions to be taken to correct those errors, and
- Other essential information for using the software.

4.a **For the subject software, has Software User Documentation been prepared, or are its constituents parts covered elsewhere?** [If available, please submit a PDF of the Software User Documentation, or include a hard copy with transmittal of SQAP]

4.b **If the intent of the Software User Documentation information is satisfied in other documents, provide the appropriate references (document number, section, and page number).**

4.c Training – How is training offered in correctly running the subject software? Complete the appropriate section in the following:

Type	Description	Frequency of training
Training Offered to User Groups as Needed		
Training Sessions Offered at Technical Meetings or Workshops		
Training Offered on Web or Through Video Conferencing		
Other Training Modes		
Training Not Provided		

Guidance for Software User Documentation:

Requirement 9 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 203
IEEE Standard 1063, *IEEE Standard for Software User Documentation*

5. Software Verification &Validation Documentation (Includes Test Reports)

Verification and Validation (*V&V*) documentation should confirm that a software V&V process has been defined, that V&V has been performed, and that related documentation is maintained to ensure that:

(a) The software adequately and correctly performs all intended functions, and
(b) The software does not perform any unintended function.

The software V&V documentation, either as a standalone document or embedded in other documents and should describe:

- The tasks and criteria for verifying the software in each development phase and validating it at completion,
- Specification of the hardware and software configurations pertaining to the software V&V
- Traceability to both software requirements and design
- Results of the V&V activities, including test plans, test results, and reviews (also see 5.b below)
- A summary of the status of the software's completeness
- Assurance that changes to software are subjected to appropriate V&V,
- V&V is complete, and all unintended conditions are dispositioned before software is approved for use, and
- V&V performed by individuals or organizations that are sufficiently independent.

5.a For the subject software, identify the V&V Documentation that has been prepared.
[If available, please submit a PDF of the Verification and Validation Documentation, or include a hard copy with transmittal of SQAP]

5.b If the intent of the V&V Documentation information is satisfied in one or more other documents, provide the appropriate references (document number, section, and page number). For example, a "Test Plan and Results" report, containing a plan for software testing, the test results, and associated reviews may be published separately.

5.c Testing of software: What has been used to test the subject software?

☐ Experimental data or observations
☐ Standalone calculations
☐ Another validated software
☐ Software is based on previously accepted solution technique

Provide any reports or written documentation substantiating the responses above.

Guidance for Software Verification & Validation, and Testing Documentation:

Requirement 6 – *Design Phase* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
Requirement 8 – *Testing Phase* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
Requirement 10 – *Acceptance Test* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 402 (Note: Some aspects of verification may be handled as part of the Design Phase).
ASME NQA-1 2000 Section 404 (Note: Aspects of validation may be handled as part of the Testing Phase).
IEEE Standard 1012, *IEEE Standard for Software Verification and Validation;*
IEEE Standard 1012a, *IEEE Standard for Software Verification and Validation -- Supplement to 1012*
IEEE Standard 829, *IEEE Standard for Software Test Documentation*.
IEEE Standard 1008, *Software Unit Testing*

6. Software Configuration Management (SCM)

A process and related documentation for SCM should be defined, maintained, and controlled.

The appropriate documents, such as project procedures related to software change controls, should verify that a software configuration management process exists and is effective.

The following points should be covered in SCM document(s):

- A Software Configuration Management Plan, either in standalone form or embedded in another document,
- Configuration management data such as software source code components, calculational spreadsheets, operational data, run-time libraries, and operating systems,
- A configuration baseline with configuration items that have been placed under configuration control,
- Procedures governing change controls,
- Software change packages and work packages to demonstrate that (1) possible impacts of software modifications are evaluated before changes are made, (2) various software system products are examined for consistency after changes are made, and (3) software is tested according to established standards after changes have been made.

6.a **For the subject software, has a Software Configuration Management Plan been prepared, or are its constituent parts covered elsewhere?** [If available, please submit a PDF of the Software Configuration Management Plan and related procedures, or include hard copies with transmittal of SQAP].

6.b **Identify the process and procedures governing control and distribution of the subject software with users.**

6.c Do you currently interact with a software distribution organization such as the Radiation Safety Information Computational Center (RSICC)?

6.d A Central Registry organization, under the management and coordination of the Department of Energy's Office of Environment, Safety and Health (EH), will be responsible for the long-term maintenance and control of the safety analysis toolbox codes for DOE safety analysis applications. Indicate any questions, comments, or concerns on the Central Registry's role and the maintenance of the subject software.

Guidance for Software Configuration Management Plan Documentation:

Requirement 12 – *Configuration Control* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 203
IEEE Standard 828, *IEEE Standard for Software Configuration Management Plans.*

7. Software Problem Reporting and Corrective Action

Software problem reporting and corrective action documentation help ensure that a formal procedure for problem reporting and corrective action development for software errors and failures is established, maintained, and controlled.

A Software Error Notification and Corrective Action Report, procedure, or similar documentation, should be implemented to report, track, and resolve problems or issues identified in both software items, and in software development and maintenance processes. Documentation should note specific organizational responsibilities for implementation. Software problems should be promptly reported to affected organizations, along with corrective actions. Corrective actions taken ensure that:

- Problems are identified, evaluated, documented, and, if required, corrected,
- Problems are assessed for impact on past and present applications of the software by the responsible organization,
- Corrections and changes are executed according to established change control procedures, and
- Preventive actions and corrective actions results are provided to affected organizations.

Identify documentation specific to the subject software that controls the error notification and corrective actions. [If available, please submit a PDF of the Error

Notification and Corrective Action Report documentation for the subject software (or related procedures). If this is not available, include hard copies with transmittal of SQAP].

7.aProvide examples of problem/error notification to users and the process followed to address the deficiency. Attach files as necessary.

7.bProvide an assessment of known errors or defects in the subject software and the planned action and time frame for correction.

Category of Error or Defect	Corrective Action	Planned schedule for correction
Major		
Minor		

7.cIdentify the process and procedures governing communication of errors/defects related to the subject software with users.

Guidance for Error/Defect Reporting and Corrective Action Documentation:

Requirement 13 – *Error Impact* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))

ASME NQA-1 2000 Section 204
IEEE Standard 1063, *IEEE Standard for Software User Documentation*

8. Resource Estimates

If one or more plans, documents, or sets of procedures identified in parts one (1) through seven (7) do not exist, please provide estimates of the resources (full-time equivalent (40-hour) weeks, FTE-weeks) and the duration (months) needed to meet the specific SQA requirement.

Enter estimate in Table 4 only if specific document has not been prepared, or requires revision.

Table 4. Resource and Schedule for SQA Documentation

Plan/Document/Procedure	Resource Estimate (FTE-weeks)	Duration of Activity (months)
1. Software Quality Assurance Plan		
2. Software Requirements Document		
3. Software Design Document		
4. Test Case Description and Report		
5. Software Configuration and Control		
6. Error Notification and Corrective Action Report		
7. User's Instructions (User's Manual)		
8. Other SQA Documentation		

Comments or Questions:

9. Software Upgrades

Describe modifications planned for the subject software.

Technical Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		

5.		

User Interface Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Software Engineering Improvements

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Other Planned Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Thank you for your input to the SQA upgrade process. Your experience and insights are critical towards successfully resolving the issues identified in DNFSB Recommendation 2002-1.

APPENDIX B.— SFPE TRAINING CLASS DESCRIPTIONS

Introduction to Computer Fire Modeling

Intended for: This seminar is intended for fire protection engineers with a desire to develop a basic understanding of models used to predict the characteristics of compartment fire growth and the operation of fire protection systems. Attendees are expected to bring a laptop with a copy of FPEtool installed, details will be provided upon registration. Attendees will receive a set of class notes and selected reading and reference materials.

Seminar Description: This seminar provides an introduction to computer fire modeling and the underlying fire science. The fundamental driving force for fire modeling and design calculations is the heat release rate history of the burning objects. The basic fire science of compartment fire development is presented along with specific computer models or tools. Attendees will be given problems to solve independently to gain experience in use of the models. Problems will involve: detector and sprinkler activation, fire growth and spread, smoke and gas flow and an introduction to human behavior and egress. Limitations of the methodologies presented will be discussed. The seminar will employ case studies and conclude with demonstration of FASTlite. Participants will receive a detailed course notebook.

Seminar Outline:

- Introduction to Computer Modeling?
- Heat Release Rate
- Ignition and Flame Spread
- Flow Through Cents
- Fire/Wind/Stack Forces on Doors
- Zone Fire Modeling Theory
- General Limitations of Zone Models
- Plume and Jet Temperatures
- Sprinkler and Detector Response
- Upper Layer Temperature
- ASET-B Room Fire
- Modeling the Occupants
- Modeling Sprinkler Suppression
- FASTlite

Advanced Computer Fire Modeling

Intended for: This seminar is intended for fire protection engineers who have a basic understanding of models used to predict the characteristics of compartment fire growth and the operation of fire protection systems and are seeking to apply these methods to fire protection engineering analysis and design. *Attendees are expected to bring a laptop with copies of FAST installed*. Other software may be used as well. Software and installation details will be provided upon registration. Attendees will receive a set of class notes and selected reading and reference materials.

Description: This seminar assumes a basic understanding of computer fire modeling and the underlying fire science. This seminar will expand on the methods introduced in *Introduction to*

Computer Fire Modeling, providing alternative approaches and discussion of how to select the right model for the job. Limitations of the methodologies presented will be discussed. Computer fire modeling is the basis for predicting fire effects for performance-based design. Attendees will be given problems to solve that will involve working from floor plans, setting design/performance criteria, developing design fires and selecting and evaluating design alternatives. The seminar will employ case studies and conclude with a discussion of computational fluid dynamics (CFD) fire modeling.

Outline:

- Introduction
- Toxic Species Modeling
- How to Select Your Model
- Performance-Based Design Criteria
- Plume and Jet Equations
- Design Application Case Studies
- Detection Issues
- Design Problems
- Modeling Effects of Suppression
- Overview of CFD
- Human Response Models
- Single & Multi-Compartment Modeling

APPENDIX C.— CFAST REVISION NOTICE

Reference: http://www.cfast.nist.gov/documents/V5p1Update.pdf

Version 5.1, dated March 1, 2004. This version fixes the oxygen key word (O2) so that the calculation follows the technical reference manual. In version 5.0 and earlier, the oxygen calculation used the oxygen to fuel ratio, whereas the technical reference manual states that it uses the oxygen to carbon ratio. The model now matches the guide.

Note 1. The combustion chemistry is based on the oxygen consumption calorimetry of Huggett,et al.[1] It is important that the species key word values and the heat of combustion be consistent. Since there is no fundamental kinetic calculation in CFAST, there is no way for the model to check the consistency. For example, a heat of combustion of 50 MJ per kilogram matches a hydrogen/carbon ratio of 0.3. Using 24 MJ with a HCR of 0.3 will yield incorrect results and can also result in the model stalling.

Note 2. When a layer is driven to zero volume, there is no way to provide species by percent, since the total mass is zero. In this case, CFAST reports 0%. This can be seen with the data file specieserror.dat. Once the upper layer is larger than the minimum volume, the species can be normalized correctly and reported.

[1] Clayton Huggett, Fire and Materials, Vol. 4, No. 2, 61-65, June 1980.

SEPARATION

PAGE

Defense Nuclear Facilities Safety Board Recommendation 2002-1
Software Quality Assurance Improvement Plan
Commitment 4.2.1.3:

Software Quality Assurance Improvement Plan: MACCS2 Gap Analysis

Final Report

U.S. Department of Energy
Office of Environment, Safety and Health
1000 Independence Ave., S.W.
Washington, DC 20585-2040

May 2004

INTENTIONALLY BLANK

FOREWORD

This report documents the outcome of an evaluation of the Software Quality Assurance (SQA) attributes of the radiological dispersion computer code, MACCS2, relative to established software requirements. This evaluation, a "gap analysis", is performed to meet commitment 4.2.1.3 of the Department of Energy's Implementation Plan to resolve SQA issues identified in the Defense Nuclear Facilities Safety Board Recommendation 2002-1.

Suggestions for corrections or improvements to this document should be addressed to –

Chip Lagdon
EH-31/GTN
U.S. Department of Energy
Washington, D.C. 20585-2040
Phone (301) 903-4218
Email: chip.lagdon@eh.doe.gov

INTENTIONALLY BLANK

REVISION STATUS

Page/Section	Revision	Change
1. Entire Document	1. Interim Report	1. Original Issue
2. Entire Document	2. Final Report, May 3, 2004	2. Updated all sections per review comments.

INTENTIONALLY BLANK

CONTENTS

Section			Page
FOREWORD			III
REVISION STATUS			V
EXECUTIVE SUMMARY			XIII
1.0	INTRODUCTION		1-1
	1.1	BACKGROUND: OVERVIEW OF DESIGNATED TOOLBOX SOFTWARE IN THE CONTEXT OF 10 CFR 830	1-1
	1.2	EVALUATION OF TOOLBOX CODES	1-1
	1.3	USES OF THE GAP ANALYSIS	1-2
	1.4	SCOPE	1-2
	1.5	PURPOSE	1-3
	1.6	METHODOLOGY FOR GAP ANALYSIS	1-3
	1.7	SUMMARY DESCRIPTION OF SOFTWARE BEING REVIEWED	1-6
2.0	ASSESSMENT SUMMARY RESULTS		2-1
	2.1	CRITERIA MET	2-1
	2.2	EXCEPTIONS TO REQUIREMENTS	2-1
	2.3	AREAS NEEDING IMPROVEMENT	2-3
	2.4	MACCS2 ISSUES CITED IN TECH-25 AND RECOMMENDED APPROACHES FOR RESOLUTION	2-5
	2.5	CONCLUSION REGARDING SOFTWARE'S ABILITY TO MEET INTENDED FUNCTION	2-6
3.0	LESSONS LEARNED		3-1
4.0	DETAILED RESULTS OF THE ASSESSMENT PROCESS		4-1
	4.1	TOPICAL AREA 1 ASSESSMENT: SOFTWARE CLASSIFICATION	4-2
		4.1.1 Criterion Specification and Result	*4-2*
		4.1.2 Sources and Method of Review	*4-3*
		4.1.3 Software Quality-Related Issues or Concerns	*4-3*
		4.1.4 Recommendations	*4-3*
	4.2	TOPICAL AREA 2 ASSESSMENT: SQA PROCEDURES AND PLANS	4-3
		4.2.1 Criterion Specification and Result	*4-6*
		4.2.2 Sources and Method of Review	*4-7*
		4.2.3 Software Quality-Related Issues or Concerns	*4-8*
		4.2.4 Recommendations	*4-8*
	4.3	TOPICAL AREA 3 ASSESSMENT: REQUIREMENTS PHASE	4-8
		4.3.1 Criterion Specification and Results	*4-8*
		4.3.2 Sources and Method of Review	*4-9*
		4.3.3 Software Quality-Related Issues or Concerns	*4-9*
		4.3.4 Recommendations	*4-9*

4.4 TOPICAL AREA 4 ASSESSMENT: DESIGN PHASE 4-10
4.4.1 Criterion Specification and Result *4-10*
4.4.2 Sources and Method of Review *4-12*
4.4.3 Software Quality-Related Issues or Concerns *4-13*
4.4.4 Recommendations *4-13*
4.5 TOPICAL AREA 5 ASSESSMENT: IMPLEMENTATION PHASE 4-14
4.5.1 Criterion Specification and Result *4-14*
4.5.2 Sources and Method of Review *4-14*
4.5.3 Software Quality-Related Issues or Concerns *4-14*
4.5.4 Recommendations *4-15*
4.6 TOPICAL AREA 6 ASSESSMENT: TESTING PHASE 4-15
4.6.1 Criterion Specification and Result *4-15*
4.6.2 Sources and Method of Review *4-16*
4.6.3 Software Quality-Related Issues or Concerns *4-16*
4.6.4 Recommendations *4-17*
4.7 TOPICAL AREA 7 ASSESSMENT: USER INSTRUCTIONS 4-17
4.7.1 Criterion Specification and Result *4-17*
4.7.2 Sources and Method of Review *4-18*
4.7.3 Software Quality-Related Issues or Concerns *4-18*
4.7.4 Recommendations *4-18*
4.8 TOPICAL AREA 8 ASSESSMENT: ACCEPTANCE TEST 4-19
4.8.1 Criterion Specification and Result *4-19*
4.8.2 Sources and Method of Review *4-20*
4.8.3 Software Quality-Related Issues or Concerns *4-20*
4.8.4 Recommendations *4-20*
4.9 TOPICAL AREA 9 ASSESSMENT: CONFIGURATION CONTROL 4-21
4.9.1 Criterion Specification and Result *4-21*
4.9.2 Sources and Method of Review *4-21*
4.9.3 Software Quality-Related Issues or Concerns *4-21*
4.9.4 Recommendations *4-22*
4.10 TOPICAL AREA 10 ASSESSMENT: ERROR IMPACT 4-22
4.10.1 Criterion Specification and Result *4-22*
4.10.2 Sources and Method of Review *4-24*
4.10.3 Software Quality-Related Issues or Concerns *4-24*
4.10.4 Recommendations *4-24*
4.11 TRAINING PROGRAM ASSESSMENT 4-24
4.12 SOFTWARE IMPROVEMENTS AND NEW BASELINE 4-25

5.0 CONCLUSIONS 5-1

6.0 ACRONYMS AND DEFINITIONS 6-1

7.0 REFERENCES 7-1

APPENDIX A. SOFTWARE INFORMATION TEMPLATE A-1

APPENDIX B. OPERATION AND MAINTENANCE CRITERION B-1

B.1 TOPICAL AREA ASSESSMENT: OPERATION AND MAINTENANCE PHASE B-1
B.1.1 Criterion Specification and Results *B-1*
B.1.2 Sources and Method of Review *B-2*
B.1.3 Software Quality-Related Issues or Concerns *B-2*
B.1.4 Recommendations *B-2*

INTENTIONALLY BLANK

TABLES

	Page
Table 1-1. — Plan for SQA Evaluation of Existing Safety Analysis Software	1-4
Table 1-2. — Summary Description of MACCS2 Software	1-7
Table 1-3 — Software Documentation Reviewed for MACCS2	1-10
Table 2-1 — Summary of Important Exceptions, Reasoning, and Suggested Remediation	2-1
Table 2-2 — Summary of Recommendations for MACCS2	2-5
Table 3-1 — Lessons Learned	3-1
Table 4.0-1 — Cross-Reference of Requirements with Subsection and Entry from DOE (2003e)	4-1
Table 4.1-1 — Subset of Criteria for Software Classification Topic and Results	4-3
Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results	4-7
Table 4.3-1 — Subset of Criteria for Requirements Phase Topic and Results	4-8
Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results	4-10
Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results	4-14
Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results	4-15
Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results	4-17
Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results	4-19
Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results	4-21
Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results	4-22
Table 4.12-1. — Comparison of NQA-1 Requirements, with SQA Upgrade Steps Discussed in Bixler (2000) with the Approach Discussed in DOE (2003e)	4-27

INTENTIONALLY BLANK

Software Quality Assurance Improvement Plan: MACCS2 Gap Analysis

EXECUTIVE SUMMARY

The Defense Nuclear Facilities Safety Board (DNFSB) issued Recommendation 2002-1 on *Quality Assurance for Safety-Related Software* in September 2002 (DNFSB 2002). The Recommendation identified a number of quality assurance issues for software used in the 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 high-use, Software Quality Assurance (SQA)-compliant safety analysis codes is one of the major improvement actions discussed in the *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities*. A DOE safety analysis toolbox would contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed and maintained for DOE-broad safety basis applications.

The MACCS2 software, for radiological dispersion and consequence analysis, is one of the codes designated for the toolbox. To determine the actions needed to bring the MACCS2 code into compliance with the SQA qualification criteria, and develop an estimate of the resources required to perform the upgrade, the Implementation Plan has committed to sponsoring a code-specific gap analysis document. The gap analysis evaluates the software quality assurance attributes of MACCS2 against identified criteria.

The balance of this document provides the outcome of the MACCS2 gap analysis compliant with NQA-1-based requirements as contained in U.S. Department of Energy, *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes*, (DOE, 2003e). It was determined that MACCS2 code does meet its intended function for use in supporting documented safety analysis. However, as with all safety-related software, users should be aware of current limitations and capabilities of MACCS2 for supporting safety analysis. Informed use of the software can be assisted by the current set of MACCS2 reports (refer to Table 1-3), and the code guidance report for DOE safety analysts, *MACCS2 Computer Code Application Guidance for Documented Safety Analysis*, (DOE, 2004). Furthermore, while SQA improvement actions are recommended for MACCS2, no evidence has been found of programming, logic, or other types of software errors in MACCS2 that have led to non-conservatisms in nuclear facility operations, or in the identification of facility controls.

Of the ten primary SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), two requirements are met at an acceptable level, i.e., *Classification* (1) and *User Instructions* (7). A third requirement, *Error Notification and Corrective Action* (10), is partially met. Improvement actions are recommended for MACCS2 to fully meet requirement (10) criteria, and the remaining seven requirements. This evaluation outcome is deemed acceptable 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.

By order of priority, it is recommended that MACCS2 software improvement actions be taken, especially:

1. correct known defects
2. upgrade user technical support activities
3. provide training on a regular basis, and
4. revise software documentation.

Performing these four primary actions should satisfactorily improve the SQA compliance status of MACCS2 relative to the primary evaluation criteria cited in this report.

A new software baseline set of documents is recommended for MACCS2 to demonstrate completion of item 4 (above), revise software documentation. The list of baseline documents for revision includes:

1. Software Quality Assurance Plan
2. Software Model Description, including, but not limited to,
    a. Software Requirements
    b. Software Design
3. User's Manual, including, but not limited to,
    a. User Instructions
    b. Test Case Description and Report
    c. Software Configuration and Control
4. Error Notification and Corrective Action Procedure.

Additionally, user documentation should be augmented to include error diagnostic advice and suggested input files for prototypic nuclear facility safety analysis problem types. Approximately two full-time equivalent years is conservatively estimated to upgrade MACCS2 software to be compliant with NQA-1-based requirements for existing software. While most of this effort is logically to be used by the code developer, independent review of the end products is necessary.

A new version of MACCS2, Version 1.13, has recently been released. It is recommended that this version be evaluated relative to the software improvement and baseline document recommendations, as well as the full set of SQA criteria discussed in this report. If this version is found to be satisfactory, it should replace Version 1.12 as the designated version of the software for the toolbox.

It is recommended that MACCS2 user training for DOE safety analysis applications be conducted formally on, at minimum, an annual basis. Prerequisites for, and core knowledge needed by, the user prior to initiating MACCS2 applications should be documented by the code developer.

Approximately one FTE-month per year would be needed to maintain a web-based error notification and corrective action process for MACCS2 (Section 4.10). However, such a process has not been defined in depth for MACCS2 and the other designated toolbox codes.

1.0 Introduction

This document reports the results of a gap analysis for Version 1.12 of the MACCS2 computer code. The intent of the gap analysis is to determine the actions needed to bring the specific software into compliance with established Software Quality Assurance (SQA) criteria. A secondary aspect of this report is to develop an estimate of the level of effort required to upgrade MACCS2 based on the gap analysis results.

1.1 Background: Overview of Designated Toolbox Software in the Context of 10 CFR 830

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* (DNFSB, 2000). 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.

While progress was made in resolving several of the issues raised in TECH-25, the DNFSB issued Recommendation 2002-1 on *Quality Assurance for Safety-Related Software* in September 2002. 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, the quality of the software used to design or develop safety-related controls, and the proficiency of personnel using the software. The Recommendation identified a number of quality assurance issues for software used in the 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 high-use, SQA-compliant safety analysis codes is one of the major commitments contained in the March 2003 *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities* (IP). In time, the DOE safety analysis toolbox will contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed and maintained for DOE-broad safety basis applications.

Six computer codes, including ALOHA (chemical release dispersion/consequence analysis), CFAST (fire analysis), EPIcode (chemical release dispersion/consequence analysis), GENII (radiological dispersion/consequence analysis), MACCS2 (radiological dispersion/consequence analysis), and MELCOR (leak path factor analysis), have been designated by DOE for the toolbox (DOE/EH, 2003). It is found that this software provides generally recognized and acceptable approaches for modeling source term and consequence phenomenology, and can be applied as appropriate to support accident analysis in Documented Safety Analyses (DSAs).

As one of the designated toolbox codes, MACCS2 Version 1.12, will likely require some degree of quality assurance improvement before meeting current SQA standards. The analysis documented herein is an evaluation of MACCS2 relative to current software quality assurance criteria. It assesses the margin of the deficiencies, or gaps, to provide DOE and the software developer the extent to which minimum upgrades are needed. The overall assessment is therefore termed a "gap" analysis.

1.2 Evaluation of Toolbox Codes

The quality assurance criteria identified in later sections of this report are defined as the set of established requirements, or bases, by which to evaluate each designated toolbox code. This gap analysis evaluation, is commitment 4.2.1.3 in the IP:

> Perform a SQA evaluation to the toolbox codes to determine the actions needed to bring the codes into compliance with the SQA qualification criteria, and develop a schedule with milestones to upgrade each code based on the SQA evaluation results.

This process is a prerequisite step for software improvement. It will allow DOE to determine the current limitations and vulnerabilities of each code as well as help define and prioritize the steps required for improvement.

Early in the SQA evaluation program, it was anticipated that each toolbox code owner would provide input information on the SQA programs, processes, and procedures used to develop their software. However, most of the designated toolbox software, including MACCS2, was developed without complete conformance to software quality standards. Furthermore, many of the software developer organizations cannot confirm that key processes were followed. Therefore, most of the SQA evaluation has been preceded with reconstructing software development processes based on anecdotal evidence and limited, supporting documentation.

For independence reasons, the gap analysis is performed by a SQA evaluator not affiliated with the MACCS2 development program. While independent of the code developer, the SQA evaluators responsible for MACCS2 are knowledgeable in the use of the software for accident analysis applications, and understand current software development standards.

1.3 Uses of the Gap Analysis

The gap analysis provides key information to DOE, code developers, and code users.

DOE obtains the following benefits:

- Estimates of the resources required to perform modifications to designated toolbox codes
- Basis for schedule and prioritization to upgrade each designated toolbox code.

Each code developer is provided:

- Information on areas where software quality assurance improvements are needed to comply with industry SQA standards and practices
- Specific areas for improvement to guide development of new versions of the software.

DOE safety analysts and code users benefit from:

- Improved awareness of the strengths, limits, and vulnerable areas of each computer code
- Recommendations for code use in safety analysis application areas.

1.4 Scope

The gap analysis is applicable to the MACCS2 code, one of the six designated toolbox codes for safety analysis. While MACCS2 is the subject of the current report, other safety analysis software considered for the toolbox in the future may be evaluated with the same process applied here. The template outlined

in this document is applicable for any analytical software as long as the primary criteria are ASME NQA-1, 10 CFR 830, and related DOE directives discussed in DOE (2003e).

The scope of this review did not include auxiliary software referenced in Section 1.7. Dose conversion factor and food pathway software are included with the transmittal of MACCS2 software from the Radiation Safety Information Computational Center (RSICC).

1.5 Purpose

The purpose of this report is to document the gap analysis performed on the MACCS2 code as part of DOE's implementation plan on SQA improvements.

1.6 Methodology for Gap Analysis

The gap analysis for MACCS2 is based on the plan and criteria described in *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes* (DOE 2003e). The overall methodology for the gap analysis is summarized in Table 1-1. The gap analysis utilizes ten of the fourteen topical areas listed in DOE (2003e) related to software quality assurance to assess the MACCS2 software. The ten areas are those particularly applicable to the software development, specifically: (1) Software Classification, (2) SQA Procedures/Plans, (5) Requirements Phase, (6) Design Phase, (7) Implementation Phase, (8) Testing Phase, (9) User Instructions, (10) Acceptance Test, (12) Configuration Control, and (13) Error Impact. Each area or requirement, is assessed individually in Section 4.

Requirements 3 (Dedication), 4 (Evaluation), 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 evaluating the areas listed above. Requirement 11 (Operation and Maintenance) is only partially applicable to software development, and is interpreted to be applicable mostly to the software user organization. Several comments on this requirement are covered as an appendix.

Table 1-1. — Plan for SQA Evaluation of Existing Safety Analysis Software[1]

Phase	Procedure
1. Prerequisites	a. Determine whether sufficient information is provided by the software developer 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 Procedure, 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.

[1] Originally documented as Table 2-2 in DOE (2003e).

Phase	Procedure
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 Procedure, 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.
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.

An information template was transmitted to the Safety Analysis Software Developers in October 2003 to provide basic information as input to the gap analysis process. The main section of the template is attached as Appendix A to the present report, with an example section and references removed. While no written response to the information template was received from the MACCS2 software developers at Sandia National Laboratories (SNL), three sources of information compensated for the lack of a formal reply:

- **Review by East** – A review of MACCS2 SQA program by East (1998b) for the Department of Energy Office of Defense Programs (now National Nuclear Security Administration) was used to assess the MACCS2 program especially during the 1995 – 1997 period.
- **Site visit by SQA Evaluators** – A site visit was made to SNL on 21 January 2004 to meet with MACCS2 software team. The software information template and the kinds of inputs needed to complete the gap analysis report, preliminary findings and observations, and expected recommendations were covered.
- **Discussions and input from MACCS2 consultant** – The SQA evaluation team contacted the primary consultant to the MACCS and MACCS2 efforts, David Chanin. Although not currently associated with the MACCS2 program, Mr. Chanin supplemented the information obtained from SNL.

1.7 Summary Description of Software Being Reviewed

The gap analysis was performed on Version 1.12 of the MACCS2 code. MACCS2 (Chanin, 1998) is a radiological atmospheric dispersion and consequence code, and is written in FORTRAN 77 and 90. The software is maintained by Sandia National Laboratories (SNL) and is an update to MACCS.[2] Since the issuance of DOE-STD-3009-94 for nuclear facility accident analysis, MACCS2 has been used for DOE applications primarily as a tool for deterministic consequence analysis. The output of MACCS2 is used to support decision-making on control selection in nuclear facilities, specifically identification of safety structures, systems, and components (SSCs).

MACCS2 predicts dispersion of radionuclides by the use of a multiple, straight-line Gaussian plume, Eulerian model. The direction, duration, sensible heat, and initial radionuclide concentration may be varied from plume to plume. Crosswind dispersion is treated by a multi-step function and both wet and dry depositions features can be modeled as independent processes. For DSA applications, the MACCS2 user can apply either the Latin Hypercube Sampling (LHS) mode or the stratified random sampling mode to process one year of site-specific meteorological data. Based on the meteorological sampling of site-specific data, and application of user-specified dose and/or health effects models, complementary cumulative distribution functions (CCDFs) are calculated for various measures of consequence. The average, median, 95th, and 99.5th percentile doses are provided in the output. A technical and operational summary of the MACCS2 software is contained in Table 1-2.

The set of MACCS2- and MACCS-specific documents reviewed as part of the gap analysis are listed in Table 1-3. The SNL software developers provided Reference 11 (Proposal to Resolve QA Deficiencies in MACCS2) as part of a Safety Analysis Software Group activity in response to TECH-25. Reference 13 (NP 19-1) was provided to support the current assessment. Other documentation was previously received from SNL or RSICC, or gleaned from the technical literature.

[2] The United States Nuclear Regulatory Commission (NRC) sponsored the development of the MACCS code (Chanin, 1990; Jow, 1990; Rollstin, 1990; and Chanin, 1993) as a successor to the CRAC2 code for the performance of commercial nuclear industry probabilistic safety assessments (PSAs). The MACCS code was used in the NUREG-1150 PSA study (NRC, 1990a) in the early 1990's. Prior to the code being released to the public, the MACCS code was independently verified by Idaho National Engineering and Environmental Laboratory (Dobbe, 1990). After verification, the NRC released MACCS, Version 1.5.11 for general distribution. Examples of MACCS applied in this period include commercial reactor PSAs (both U.S. and international), as well as non-reactor nuclear facilities (primarily U.S.).

Table 1-2. — Summary Description of MACCS2 Software

Type	Specific Information
Code Name	MACCS2 - MELCOR Accident Consequence Code System for the Calculation of the Health and Economic Consequences of Accidental Atmospheric Radiological Releases
Developing Organization and Sponsor	Sandia National Laboratories (SNL) for the U.S. Nuclear Regulatory Commission (primary) and U.S. Department of Energy (minor)
Version of the Code	Version 1.12
Auxiliary Codes	AUXILIARY CODES: DOSFAC2:NRC dose conversion factor (DCF) preprocessor. FGRDCF: DCF preprocessor based on the DCF databases of Federal Guidance Reports 11 and 12 from ORNL (DLC-172). IDCF2: DCF preprocessor based on the IDCF code developed at the Idaho National Engineering Laboratory. COMIDA2: Food pathway preprocessor based on the COMIDA (PSR-343) food pathway preprocessor developed at the Idaho National Engineering Laboratory. Note: MELMACCS (MACCS input generator from MELCOR runs) and CHAIN (Radionuclide progeny) are auxiliary codes, and not available from RSICC. CHAIN was developed by Keith Eckerman at ORNL.
Software Platform/Portability	FORTRAN 77/90, PC based some system dependencies
Coding and Computer	Fortran 77, PC based 80486 or Pentium processor (C00652/PC486/00).
Technical Support	Nathan Bixler Sandia National Laboratories P.O. Box 5800 Albuquerque, NM 87185-0748 (505) 845-3144 nbixler@sandia.gov;
Code Procurement	Radiation Safety Information Computational Center (RSICC)[3] Oak Ridge National Laboratory Post Office Box 2008 Bethel Valley Road Oak Ridge, Tennessee 37831-6171 Phone: 865-574-6176; Fax: 865-241-4046 Email: pdc@ornl.gov; Internet: http://www-rsicc.ornl.gov/rsicc.html Contact Nathan Bixler (above) or Jocelyn Mitchell @ NRC for authorization Phone: 301-415-5289 Email: jam@nrc.gov

[3] Recommended procurement route is through N. Bixler/J. Mitchell (see below). Except where noted, items shown here are valid when MACCS2 is obtained through RSICC.

Table 1-2. Summary Description of MACCS2 Software (Continued)

Code Package Identification at RSICC	CCC-652; Included are the references cited below and the Fortran source code, executables and data, which are distributed on 1 CD in self-extracting compressed DOS files.
Contributors	Sandia National Laboratories, Albuquerque, New Mexico, Oak Ridge National Laboratory, Oak Ridge, Tennessee, Idaho National Engineering and Environmental Laboratory, Idaho Falls, Idaho.
Documentation Supplied with Code Transmittal[4]	1. D. Chanin and M. L. Young, "Code Manual for MACCS2, User's Guide," NUREG/CR-6613, Vol. 1, SAND97-0594 (May 1998), Sandia National Laboratories, Albuquerque, NM. 2. D. Chanin and M. L. Young, "Code Manual for MACCS2, Preprocessor Codes COMIDA2, FGRDCF, IDCF2," NUREG/CR-6613, Vol. 2, SAND97-0594 (May 1998), Sandia National Laboratories, Albuquerque, NM. 3. M. L. Young and D. Chanin, "DOSFAC2 User's Guide," NUREG/CR-6547, SAND97-2776 (December 1997). 4. H-N. Jow, J. L. Sprung, J. A. Rollstin, L. T. Ritchie, D. I. Chanin, "MELCOR Accident Consequence Code System (MACCS), Model Description," NUREG/CR-4691, SAND86-1562, Vol. 2 (February 1990). 5. J. Gregory, "Software Defect Notifications" (May 1998). 6. M. L. Young, "READMAC2.txt" (April 1997).
Nature of Problem	MACCS2 simulates the impact of accidental atmospheric releases of radiological materials on the surrounding environment. This package is a major enhancement of the previous CCC-546/MACCS 1.5.11 package. The principal phenomena considered in MACCS are atmospheric transport, mitigative actions based on dose projection, dose accumulation by a number of pathways including food and water ingestion, early and latent health effects, and economic costs. MACCS can be used for a variety of applications including probabilistic risk assessment (PRA) of nuclear power plants and other nuclear facilities, sensitivity studies to gain a better understanding of the parameters important to PRA, and cost benefit analysis.

[4] Transmittal of MACCS2 directly from SNL includes training slides on use of MACCS2.

Table 1-2. Summary Description of MACCS2 Software (Continued)

Restrictions or Limitations	The atmospheric model included in the code does not model the impact of terrain effects on atmospheric dispersion, nor can it accept more than one weather spatial location. Like all Gaussian models, MACCS2 is not well suited for modeling dispersion close to the source (less than 100 meters from the source) or long-range dispersion (beyond 15 to 20 miles from the source).[5] Momentum effects of highly energetic releases can be approximated. The economic model included in the code models only the economic cost of mitigative actions.
Run Time	One source term for one meteorological sequence requires less than one second on a Pentium 2 or 3 GHZ. Running two source terms and sampling a year of weather data (Sample Problem A) requires approximately times on the order of seconds to minutes.
Computer Hardware Requirements	IBM-compatible 486/DX or Pentium PC with 8 MB of RAM The MACCS2 package files require approximately the following disk space when decompressed: MAC2ZIPA.EXE 6 MB MAC2ZIPB.EXE 4 MB FGR_DCF.EXE 2 MB COMIDA2A.EXE 3 MB IDCF_2.EXE 2 MB DOSFAC_2.EXE 4 MB COMIDA2B.EXE 3 MB. Approximately 30 MB of hard disk space is required to load the complete MACCS2 package. Approximately 11 MB of hard disk space is required to load MACCS2 without the preprocessors included in the MACCS2 package.
Computer Software Requirements	The MACCS2 software was developed in a DOS environment. Lahey F77L-EM/32 Version 5.2 compiler was used to create the executables included in the code transmittal package from RSICC, which run successfully in a DOS window of Windows 3.1, Windows 95, Windows NT, and Windows 2000. The programs can also be compiled for those PC operating systems with the Microsoft Powerstation FORTRAN 1.0a compiler. The distributed executables will not run under Windows XP. However, upon request, the code developer will supply executables for Windows XP that were compiled using Compaq FORTRAN 95.
Other Versions Available	MACCS 1.5.11.1 (PC486); MACCS 1.5.11.0 (IBM RISC); Version 1.13 was released in March 2004 to RSICC.

[5] Typical PRA calculations often apply a 1000-mile radius basis.

Table 1-3 — Software Documentation Reviewed for MACCS2

No.	Reference
1.	Chanin, 1998, D. Chanin and M. Young, *Code Manual for MACCS2: Volume 1, User's Guide;*, M. Young, D. Chanin, and V. Banjac, *DOSFAC2 User's Guide*, NUREG/CR-6613, SAND97-0594, May 1998, Sandia National Laboratories, Albuquerque, NM.
2.	Chanin, 1998, D. Chanin and M. Young, *Code Manual for MACCS2, Volume 2, Pre-Processor Codes COMIDA2, FGRDCF, IDCF2*; May 1998, NUREG/CR-6613, SAND97-0594, May 1998, Sandia National Laboratories, Albuquerque, NM.
3.	Young, 1997, M. Young, D. Chanin, and V. Banjac, *DOSFAC2 User's Guide*, NUREG/CR-6613, SAND97-0594, December 1997, Sandia National Laboratories, Albuquerque, NM.
4.	Chanin, 1990, D.I. Chanin, J.L. Sprung, L.T. Ritchie, H-N Jow, and J.A. Rollstin, *MELCOR Accident Consequence Code System (MACCS). Volume 1: User's Guide*; H-N Jow, J.L. Sprung, J.A. Rollstin, L.T. Ritchie, and D.I. Chanin, *Volume 2: Model Description*; J.A. Rollstin, D.I. Chanin, and H-N Jow, *Volume 3: Programmer's Reference Manual*; NUREG/CR-4691, Sandia National Laboratories, published by the U.S. Nuclear Regulatory Commission, Washington, DC, 1990.
5.	Chanin, 1992a, D. Chanin, J. Rollstin, J. Foster, and L. Miller, *MACCS Version 1.5.11.1: A Maintenance Release of the Code*, Sandia National Laboratories, Albuquerque, NM, July 14, 1992.
6.	Chanin, 1992b, D.I. Chanin, *A New Emergency Response Model for MACCS*, LA-SUB-94-67, prepared by Teledyne Engineering Consultants, Inc., Albuquerque, NM for Los Alamos National Laboratory, Los Alamos, NM, November 11, 1992.
7.	Dobbe 1990, C.A. Dobbe, E.R. Carlson, N.H. Marshall, E.S. Marwil, J.E. Tolli. *Quality Assurance and Verification of the MACCS Code, Version 1.5,* Idaho National Engineering Laboratory, Idaho Falls, ID, NUREG/CR-5376 (EGG-2566)
8.	DNFSB, 2000, 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).
9.	Gregory, 1998, J. Gregory, "Software Defect Notifications" (May 1998).
10.	WSRC, 1998, Westinghouse Savannah River Company, *MACCS Input Guidance for SRS Applications (U)*, WSRC-RP-98-00978, (October 1998).
11.	East, 1998a, J.M. East, *Verification of MACCS2 Peak Dose Output (U)*, WSRC-TR-97-204, Westinghouse Savannah River Company, Aiken, SC (July 1998).

Table 1-3 — Software Documentation Reviewed for MACCS2 (continued)

No.	Reference
12.	East, 1998b, J.M. East and E.P. Hope, *Independent Evaluation of the MACCS2 Software Quality Assurance Program (U)*, WSRC-RP-98-00712, Westinghouse Savannah River Company, Aiken, SC (August 1998).
13.	LANL, 2001 Draft, Los Alamos National Laboratory, *LANL Guidelines for Performing Atmospheric Dispersion Analysis*, Operational Support Tool 300-00-06H, Los Alamos, NM (June 2001).
14.	Bixler, 2000, N. Bixler, *Proposal to Resolve QA Deficiencies in MACCS2*, Informal Memorandum to D. Chung (DOE/DP), Sandia National Laboratories, Albuquerque, NM (2000).
15.	DOE, 2004, U.S. Department of Energy. *MACCS2 Computer Code Application Guidance for Documented Safety Analysis*, Interim Report, (2004). For latest DOE/EH Central Registry report, go to: http://www.eh.doe.gov/sqa/central_%20registry/MACCS2/maccs2.htm
16.	SNL, 2003, Sandia National Laboratories. *Nuclear Waste Management Procedure*, NP 19-1, *Software Requirements*, Revision 10, Waste Isolation Pilot Plant, (May 2003).
17.	Summa, 1996 Draft, F.J. Summa and F.E. Haskin. *Pre-Release Verification Testing of the MACCS2 Code.* University of New Mexico, Albuquerque, NM. Unpublished report prepared for Sandia National Laboratories (1996).
18.	Chanin, 1997, D. Chanin, *Software Quality Assurance Procedures Followed with MACCS2*, Letter to K. O'Kula (September 1997).
19.	Gregory, 1998, J. Gregory, *Software Defect Notification.* Sandia National Laboratories, Albuquerque, NM (1998).

2.0 Assessment Summary Results

2.1 Criteria Met

Of the ten general topical quality areas assessed in the gap analysis, two satisfactorily met the criteria. The analysis found that the MACCS2 SQA program, in general, met criteria for *Software Classification* and *User Instructions*, Requirements 1 and 7, respectively. A third topical quality area, *Error Notification*, partially met criteria, but it and the remaining seven topical quality areas were judged either not wholly compliant with the SQA criteria, and/or lacked documentation to confirm compliance. The eight areas that should be addressed for improvement actions are listed in Section 2.2 (Exceptions to Requirements). Detail on the evaluation process relative to the requirements, and the criteria applied, are found in Section 4.

2.2 Exceptions to Requirements

Some of the more important exceptions to criteria found for MACCS2 are listed below in Table 2-1. The requirement is given, the reason the requirement was not met is provided, and remedial action(s) are listed to correct the exceptions. The ten criteria evaluated are those predominantly executed by the software developer. However, it is noted that criteria for SQA Procedures/Plan, Testing, Acceptance Test, Configuration Control, and Error Notification also have requirements for the organization implementing the software. These criteria were assessed in the present evaluation only from the code developer perspective.

Table 2-1 — 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	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

No.	Criterion	Reason Not Met	Remedial Action(s)
		was never completed.	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 report.
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.
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	While a Software Problem Reporting system is apparently functional at SNL, written documentation should be provided to

No.	Criterion	Reason Not Met	Remedial Action(s)
		documentation was provided.	demonstrate its effectiveness.

2.3 Areas Needing Improvement

The gap analysis, communications with DOE, oversight organizations, safety analysts, and inputs from the long-term MACCS/MACCS2 users have identified a number of improvements that could be made to the code's technical model and its quality assurance. The major areas to be addressed are described in this section.

Multiple-plume release. The software upgrade that should be addressed as soon as possible is that impacting calculations containing multiple plume segments (Gregory 1998). Essentially the same coding error affects calculations with multiple emergency response assumptions, e.g. Sample Problem A. Other identified errors in the MACCS2 software, while deserving corrective action as part of good SQA processes and practices, are insignificant relative to most DOE DSA applications. *SNL has indicated that Version 1.13 of the code addresses known errors, especially the multiple plume and multiple emergency response inputs. Version 1.13 was released in March 2004 to the RSICC software center.*

Multiple versions of MACCS2. There are instances reported of multiple versions of MACCS2 having been disseminated over the last five years. This is not good practice from a software configuration control perspective. Multiple versions arose, in part, as a result of dissemination of instructions on how users can correct the source-term and emergency-response looping errors, combined with the lack of a code update to RSICC after considerable time had elapsed since discovery of these errors. It is recommended that all capabilities be made available through one common distribution site, such as the DOE Central Registry, or the Radiation Safety Information Computational Center (RSICC).

User Interface. Other modifications are recommended on a less urgent basis. Included are improvements to the user interface. MACCS2 still uses a DOS-based operating system, and requires experienced user experience and insights to correctly build a usable input file. A U.S. NRC-sponsored upgrade for MACCS2, MACCS3, will improve this feature by developing a Windows©-based system (Bixler, 2000). The new Windows-GUI based "front end" will preserve the core of the code as a standalone executable that can be run in a MS-DOS command window under various versions of Windows in the same manner as MACCS2. The executable for MACCS3 is designed to be "backward-compatible" and allows the use of input files created for MACCS2.

DSA Dispersion/Dose Analysis. Using MACCS2 to quantify 95th percentile direction-independent doses to receptors at non-equidistant locations is treated differently throughout the DOE Complex. Several sites have developed post-processing routines to approach the requirements of Appendix A to DOE-STD-3009-94. This situation is not ideal because it leaves the calculation of doses to be completed with various approaches, rather than standardizing within a configuration controlled version. A modest effort should be undertaken to identify the best approach for performing this type calculation in MACCS2, and making the option available to all users through SNL or RSICC distribution.

Source Term Types. The treatment of several source term types important to DOE applications could be improved in MACCS2. Sensible heat algorithms for modeling fire source terms have been implemented for some customers, but systematic treatment of this phenomenology should be standardized in the version of the code available to all DOE users. The current model is limited and may be non-conservative unless combined with a building wake effect model (DOE, 2004). The code developers could add an option developed by Mills (1987). The Mills fire model was implemented in a variant of MACCS2 used by Pantex Plant. SNL may include the new fire model in successors to MACCS2. Additionally, the code

as built was not intended to treat deflagration/detonation events. While MACCS2 may not be suitable for mechanistically modeling highly energetic source terms, User's Manual documentation should be expanded to include methods of modeling these events using Gaussian plume modeling (Steele 1998).

Additionally, MACCS2 documentation would be improved by listing user requirements, i.e. prerequisites to meet before running the software. While a set of problems are provided with transmittal of MACCS2, it would be useful to identify the basic knowledge and skill level necessary to meaningfully run the software.

The user interface should be improved in the area of diagnostics. Currently, software diagnostic messages are difficult to interpret and do not provide the user with sufficient understanding of incorrect input or suggested fixes.

Finally, it is suggested that sample problems be augmented to provide guidance in performing primary types of analyses frequently encountered in safety analysis. Suggested are spill, fire, deflagration, and criticality types of releases.

Other user options for treating various aspects of dispersion phenomenology can be explored in future versions of MACCS2. These include treatment of multi-year meteorological data sets, and expanding model selection for building wake effects, low wind speed conditions, plume trajectory, puff/plume rise behavior, mixing layer penetration, resuspension, and dry deposition. While expanded user options would be useful to the DOE consequence analyst, they are not critical to completing current analyses.

Several dose factor options are now available. MACCS2 documentation should provide discussion on the use of different dose conversion factor data sets and provide several default options. Several issues on the user interface and dose conversion factor file look-up by MACCS2 are outside the scope of this report and shall be addressed in the code guidance report (DOE, 2004).

The key recommendations for improvements to MACCS2 are summarized in Table 2-2.

Table 2-2 — Summary of Recommendations for MACCS2

No.	UI – User Interface Enhancements TM – Technical Model Upgrade	Recommendation
1.	TM	Correct software to allow multiple plume calculations
2.		Multiple versions of MACCS2 should be curtailed. Existing special purpose versions should be made available to all users through SNL and/or RSICC.
3.	UI	Update User interface (planned as part of USNRC program) and make available to all users.
4.	TM	Add DOE-STD-3009-94 Appendix A Post-Processing Algorithm for 95th Percentile, Direction-Independent Doses
5.	TM	Extend sensible energy model to account for area releases (e.g. pool) as well as stack releases with momentum effects.
6.	TM	Improve detonation/deflagration (explosive release) approach in code and/or documentation
7.	User Requirements	Document knowledge and skill level for users.
8.	UI	Provide Error Diagnostic guidance.
9.	UI	Expand selection of sample problems to include those problem and source term type that are often treated for DOE nuclear facility safety analysis.
10.	TM	Consider multiple year option to better sample site data sets that are greater than one year in length.
11.	TM	Allow use of multi-year meteorological data sets. Consider expanding model selection for close-in or building wake effects, low wind speed conditions, plume trajectory, puff/plume rise behavior, mixing layer penetration, resuspension, and dry deposition.
12.	UI	Document options for using sets of dose conversion factors (DCFs). Provide default databases with older and more current DCFs.

2.4 MACCS2 Issues Cited in TECH-25 and Recommended Approaches for Resolution

Four broad technical issues were explicitly noted in TECH-25 that centered on the MACCS2 software. This section discusses the four main issues and recommended dispositioning.

- Phenomenology: The fire plume model may be non-conservative. It is recommended that the current treatment be carefully used in MACCS2, taking into account building wake effects, sensible energy and spatial dependence of the source term and combustible loading. As a long-term consideration, area source models, such as that proposed by Mills (1987) for pool fire analysis could be made available as a user-specified option in MACCS2. (This topic was addressed in Section 2.3).
- Coding Errors: Software defects encountered exercising (1) multiple plume segments and (2) the emergency response model, should be addressed immediately by the code developers. A maintenance version with the major defects corrected should be made available to RSICC. A similar strategy was used for the predecessor software to MACCS2, MACCS, in creating

Version 1.5.11.1. In the interim, DOE user guidance should be applied to avoid these conditions in MACCS2 (DOE, 2004).

- End User Quality Assurance Problem: Dose conversion factors are user-specified data file input options in MACCS2. For example, non-conservative inputs for plutonium radionuclides can be unintentionally selected by users. It is recommended that user instructions (user's manual) address this potential pitfall in running MACCS2. In addition, enhanced training on the options in MACCS2 for dose factor file selection is recommended. (This topic was addressed in Section 2.3).
- Poor Documentation: Documentation for MACCS2 should be revised as part of the new software baseline. In particular, the user's guide should provide sample input files for various types of "standard" problem types encountered in both reactor and non-reactor nuclear facility safety analysis. (This topic was addressed in Section 2.3).

2.5 Conclusion Regarding Software's Ability to Meet Intended Function

The MACCS2 code was evaluated to determine if the software, in its current state, meets the intended function in a safety analysis context as assessed in this gap analysis. When the code is run for the intended applications as detailed in the code guidance document, MACCS*2 Computer Code Application Guidance for Documented Safety Analysis*, (DOE 2004) and also utilizing information from documentation available from SNL and other sources (Table 1-3), it is judged that it will meet the intended function.

Current software concerns and issues can be avoided by understanding MACCS2 limitations and capabilities. The software can be applied for modeling those types of scenarios where precedents exist, and there is confidence that alternative analysis or experimental data would adequately confirm the code predictions.

Confidence in MACCS2 to meet its intended function is expected to increase with the release of Version 1.13. The software developer has indicated that Version 1.13 corrects known errors in Version 1.12.

3.0 Lessons Learned

Table 3-1 provides a summary of the lessons learned during the performance of the MACCS2 gap analysis.

Table 3-1 — Lessons Learned

No.	Lesson
1.	Use of NQA-1 or other SQA criteria could not be fully verified. It is obvious that many actions characteristic of sound SQA practices have been applied in developing MACCS2, but independent confirmation of the SQA program, practices, and procedures is not possible. The principal reason for this outcome was that key files were not maintained.
2.	Observance of SQA requirements in the development of safety analysis software such as MACCS2 has not been consistent. Previous versions of the code were more consistent with SNL SQA practices. MACCS2 SQA for Version 1.12 lacks a documented record of many of the key processes used to develop the new code. Anecdotal evidence is available, but this type of information is difficult to verify.
3.	While some evidence of pre-development planning is found for early versions of the MACCS2 software, documentation is not maintained as would be expected for compliance with Quality Assurance criteria in Subpart A to 10 CFR 830 (Nuclear Safety Management).
4.	A new software baseline can be produced with "modest" resources (~2 full-time equivalent years) and should be a high priority. The main products will be a well-integrated set of code manuals providing the detailed design and model description information consistent with current SQA standards.
5.	Additional opportunities and venues should be sought for training and user qualification on safety analysis software. This is a long-term deficiency that needs to be addressed for MACCS2 and other designated software for the DOE toolbox.
6.	MACCS2, as well as some of the other software designated for the DOE Safety Analysis Toolbox, is characterized by its developers as a best-estimate, "research tool", and therefore without the compelling need to be compliant with stringent SQA requirements. The original and current sponsor of the MACCS development effort, the Nuclear Regulatory Commission, did not specify rigid SQA requirements be followed, because PSA applications were (and are) intended for the code rather than as a tool for nuclear "limits". This perspective underestimates the benefit of robust SQA practices used *with, not after*, any scientific or engineering software development.

4.0 Detailed Results of the Assessment Process

Ten topical areas, or requirements, are presented in the assessment as listed in Table 4.0-1. Training and Software Improvements sections follow the ten topical areas. Included in the software improvements section is an estimate of the resources required to upgrade MACCS2.

In the tables that follow, the topical areas or requirements are labeled as (1.x, 2.x,...,10.x) with the first value corresponding to the topical area and the second value (x), the sequential table order of each criterion. Four qualitative values shall be used to evaluate whether a specific criterion is met:

- Yes – evidence is available to confirm that the program, practices, and/or procedures followed in developing the version of code satisfy the criterion.
- No – sufficient evidence does not exist to demonstrate that the code meets the criterion
- Partial – some evidence exists that the criterion is met, but has not been finalized or is incomplete
- Uncertain – no basis is available to confirm that the criterion is met.

The overall evaluation for a specific requirement is based on the evaluation of the software against the criteria.

Table 4.0-1 — Cross-Reference of Requirements with Subsection and Entry from DOE (2003e)

Subsection (This Report)	**Corresponding Entry Table 3-3 from DOE (2003e)**	**Requirement**	**ASME NQA-1 2000 Section/Consensus Standards**
4.1	1	Software Classification	ASME NQA-1 2000 Section 200
4.2	2	SQA Procedures/Plans	ASME NQA-1 2000 Section 200; IEEE Std. 730, *IEEE Standard for Software Quality Assurance Plans*
4.3	5	Requirements Phase	ASME NQA-1 2000 Section 401; IEEE Standard 830, *Software Requirements Specifications*
4.4	6	Design Phase	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*
4.5	7	Implementation Phase	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*
4.6	8	Testing Phase	ASME NQA-1 2000 Section 404; IEEE Std. 829, *IEEE Standard for Software Test Documentation;*

			IEEE Standard 1008, *Software Unit Testing*
4.7	9	User Instructions	ASME NQA-1 2000 Section 203; IEEE Standard 1063, *IEEE Standard for Software User Documentation*
4.8	10	Acceptance Test	ASME NQA-1 2000 Section 404; IEEE Std. 829, *IEEE Standard for Software Test Documentation;* IEEE Standard 1008, *Software Unit Testing*
4.9	12	Configuration Control	ASME NQA-1 2000 Section 405; ASME NQA-1 2000 Section 406
4.10	13	Error Notification	ASME NQA-1 2000 Section 203

4.1 Topical Area 1 Assessment: Software Classification

This area corresponds to the requirement entitled Software Classification in Table 3-3 of DOE (2003e).

4.1.1 Criterion Specification and Result

Table 4.1-1 lists the subset of criteria reviewed for this topical area and summarizes the findings. Sufficient documentation is provided with software transmittal from the RSICC software center (see Table 1-2, under "Documentation Supplied with Code Transmittal"), to make an informed determination of the classification of the software. A user of the MACCS2 software for safety analysis applications would be expected to interpret the information on the software in light of the requirements for dispersion and dose analysis discussed in Appendix A to DOE-STD-3009-94 to decide on an appropriate safety classification.

For most organizations, the safety class or safety significant classification, Level B (or the equivalent) in the classification hierarchy discussed in DOE (2003e), would be selected. For example, based on the software requirements procedure used by SNL for the Waste Isolation Pilot Plant, the MACCS2 software would be deemed Compliance Decision (CD) software SNL (2003).

Table 4.1-1 --- Subset of Criteria for Software Classification Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
1.1	The code developer must provide sufficient information to allow the user to make an informed decision on the classification of the software.	Yes	Sufficient information is provided from RSICC and documentation from the software developer. Interpreted in light of Appendix A to DOE-STD-3009-94, MACCS2 is used by most sites to assist in making control set decisions. However, it is never used to take action alone, i.e., without consideration of other data or supporting information.

4.1.2 Sources and Method of Review

Documentation supplied with the MACCS2 software package was used along with previously obtained MACCS2 documents as the basis for response to this requirement.

4.1.3 Software Quality-Related Issues or Concerns

There are no SQA issues or concerns relative to this requirement.

4.1.4 Recommendations

This requirement is met. No recommendations are required at this time to improve compliance with the requirement.

4.2 Topical Area 2 Assessment: SQA Procedures and Plans

This area corresponds to the requirement entitled *SQA Procedures and Plans* in Table 3-3 of DOE (2003e).

To assess this area, extensive use is made of an earlier independent review of the MACCS2 SQA Program (East 1998b). The documented review was preceded by an in-depth review at Sandia National Laboratories in 1997. The following, based on the 1997 review, provides a synopsis of the SQA program, prior to and during the period that MACCS2 was in development.

> SNL established a SQA program for Laboratory software in the late 1980s and early 1990s that was compliant with the IEEE Standard for Software Quality Assurance Plans.

The final volume, Volume 2, was put into place in 1995. The guidelines[6] are documented as listed:

Volume 1 – Software Quality Planning [SNL, 1987]
Volume 2 – Documentation [SNL, 1995]
Volume 3 – Standards, Practices, and Conventions [SNL, 1986]
Volume 4 – Configuration Management [SNL, 1992]; and
Volume 5 –Tools, Techniques, and Methodologies [SNL, 1989].

The following is a list and description of the necessary documents required for a complete SNL SQA package [SNL, 1986]:

Project Plan: The project plan is a brief overview of the project. It defines the project, describes the organization, proposes schedules and milestones, and defines procedures to ensure the quality of the final product.

Software Requirements Specification (SRSp): The SRSp is a description of the external interfaces and essential requirements of the software in terms of functions, performance, constraints, and attributes. Requirements are objective and measurable. The SRSp is concerned with what is required, not how to achieve it. This document is reviewed by project members, users, and management. They verify that the intent of the SRSp is clear, the software proposed by the SRSp is what is desired, and that the project can proceed to the next development stage.

Design Description: A Design Description documents the design work accomplished during the design phase. Documenting the design prior to coding avoids (or reduces) any design misunderstandings and subsequent re-coding.

Design Review Results: The results of the Design Review are documented in a report, which identifies all deficiencies discovered during the review along with a plan and schedule for corrective actions. The updated design description document, when placed under configuration control, will establish the baseline for subsequent phases of the software life cycle.

Structured Source Code: Implementation is the translation of the detailed design into a computer language; a process commonly called *coding*.

Test Set: The Test Set includes "rich" test data and relevant test procedures and tools to adequately test the application's response to valid as well as invalid data.

Test Set Documentation: The Test Set Documentation (or Software Test Plan) describes the test data, procedures, tools, and overall plan.

Test Results: The results of the tests should be documented to identify all deficiencies discovered.

Maintenance Documentation: Well-documented code and the software design document provide the backbone of maintenance documentation and the starting point for determining training needs.

Training Plan: The preparation of a well thought out training plan is an essential part of bringing a system into smooth operation. If the people, documents, and training techniques are not considered in the early planning for a new system, resources may not be available and training will be haphazard.

---

6 - The SNL documentation is clearly described as guidance. The management directing a given software project may choose all, some, or none, of the recommendations outlined in the guidelines.

User's Manual or Operating Procedures: A user's manual is organized to contain practical information for individuals required to execute the software. Depending on the size and type of system, operating procedures may be required as a separate document to cover management of the logical and physical components. Without a properly prepared user's guide or operator instructions, either the time of the user will be wasted determining what to do, or the system will be inappropriately used, or both.

Configuration Management Plan: The Configuration Management Plan lists all modules used by the project, module locations, personnel responsible for controlling changes, and change procedures.

Baseline Table: The Baseline Table lists modules and versions in the project's baselined system.

Change Table: The Change Table lists all changes and enhancements made to the modules. Additional update supporting documents reflect changes and enhancements made to the system.

Of the five SNL software guideline volumes, two[7] were published after the completion of the original MACCS code. The other three[8] were published during the development phase of the MACCS code, but preceded MACCS2 development.

Although the guidelines were published after the completion of the MACCS code, the MACCS development followed a systematic method in its planning and execution, and in the error reporting and correction phase. In the initial code development for MACCS2, the same systematic method was followed. It is noted that while draft project, development and test plans were developed and partially implemented at some stages of the software development, formal approval and implementation were not achieved. A draft test plan was followed through MACCS2 Version 1.02, and then apparently abandoned. In summary, the set of SQA plans were never finalized and subsequently, a formal SQA plan was not put into place.

In this criterion and others that are deficient, the SQA evaluators noted that the MACCS predecessor software development tended to be more compliant relative to SNL guidelines. In contrast, the MACCS2 development was not as intentional in applying and maintaining the SNL guidance. Frequently, the MACCS2 development program was characterized by insufficient review prior to releases of the code that would have precluded software inconsistencies. It is believed that an insufficient level of staffing, especially in the latter portion of the MACCS2 development effort, was the chief cause.

---

7 - The two volumes published after the start of the MACCS2 development activity were Documentation and Configuration Management volumes. The Documentation volume [SNL, 1995] presents a description of documents needed for developing, maintaining, and defining software projects. The Configuration Management volume [SNL, 1992] presents a discussion of configuration management objectives and approaches throughout the software life-cycle for software projects at SNL.

8 - The three volumes published before the start of the MACCS2 development were Software Quality Planning, Standards, Practices, and Conventions, and Tools, Techniques, and Methodologies volumes. The Software Quality Planning volume [SNL, 1987] presents an overview of procedures designed to ensure software quality. The Standards, Practices, and Conventions volume [SNL, 1986] presents standards and practices for developing and maintaining quality software at SNL and includes a description of the documents needed for a complete SQA package at SNL. The Tools, Techniques, and Methodologies volume [SNL, 1989] presents evaluations and a directory of software tools and methodologies available to SNL personnel.

Monthly reports to DOE from SNL, and to SNL management from a MACCS2 subcontractor indicated that testing was being performed during the development of the code. However, copies of the testing reports have not been maintained.

In addition to the testing, SNL contracted the University of New Mexico (UNM) to independently test MACCS2 during development. This testing was published in a draft document [Summa, 1996], but never finalized. The report focused on the following areas:

ATMOS Module: Calculation of the downwind relative air concentration (χ/Q) and of the diffusion parameters by using both the power law and the new look-up table methods

EARLY Module: Calculation of the acute thyroid dose, of the network evacuation centerline dose, of the radial evacuation peak dose, of the crosswind evacuation dose, and the dose when the evacuation speed changes

CHRONC Module: Testing of the ability to turn off the long-term phase and the decontamination model, comparison of intermediate phase and long-term phase doses, and calculation of the intermediate phase dose.

The testing by UMN was done in an iterative manner. Errors discovered by UNM resulted in coding changes and a new version of the code. The new code version would then be retested by UNM for the function in question. This process would continue until the function worked correctly. Once a program section was accepted as valid, testing of the next area of the model began. Upon completion of the testing effort, the test cases were not rerun to ensure that subsequent modifications had no effect on the validity of previously tested functionality. The code testing was a "work in progress" effort, and the final released version of MACCS2 had differences from the successive code versions tested by UNM. The UNM testing also did not include any of the preprocessors developed by SNL nor did it include the COMIDA (food pathways) module. These factors combined with lack of resources led to the Summa (1996) report not being published.

4.2.1 Criterion Specification and Result

Table 4.2-1 lists the subset of criteria reviewed for this topical area and summarizes the findings. Because SQA plan and procedures from the software developer were not available, a thorough evaluation was not possible. Based on discussions with previous MACCS2 project leads, the SQA Program reviewer from 1997-1998 (J. East), and East (1998b), it is believed that many elements of a compliant SQA plan and procedures were followed informally, at least in part. However, definitive confirmation through written, approved documentation is not available.

Table 4.2-1 --- Subset of Criteria for SQA Procedures and Plans Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
2.1	Verify that procedures/plans for SQA (**SQA Plan**) have identified organizations responsible for performing work; independent reviews, etc.	Partial. Documentation was not maintained.	Parts of SNL plan were followed early in the MACCS2 project. Later in the project, schedule and resource constraints led to SQA guidelines not being followed.
2.2	Verify that procedures/plans for SQA (**SQA Plan**) have identified software engineering methods.	No.	Parts of SNL plan were followed early in the MACCS2 project. Later in the project, schedule and resource constraints led to SQA guidelines not being followed.
2.3	Verify that procedures/plans for SQA (**SQA Plan**) have identified documentation to be required as part of program.	Partial.	Parts of SNL plan were followed early in the MACCS2 project. Later in the project, schedule and resource constraints led to SQA guidelines not being followed.
2.4	Verify that procedures/plans for SQA (**SQA Plan**) have identified standards, conventions, techniques, and/or methodologies that shall be used to guide the software development, methods to ensure compliance with the same.	No.	Parts of SNL plan were followed early in the MACCS2 project.
2.5	Verify that procedures/plans for SQA (**SQA Plan**) have identified software reviews and schedule.	Partial.	Parts of SNL plan were followed early in the MACCS2 project. Later in the project, schedule and resource constraints led to SQA guidelines not being followed.
2.6	Verify that procedures/plans for SQA (**SQA Plan**) have identified methods for error reporting and corrective actions.	Partial.	Parts of SNL plan were followed early in the MACCS2 project. Later in the project, schedule and resource constraints led to SQA guidelines not being followed.

4.2.2 Sources and Method of Review

This review was based on Chanin (1997), East (1998b) and Summa (1996), and several emails documented as appendices to East (1998b). It also includes discussions with SNL code developers and David Chanin. Both discussions occurred in January 2004.

4.2.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plan and procedures for MACCS2 should be addressed for Version 1.12 of MACCS2. A preferred course of action is that future versions of MACCS2 be shown to be compliant with SNL SQA Plan and Procedures, or their equivalent.

4.2.4 Recommendations

The criteria are not or partially met. Thus the requirement is not met. Recommendations related to this topical area are provided as follows:

- Incorporate the draft report by Summa (1996) on *Pre-Release Verification Testing of the MACCS2* Code into a document that describes the evolution of MACCS into MACCS2 and also describes the internal SQA procedures followed during that development effort, which spanned 1992 to 1996, and involved contributions from a beta-test group that spanned the DOE Complex.
- Document brief SQA plan for Version 1.12 of MACCS2. Revise as needed for future updates such as Version 1.13 for public distribution, including both SQA plan and procedures.

4.3 Topical Area 3 Assessment: Requirements Phase

This area corresponds to the requirement entitled Requirements Phase in Table 3-3 of DOE (2003e).

While requirements for MACCS2 may have been informally agreed upon, and a draft requirements file maintained, no finalized written documentation exists. Anecdotal evidence suggests that there was an understanding of the requirements that MACCS would meet, but this understanding was not documented.

4.3.1 Criterion Specification and Results

Table 4.3-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.3-1 — Subset of Criteria for Requirements Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
3.1	Software requirements for the subject software have been established.	No.	A series of requirements documents were apparently drafted but no longer exist. A full peer-reviewed report would likely need to be written, especially to capture the changes intended in going from MACCS 1.5.11.1 to MACCS2 1.12.
3.2	Software requirements are specified, documented, reviewed and approved.	No.	A series of requirements documents were apparently drafted but no longer exist. Consequently, a full, comprehensive report does not exist to check whether software requirements are specified, documented, reviewed and approved.
3.3	Requirements define the functions to	No.	A series of requirements documents were apparently drafted but no longer exist.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	be performed by the software and provide detail and information necessary to design the software.		Consequently, a full, comprehensive report does not exist to check whether the software requirements define the functions to be performed by the software and provide detail and information necessary to design the software.
3.4	A **Software Requirements Document**, or equivalent defines requirements for functionality, performance, design inputs, design constraints, installation considerations, operating systems (if applicable), and external interfaces necessary to design the software.	No.	A series of requirement-related files were apparently drafted but were not maintained. Consequently, a full, comprehensive report does not exist to check whether requirements are defined for functionality, performance, design inputs, design constraints, installation considerations, operating systems (if applicable), and external interfaces necessary to design the software.
3.5	Acceptance criteria are established in the software requirements documentation for each of the identified requirements.	No.	Memos and e-mails were written on acceptance criteria. However, these were not maintained. Consequently, a full, comprehensive report does not exist to check whether acceptance criteria are established in the software requirements documentation for each of the identified requirements.

4.3.2 Sources and Method of Review

This review was based on information contained in East (1998b) and Bixler (2000). It also includes discussions with SNL code developers and David Chanin. Both discussions occurred in January 2004.

4.3.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written Requirements Document for MACCS2 should be addressed as part of the written SQA Plan and Procedures for this software.

4.3.4 Recommendations

The five criteria are not met. Thus the requirement is not met. It is recommended to develop a Requirements Document for MACCS2 that is consistent with the draft information developed early in the MACCS2 project but never completed in a formal report. It should reflect NRC-specified needs for the software as well as those required by DOE and other organizations that sponsored revisions to the software. Especially important are descriptions of the software changes that were made in going from MACCS to MACCS2, including the architectural changes. It would be useful to capture this information in Version 1.13 documentation.

4.4 Topical Area 4 Assessment: Design Phase

This area corresponds to the requirement entitled Design Phase in Table 3-3 of DOE (2003e).

A Software Design Document was not written for MACCS2. To permit a limited evaluation, an alternative process, that of reviewing model description sections in four reports was applied (Chanin, 1990 Volumes 1 and 2; Chanin, 1992a, and Chanin, 1998). The key assumption was made that documentation describing earlier MACCS development could be extended *partially* to MACCS2.

4.4.1 Criterion Specification and Result

Table 4.4-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results

Criterion Number	Criterion Specification	Compliant?	Summary Remarks
4.1	A software design was developed, documented, reviewed and controlled.	Partial.	Elements of this criterion may be inferred from documentation, a formal design document was not prepared.
4.2	Code developer 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.	Partial.	Design may be inferred from the final software product, but requirements documentation was not maintained.
4.3	The following design elements should be present and documented: 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).	Partial.	Inferred in part from MACCS and MACCS2 documentation. However, the documentation for MACCS in many areas does not reflect what was implemented in MACCS2.
4.4	The following design elements should be present and documented: that 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.	Partial.	Documentation and discussions imply that operating environment was considered.
4.5	The following design elements should be present and documented: evidence of measures to mitigate the consequences of software design problems. These potential problems include external and internal abnormal conditions and events that can affect the computer program.	Not applicable to non-process, instrumentation and control software.	-
4.6	A Software Design Document, or	Partial.	Evidence of the design intent

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	equivalent, is available and contains a description of the major components of the software design as they relate to the software requirements.		relating back to requirements may be inferred from MACCS and MACCS2 documents.
4.7	A Software Design Document, or equivalent, is available and contains 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.	Partial.	Most of the listed elements are addressed in documentation listed in Section 4.4.2. Note that the basic code features preserved in MACCS2 (from MACCS) as well as the code changes introduced in MACCS2 are described in the MACCS2 User's Guide (Chanin, 1998).
4.8	A Software Design Document, or equivalent, is available and contains a description of the allowable or prescribed ranges for inputs and outputs.	Partial.	Allowed ranges of input parameters are covered in Chanin (1998). User knowledge and accident analysis background is required to understand if outputs are logical.
4.9	A Software Design Document, or equivalent, is available and contains the design described in a manner that can be translated into code.	No.	The required detail to establish compliance is not available.
4.10	A Software Design Document, or equivalent, is available and contains 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.	Uncertain.	It is uncertain whether the software developer has maintained this information during development of MACCS2. User implementation of MACCS2 is guided by a loading and execution sequence provided as part of a "read me" file.
4.11	The organization responsible for the design identified and documented the particular verification methods to be used and assured that an Independent Review was performed and documented. This review evaluated the technical adequacy of the design approach; assured internal completeness, consistency, clarity, and correctness of the software design; and verified that the software design is traceable to the requirements.	Partial.	Some measure of verification provided in the draft report by the University of New Mexico (Summa, 1996).
4.12	The organization responsible for the design assured that the test results adequately demonstrated the requirements were met.	Uncertain.	Not possible to verify this area because no documentation is available.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.13	The Independent Review was performed by competent individual(s) other than those who developed and documented the original design, but who may have been from the same organization.	Partial: Yes (1992 – 1995); No (1995 – 1997)	Early MACCS2 project had adequate independence. Second period of effort lacked independence.
4.14	The results of the Independent Review are documented with the identification of the verifier indicated.	Partial.	A portion of the independent review was documented but never published.
4.15	If review alone was not adequate to determine if requirements are met, alternate calculations were used, or tests were developed and integrated into the appropriate activities of the software development cycle.	Uncertain.	Discussions with code staff suggest that this technique was used in the early (1992-1995) program, but lapsed in the later period (1995 – 1997).
4.16	Software design documentation was completed prior to finalizing the Independent Review.	Uncertain.	Not likely, but cannot be confirmed.
4.17	The extent of the Independent Review and the methods chosen are shown to be a function of: the importance to safety, the complexity of the software, the degree of standardization, and the similarity with previously proven software.	Uncertain.	Insufficient information is available to demonstrate that a graded application was followed.

4.4.2 Sources and Method of Review

Design requirements information was evaluated through discussions with David Chanin and the current MACCS2 staff, and review of the reports listed in Table 1-3. In particular, the following were used:

- Chanin, 1990, D.I. Chanin, J.L. Sprung, L.T. Ritchie, H-N Jow, and J.A. Rollstin, *MELCOR Accident Consequence Code System (MACCS). Volume 1: User's Guide*; H-N Jow, J.L. Sprung, J.A. Rollstin, L.T. Ritchie, and D.I. Chanin, *Volume 2: Model Description*; J.A. Rollstin, D.I. Chanin, and H-N Jow, *Volume 3: Programmer's Reference Manual*; NUREG/CR-4691, Sandia National Laboratories, published by the U.S. Nuclear Regulatory Commission, Washington, DC, 1990.

- Chanin, 1992a, D. Chanin, J. Rollstin, J. Foster, and L. Miller, *MACCS Version 1.5.11.1: A Maintenance Release of the Code*, Sandia National Laboratories, Albuquerque, NM, July 14, 1992.

- Dobbe 1990, C.A. Dobbe, E.R. Carlson, N.H. Marshall, E.S. Marwil, J.E. Tolli. *Quality Assurance and Verification of the MACCS Code, Version 1.5,* Idaho National Engineering Laboratory, Idaho Falls, ID, NUREG/CR-5376 (EGG-2566)

- Summa, F.J., (Draft-1996) and F.E. Haskin. *Pre-Release Verification Testing of the MACCS2 Code*. University of New Mexico, Albuquerque, NM. Unpublished report prepared for Sandia National Laboratories.

- Chanin, D., (1997). Software Quality Assurance Procedures Followed with MACCS2, Letter to K. O'Kula (September 1997).

4.4.3 Software Quality-Related Issues or Concerns

A comprehensive Software Design Document for MACCS2 should have been part of the written SQA Plan and Procedures for this software. A SQA assessment must rely on inferences made from the MACCS2 User's Manual (Chanin, 1998).

4.4.4 Recommendations

Of the seventeen criteria evaluated for this requirement, two (2) are not met, nine (9) are partially met, five (5) are uncertain, and one (1) is not applicable. Thus the requirement is not met. Documenting the software design implemented in MACCS2 Version 1.12 is not required at this time. Upgrades to the Model Description and other documentation can meet the intent of the Software Design Document, but only if the discussion is comprehensive and detailed and provides insights in design differences in transitioning from MACCS to MACCS2. In the long-term, software design information is recommended for the version of MACCS2 ultimately maintained in the toolbox.

4.5 Topical Area 5 Assessment: Implementation Phase

This area corresponds to the requirement entitled Implementation Phase in Table 3-3 of DOE (2003e).

4.5.1 Criterion Specification and Result

Table 4.5-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
5.1	The implementation process resulted in software products such as computer program listings and instructions for computer program use.	Yes.	User guide, model description, and code listing from RSICC confirm that the code developer met this criterion.
5.2	Implemented software was analyzed to identify and correct errors.	Uncertain.	Not possible to verify due to lack of written documentation. Discussions with code consultant suggest criterion was likely to have been met.
5.3	The source code finalized during verification (this phase) was placed under configuration control.	Uncertain.	Not possible to verify due to lack of written documentation. Discussions with code consultant suggest criterion was likely to have been met.
5.4	Documentation during verification included a copy of the software, test case description and associated criteria that are traceable to the software requirements and design documentation.	Partial.	Copy of software and test case description are available. However, not possible to trace to requirements and design documents.

4.5.2 Sources and Method of Review

Documentation listed in Table 1-3 was reviewed to complete review of this criterion. The MACCS2 developer notes that an Implementation Document does not currently exist for MACCS2.

The code listing is available from RSICC upon transmittal of MACCS2 to user groups.

4.5.3 Software Quality-Related Issues or Concerns

Not all criteria can be confirmed due to the lack of written records on implementation. However, based on discussions with project lead for MACCS2 and the subcontractor whom supported the project, it is inferred that most of these requirements were met, at least partially.

4.5.4 Recommendations

Of the four criteria evaluated for this requirement, one (1) is met, two (2) are uncertain, and one (1) is partially met. Thus the requirement is not met. However, it is not recommended that improvements be made related to this topical area for Version 1.12 of MACCS2. Instead, indication of the Implementation processes should be maintained on file for newer versions of the software.

4.6 Topical Area 6 Assessment: Testing Phase

This area corresponds to the requirement entitled Testing Phase in Table 3-3 of DOE (2003e). A Software Test Report has not been provided by the MACCS2 software developers. Instead, a limited evaluation is performed based on review of Chanin (1997), East (1998a), East (1998b), and the related documents listed in Table 1-3 as a basis to address the criteria in Table 4.6-1.

4.6.1 Criterion Specification and Result

Table 4.6-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
6.1	The software was validated by executing test cases.	Yes.	Documentation supports the satisfaction of this criterion.
6.2	Testing demonstrated the capability of the software to produce valid results for test cases encompassing the range of permitted usage defined by the program documentation. Such activities ensured that the software adequately and correctly performed all intended functions.	Uncertain.	Not able to confirm this criterion.
6.3	Testing demonstrated that the computer program properly handles abnormal conditions and events as well as credible failures	Uncertain.	No detailed record is available on outcome of testing for abnormal conditions and credible failures.
6.4	Testing demonstrated that the computer program does not perform adverse unintended functions.	Uncertain.	No detailed record is available on outcome of testing for adverse unintended functions.
6.5	Test Phase activities were performed to assure adherence to requirements, and to assure that the software produces correct results for the test case specified. Acceptable methods for evaluating adequacy of software test case results include: (1) analysis without computer assistance, e.g. hand or spreadsheet calculations; (2) other validated computer programs; (3) experiments and tests; (4) standard problems with known solutions; (5) confirmed published data and	Uncertain.	Test report(s) are not available so the extent of the in-house test program at SNL is not known. No evidence exists of comparisons of MACCS2 to other independent test results. Current suite of test cases supplied with software for user purposes include commercial reactor and DOE nuclear facility examples.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	correlations.		
6.6	Test Phase documentation includes test procedures or plans and the results of the execution of test cases. The test results documentation demonstrates successful completion of all test cases or the resolution of unsuccessful test cases and provides direct traceability between the test results and specified software requirements.	Partial.	No detailed record of testing is available. It is known that testing was conducted on MACCS2, and it is judged that the final version (1.12) performs as intended. However, resolution of unsuccessful cases is not possible to check, nor is traceability between test results and software requirements.
6.7	Test procedures or plans specify the following, as applicable: (1) required tests and test sequence, (2) required range of input parameters, (3) identification of the stages at which testing is required, (4) requirements for testing logic branches, (5) requirements for hardware integration, (6) anticipated output values, (7) acceptance criteria, (8) reports, records, standard formatting, and conventions, (9) identification of operating environment, support software, software tools or system software, hardware operating system(s) and/or limitations.	Partial, or uncertain.	No detailed record of test procedures and plans was available. It is believed that this criterion was partially met with respect to: (1), (2), (3), (6), and (9). Complete verification is not possible based on lack of documentation from developer.

4.6.2 Sources and Method of Review

Discussions with the current MACCS2 staff and the MACCS2 consultant were conducted in January 2004 to supplement information gleaned from the documentation listed in Table 1-3 to complete review of this criterion.

4.6.3 Software Quality-Related Issues or Concerns

Lack of a test report for MACCS2 forces the review to infer test case program results and outcome based on limited information. As was noted previously, the initial period (1992 – 1994) of MACCS2 development had satisfactory procedures and independence during testing. Later testing (1995 – 1997) was not as robust, but did feature an appropriate level of independence in work by the University of New Mexico as an independent checker of changes by SNL (Summa, 1996). It is not possible to verify how complete the University program was, relative to the full software source code package. Apparently, most but not all changes were checked during this phase of the MACCS2 program. It is noted by East (1998a) that of the enhancements and new output types provided in MACCS2, all but the sector independent and sector dependent peak doses for any organ were tested and verified in the Summa report. Both features were found implemented correctly in MACCS2 (East, 1998a).

Other testing of the MACCS2 software is encouraged in terms of comparing test output with other, independent results, as listed in Criterion 6.5. (See Recommendations below, Section 4.6.5).

4.6.4 Recommendations

Of the seven criteria evaluated for this requirement, one (1) is met, four (4) are uncertain, and two (2) are partially met. Thus the requirement is not met. A verifiable, written Test Report Document for MACCS2 should have been part of the written SQA Plan and Procedures for Version 1.12 of this software. Upgrades to the MACCS2 new software baseline will require that a Test Case Description and Report be completed, particularly if Version 1.13 is designated for the toolbox instead of Version 1.12. In addition, review and approval of the test report (Summa, 1996) prepared by the University of New Mexico is strongly recommended

In terms of user implementation, test cases should include example types that serve to demonstrate adequacy of MACCS2 software for specific source term types typically encountered for DOE Documented Safety Analysis. It is recommended that a standard set of problem types include deflagration/detonation and fire-related source terms. Observed results and data from experiments, field tests, or specific "known" dispersion results could be compared to test runs made with the MACCS2 software made by the code developer and stored on the MACCS2 CD sent to users.

4.7 Topical Area 7 Assessment: User Instructions

This area corresponds to the requirement entitled User Instructions in Table 3-3 of DOE (2003e).

User instructions for MACCS2 and its preprocessor programs have been documented (Chanin, 1997; Chanin, 1998). Considered along with DOE-specific input preparation guidance in DOE (2003e), and the older MACCS model (Chanin, 1990; Chanin, 1992a), there is sufficient information to evaluate compliance to this requirement.

4.7.1 Criterion Specification and Result

Table 4.7-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
7.1	A description of the model is documented.	Yes	MACCS and MACCS2 models are described sufficiently.
7.2	User's manual or guide includes approved operating systems (for cases where source code is provided, applicable compilers should be noted).	Yes	RSICC software center distribution notes are available.
7.3	User's manual or guide includes description of the user's interaction with the software.	Yes.	User's Manual discusses this aspect to a limited extent.
7.4	User's manual or guide includes a description of any required training necessary to use the software.	No.	Training requirements are not discussed in MACCS2 documentation.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
7.5	User's manual or guide includes input and output specifications.	Yes.	Well documented input specifications. Output discussion should be expanded.
7.6	User's manual or guide includes a description of software and hardware limitations.	Partial.	Some areas in terms of software/hardware limitations are discussed. This aspect of the MACCS2 document should be clearer in terms of regimes of applicability.
7.7	User's manual or guide includes a description of user messages initiated as a result of improper input and how the user can respond.	Partial.	The user has limited diagnostic assistance to correct errors. MACCS2 documentation does not address error messages satisfactorily.
7.8	User's manual or guide includes information for obtaining user and maintenance support.	Partial.	RSICC-distributed software packages contain email and phone contact information. User interaction with code developer at SNL is limited.

4.7.2 Sources and Method of Review

Compliance with this requirement was evaluated by review of documentation listed in Table 1.3, and from discussions with MACCS2 consultant (Chanin) and with current MACCS2 SNL staff.

4.7.3 Software Quality-Related Issues or Concerns

User instruction documentation is, on the whole, good. No substantive issues or concerns have surfaced.

4.7.4 Recommendations

Of the eight criteria evaluated for this requirement, four (4) are met, one (1) is not met, and three (3) are partially met. The one negative criterion is more than compensated for by the four other positive criteria, and coupled with sufficient written information on which to base the evaluation of this criterion, the requirement is evaluated as met.

Recommendations related to this topical area are as follows:

- User diagnostic assistance during software execution is limited and should be expanded. The User's Guide content is too brief on user-induced software problems. Common errors and warning messages could be included with suggested solutions.
- A list of prerequisites for perspective MACCS2 users should be listed in a MACCS2 website or in the code documentation.
- A simple training set of recommendations would be useful. The input files to start with and the output files that one should see upon successful execution could be placed on a MACCS2 website. The novice user could be tasked with two to three simple problem types to set up. The

associated output files would allow a self-check on the code runs. The current sample case file could take on this function if prioritized correctly.
- Help and internet/email technical contact information for SNL should be provided.
- MACCS2 limitations should be made more explicit in the User's Guide.
- Specific guidance should be provided in selecting various options for dose conversion factors.

4.8 Topical Area 8 Assessment: Acceptance Test

This area corresponds to the requirement entitled Acceptance Test Table 3-3 of DOE (2003e). During this phase of the software development, the software becomes part of a system incorporating applicable software components, hardware, and data, and then is accepted for use.

During development of the software, the developing organization is responsible for documenting its procedures and acceptance tests it uses. Once the software is released, user organizations need a test protocol to determine if the software is correctly installed. Implementation for this type of acceptance testing is the responsibility of the user organization.

The criteria below were applied to the developing organization's processes.

4.8.1 Criterion Specification and Result

Table 4.8-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
8.1	To the extent applicable to the developer, acceptance testing includes a comprehensive test in the operating environment(s).	Uncertain.	No documentation was received describing the acceptance testing of MACCS2 development.
8.2	To the extent applicable to the developer, acceptance testing was performed prior to approval of the computer program for use.	Uncertain.	No documentation was received describing the acceptance testing of MACCS2 development.
8.3	To the extent applicable to the developer, software validation was performed to ensure that the installed software product satisfies the specified software requirements. The engineering function (i.e., the engineering operation an item is required to perform to meet the component or system design basis) determines the acceptance testing to be performed prior to approval of the computer program for use.	Uncertain.	No documentation was received describing the acceptance testing of MACCS2 development.
8.4	Acceptance testing documentation includes results of the execution of test	Partial	The MACCS2 software package from RSICC includes

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	cases for system installation and integration, user instructions (Refer to Requirement 7 above), and documentation of the acceptance of the software for operational use.		a series of test case inputs/outputs. These cases can be viewed as providing users and user groups with a mechanism for deciding if the MACCS2 software is correctly installed and functioning properly.

4.8.2 Sources and Method of Review

Software package for code transmittal and documentation listed in Table 1.3 were reviewed.
An Acceptance Test protocol was not provided to the gap analysis. There is no known formal procedure to assure that an installed version of MACCS2 is working properly. An Installation and Checkout procedure does not exist for MACCS2 (Bixler, 2000).

4.8.3 Software Quality-Related Issues or Concerns

Acceptance Test information should have been provided by the software developers to assure users that the distributed version of MACCS2 met its testing criteria.

4.8.4 Recommendations

Of the four criteria evaluated for this requirement, three (3) are uncertain, and one (1) is partially met. Thus the requirement is not met.

Documentation, either as a separate report or as part of another SQA report, is needed to describe the testing used for Version 1.12 or successor versions of the MACCS2 software. In addition, the developer is recommended to provide an acceptance test protocol to allow users to determine that MACCS2 is properly installed.

Through discussions with the MACCS2 consultant in January 2004, it was learned that some of the requirements for the Operation and Maintenance area were performed in the development of MACCS2. Although evaluation of this area is not included formally in the ten criteria for this gap analysis, the information is included as Appendix B to this report.

4.9 Topical Area 9 Assessment: Configuration Control

This area corresponds to the requirement entitled Configuration Control in Table 3-3 of (DOE 2003e).

No Software Configuration and Control Document was provided by the software developers. The requirement was assessed mostly through discussions with the current code developer and the MACCS2 consultant.

4.9.1 Criterion Specification and Result

Table 4.9-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
9.1	For the developers the methods used to control, uniquely identify, describe, and document the configuration of each version or update of a computer program (for example, source, object, back-up files) and its related documentation (for example, software design requirements, instructions for computer program use, test plans, and results) are described in implementing procedures.	Uncertain	MACCS2 is labeled and documented for release as Version 1.12. However, no documentation was provided to provide detail on how configuration control was achieved and maintained during development.
9.2	Implementing procedures meet applicable criteria for configuration identification, change control and configuration status accounting.	No.	No implementing procedures were identified.

4.9.2 Sources and Method of Review

Discussions with previous SNL staff and with the consultant at the time changes were made to MACCS2 have provided limited information on which to evaluate this requirement. It has been indicated that a Configuration Control system was in place during development of MACCS2 (Bixler, 2000). However, no written description of this system and the methods employed to assure configuration control were made available.

4.9.3 Software Quality-Related Issues or Concerns

Lack of a Software Configuration and Control document for MACCS2 forced the review to infer compliance based on limited information from discussions. This would imply lack of good protocol with regard to configuration control practices.

Additionally, discussions with MACCS2 users in the DOE Complex have indicated that several versions may be in existence. For example, a version of MACCS2 at Pantex, MAX2_MIIC, was developed by SNL to allow use of a different fire plume model and allows the user to input five years of meteorology in

a given run. A NRC-sponsored version, WinMACCS, is being developed for NRC-specific applications. While not violating software practices, multiple versions could cause confusion to users on the MACCS2 version recommended for DSA applications.

4.9.4 Recommendations

Of the two criteria evaluated for this requirement, one (1) is uncertain, and one (1) is not met. Thus the requirement is not met. It is recommended that a full-scope Software Configuration and Control document be issued as part of the new software baseline for the version of MACCS2 that is designated for the DOE software toolbox. Variants of MACCS2 that are used in the DOE Complex should be identified to the full user community with the distinction between the base software and the variants made clear.

4.10 Topical Area 10 Assessment: Error Impact

This area corresponds to the requirement entitled Error Impact in Table 3-3 of DOE (2003e).

An Error Notification and Corrective Action document was not transmitted by the SNL software developers, but an SNL protocol for dealing with software issues has been followed during and after release of Version 1.12 of MACCS2. Thus, the evaluation of compliance with this criterion is limited and is based on interpretation of the documents listed in Table 1.3 and from discussions with MACCS2 code staff and the MACCS2 consultant associated with SNL at the time of MACCS2 development.

4.10.1 Criterion Specification and Result

Table 4.10-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
10.1	The problem reporting and corrective action process used by the software developing organization addresses the appropriate requirements of the developing organization's corrective action system, and are documented in implementing procedures.	Partial.	The process used for monitoring errors and user feedback on MACCS2 was inferred from past practice at SNL but limited documentation was made available to the SQA evaluation.
10.2	Method(s) for documenting (Error Notification and Corrective Action Report), evaluating, and correcting software problems describe the evaluation process for determining whether a reported problem is an error.	Yes.	The method(s) used for evaluating and correcting MACCS2 problems were not provided in a written document. The evaluation process followed was inferred.
10.3	Method(s) for documenting (Error Notification and Corrective Action Report), evaluating, and correcting software problems define the responsibilities for disposition of the problem reports, including notification to the originator of the results of the evaluation.	Partial.	An overall error notification and corrective action process document was not available. Inference drawn from a SNL Software Defect Notice was used to determine that some organizational process must

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			have been followed.
10.4	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the error relates to appropriate software engineering elements.	Yes.	A written procedure on the action to address how the potential software error relates to software engineering elements was not available. Inference drawn from two SNL Software Defect Notices was used to determine that some organizational process must have been followed.
10.5	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the error impacts past and present use of the computer program	Partial.	A written procedure on the action to address how the potential software error impacts past and present use of the computer program was not available. Inference drawn from two SNL Software Defect Notices was used to determine that some organizational process must have been followed.
10.6	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the corrective action impacts previous development activities	Yes.	A written procedure on the action to address how the corrective action impacts previous development activities was not available. Inference drawn from two SNL Software Defect Notices was used to determine that some organizational process must have been followed.
10.7	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the users are notified of the identified error, its impact; and how to avoid the error, pending implementation of corrective actions.	Partial.	A written procedure on the action to address handling how users are notified of identified errors, impacts, error avoidance, and pending implementation of corrective actions was not available. Inference drawn from two SNL Software Defect Notices was used to determine that some organizational process must have been followed.

4.10.2 Sources and Method of Review

Limited documentation was available for this review. SNL has reported that a Software Reporting system was implemented for MACCS2 (Bixler, 2000). However, its effectiveness could not be reviewed directly. Instead, two software defect notifications have been used to infer the approach taken for error/defect reporting and dispositioning.

4.10.3 Software Quality-Related Issues or Concerns

While an error/defect notification process is institutionalized at Sandia National Laboratories, it is not clear how it is effectively used. There appears to be limited use of the reporting system at RSICC. While software defects in MACCS2 were known beginning in 1997, it took over a year for software defect notices to be disseminated (Gregory, 1998). Thus, the efficacy of the current system of developer – user notification and communication of planned corrective actions is uncertain.

Known software defects still exist in MACCS2 despite developer awareness and the obvious approach toward correction (Bixler, 2000). The two defects impact results during multiple-plume segment calculations, and in use of the emergency response model. Only the first defect would impact typical calculations supporting Documented Safety Analyses. Nonetheless, both defects should be corrected without additional delay.

4.10.4 Recommendations

Of the seven criteria evaluated for this requirement, three (3) are met, and four (4) are partially met. However, several examples of an Error Notification and Corrective Action process have been confirmed by written document that demonstrate a satisfactory process exists. Therefore, it is judged that the overall requirement is partially met.

As part of the new software baseline for MACCS2, a comprehensive Software Error Notification and Corrective Action process should be provided. Expanded use of the RSICC user network is also suggested to provide more timely reporting of user issues, software news, suggested strategies for resolving software problems, and general communications. In the future, communication of software concerns among MACCS2 users, and the corrective actions to be taken, need to be executed in a timely manner.

Known software defects in MACCS2 should be corrected immediately, and a new maintenance version of the software made available to the user community. Discussions with the staff at SNL in January 2004 indicate that identified MACCS2 software flaws are corrected in Version 1.13. This Version has been released immediately prior to this gap analysis report.

4.11 Training Program Assessment

Current MACCS2 training opportunities are limited and not well publicized. Comprehensive training should be provided on a more frequent basis.

The Energy Facility Contractors Group (EFCOG) Workshops provide two annual opportunities to provide training to the core DOE user group. The winter session is during the Safety Basis Subgroup meeting and the summer session is organized for the full Safety Analysis Working Group. Multi-day MACCS2 training at these two workshops would potentially reach 300 DOE MACCS2 users, managers,

regulators, and oversight groups. Site-specific training is also suggested, similar to training conducted by other designated toolbox software developers.

It is also strongly suggested that training be offered for certification. This level of user proficiency could be measured by demonstrating competency through a written exam and software execution of a set of test cases.

4.12 Software Improvements and New Baseline

Software improvements for MACCS2 for a Nuclear Regulatory Commission (NRC)-sponsored program have been documented by Bixler (2000). The new software, WinMACCS, will focus on developing a graphical user interface to MACCS2, its preprocessors, and the related post-processors. For this modification, a slightly modified version of MACCS2 will become a module of WinMACCS. Modifications to the existing MACCS2 for WinMACCS were described as falling in two categories: (1) correcting all known FORTRAN errors/problems; and (2) supporting the interface between the "front" end and the FORTRAN modules.

MACCS2 Version 1.13 has recently been released. While the new version corrects known software errors identified since the late 1990s, other improvements made in the software are not known at this time.

The NRC-sponsored WinMACCS version, despite user interface improvements, does not address the majority of SQA issues associated with Version 1.12 of MACCS as identified in this report. The minimum remedial program required to yield the new software baseline for MACCS2 was discussed earlier as part of Table 1.1. Included are creation of, or upgrades to, software documents that can then constitute the new baseline for the software, including:

1. Software Quality Assurance Plan
2. Software Model Description, including, but not limited to,
    a. Software Requirements
    b. Software Design
3. User's Manual, including, but not limited to,
    a. User Instructions
    b. Test Case Description and Report
    c. Software Configuration and Control
4. Error Notification and Corrective Action Process.

As is noted in the above list, many of the functions of these documents can be met in the same report. For example, much of the intent of the requirements and design documents can be captured in a comprehensive Model Description report. Furthermore, Document 4 (above) can be a reference to a SNL-wide report as long as the error notification and corrective action process followed for MACCS2 is addressed. The key point of this remedial SQA document list is to provide a comprehensive and cohesive set of documentation that describes the migration path from MACCS to MACCS2, including the architectural changes that were made. In addition, a model description is needed that fully reflects the applicable models in MACCS2, and indicates those MACCS models that are no longer active. Reference to earlier MACCS2- and MACCS-related documents is encouraged, as long as the document in question is still applicable to the version of MACCS2 being described. A comprehensive documentation effort of this level will greatly enhance user understanding of the MACCS2 models.

Despite the priority and attention to the user interface, Bixler (2000) provides a reasonable estimate of the level of effort needed to meet an earlier version of ASME NQA-1. The estimate of the program and level of effort required to upgrade the MACCS2 computer software was prepared based on NP-19 for WIPP applications. NP-19 was identified earlier, and is a SNL procedural guide that implements an earlier version of Subpart 2.7 to NQA-1, specifically NQA-2a-1990. The minimum set of actions described in Bixler (2000) includes:

- Create a Primitive Baseline (PB) document to establish the SQA status of the existing code
- Establish a Verification and Validation Plan (VVP) based on the above
- Create an Implementation Document (ID) to describe the process of generating the executable software modules
- Update, the User's Manual (UM)
- Generate a Validation Document (VD), to measure the performance of the software against the criteria specified in the VVP
- Perform Installation and Checkout (I&C) to verify correct installation on all supported platforms
- Implement a Software Configuration Control System (CC)
- Implement a Software Problem Reporting System (SPR).

While not exactly matching up with the program proposed here, the SNL proposed program is similar to the improvement actions outlined in this report. Furthermore, the estimates are based on Sandia National Laboratory resources, and as such, are taken as more accurate resource estimates than could be provided otherwise. The overall SQA upgrade program in the SNL program is estimated to require 1.5 full-time equivalent years to complete. The requirements are matched against the requirements earlier, in this document (Table 4.12-1). The overall level of effort, 1.5 FTE-years is rounded up to 2 FTE-years as the final estimate for resource allocation to perform the upgrades required to compensate for MACCS2's known SQA gaps. The estimate compares favorably with an independent 2-FTE-year value generated for a SQA plan that follows ANSI/ANS-10.4 (East, 1998b).

Table 4.12-1. — Comparison of NQA-1 Requirements, with SQA Upgrade Steps Discussed in Bixler (2000) with the Approach Discussed in DOE (2003e)

ASME NQA-1-2000 Requirements	Bixler (2000)	DOE (2003e)
	SNL NP 19-1	Level B Existing Software*
Software Classification		4.1
SQA Procedures/Plans		4.2
Dedication		-
Evaluation	PB	-
Requirements	SRD	4.3
Design		4.4
Implementation		4.5
Testing	VVP, VD	4.6
User Instructions	ID, UM	4.7
Acceptance Test	I&C	4.8
Operation and Maintenance		Appendix B
Configuration Control	CC	4.9
Error Impact	SPR	4.10
Access Control		-

*** Section covered in this report.**

5.0 Conclusions

The gap analysis for Version 1.12 of the MACCS2 software, based on a set of requirements and criteria compliant with NQA-1, has been completed. Of the ten primary SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), two requirements are met at an acceptable level, i.e., *Classification* (1) and *User Instructions* (7). A third requirement, *Error Notification and Corrective Action* (10), is partially met. Improvement actions are recommended for MACCS2 to fully meet requirement (10) criteria, and the remaining seven requirements. This evaluation outcome is deemed acceptable 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.

It was determined that MACCS2 code does meet its intended function for use in supporting documented safety analysis. However, as with all safety-related software, users should be aware of current limitations and capabilities of MACCS2 for supporting safety analysis. Informed use of the software can be assisted by the current set of MACCS2 reports (refer to Table 1-3), and the code guidance report for DOE safety analysts, *MACCS2 Computer Code Application Guidance for Documented Safety Analysis*, (DOE, 2004). Furthermore, while SQA improvement actions are recommended for MACCS2, no evidence has been found of programming, logic, or other types of software errors in MACCS2 that have led to non-conservatisms in nuclear facility operations, or in the identification of facility controls.

By order of priority, it is recommended that MACCS2 software improvement actions be taken, especially:

- correcting know defects
- upgrading user technical support activities
- providing training on a regular basis, and
- revising software documentation.

Performing these four primary actions should satisfactorily improve the SQA compliance status of MACCS2 relative to the primary evaluation criteria cited in this report.

A new software baseline set of documents is recommended for MACCS2 to demonstrate completion of the revision to software documentation item (above). The list of baseline documents for revision includes:

1. Software Quality Assurance Plan
2. Software Model Description, including, but not limited to,
    a. Software Requirements
    b. Software Design
3. User's Manual, including, but not limited to
    a. User Instructions
    b. Test Case Description
    c. Software Configuration and Control
4. Error Notification and Corrective Action Process.

Additionally, user documentation should be augmented to include error diagnostic advice and suggested input files for prototypic nuclear facility safety analysis problem types. Approximately two full-time equivalent years (2 FTEs) is estimated to complete these actions. Of this level of effort, 1.5 FTE is

estimated for the current software owner, Sandia National Laboratories, and roughly 0.5 FTE is estimated to be required for independent review.

A new version of MACCS2, Version 1.13, has recently been released. It is recommended that this version be evaluated relative to the software improvement and baseline recommendations, and the complete set of SQA criteria discussed in this report. If this version is found to be satisfactory, it should replace Version 1.12 as the designated version of the software for the DOE Safety Software Toolbox.

Currently, MACCS2 training is occasionally offered on an informal basis, and user requirements are not discussed in the code developer's documentation. It is recommended that user training for DOE safety analysis applications be conducted formally on at minimum, an annual basis. Prerequisites for, and core knowledge needed by, the user prior to initiating MACCS2 applications should be documented by the code developer.

Approximately one FTE-month per year would be needed to maintain a web-based error notification and corrective action process for MACCS2 (Section 4.10). However, such a process has not been defined in depth for MACCS2 and the other designated toolbox codes.

6.0 Acronyms and Definitions

ACRONYMS:

AEC	Atomic Energy Commission
ALOHA	Areal Locations of Hazardous Atmospheres (designated toolbox software)
ANS	American Nuclear Society
ANSI	American National Standards Institute
ASME	American Society of Mechanical Engineers
CCPS	Center for Chemical Process Safety
CD	Compliance Decision
CFAST	Consolidated Fire and Smoke Transport Model (designated toolbox software)
CFR	Code of Federal Regulations
DCF	Dose Conversion Factor
DNFSB	Defense Nuclear Facilities Safety Board
DoD	Department of Defense
DOE	Department of Energy
DSA	Documented Safety Analysis
EFCOG	Energy Facility Contractors Group
EH	DOE Office of Environment, Safety and Health
EIA	Electronic Industries Alliance
EM	DOE Office of Environmental Management
EPIcode	Emergency Prediction Information code (designated toolbox software)
EPRI	Electric Power Research Institute
FTE	Full-time equivalent
GENII	Generalized Environmental Radiation Dosimetry Software System - Hanford Dosimetry System (Generation II) (designated toolbox software)
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IP	Implementation Plan
ISO	International Organization for Standardization
MACCS2	MELCOR Accident Consequence Code System 2 (designated toolbox software)
MELCOR	Methods for Estimation of Leakages and Consequences of Releases (designated toolbox software)
NNSA	National Nuclear Security Administration
NRC	Nuclear Regulatory Commission
OCRWM	Office of Civilian Radioactive Waste Management
PSA	Probabilistic Safety Analysis (or Assessment)
QAP	Quality Assurance Program (alternatively, Plan)
RSICC	Radiation Safety Information Computational Center
SNL	Sandia National Laboratories
SQA	Software Quality Assurance
SRS	Savannah River Site
V&V	Verification and Validation
WSRC	Westinghouse Savannah River Company
YMP	Yucca Mountain Project

DEFINITIONS:

The following definitions are taken from the Implementation Plan. References in brackets following definitions indicate the original source, when not the Implementation Plan.

Acceptance Testing — The process of exercising or evaluating a system or system component by manual or automated means to ensure that it satisfies the specified requirements and to identify differences between expected and actual results in the operating environment. [NQA-1]

Central Registry — An organization designated to be responsible for the storage, control, and long-term maintenance of the Department's safety analysis "toolbox codes." The central registry may also perform this function for other codes if the Department determines that this is appropriate.

Computer Code — A set of instructions that can be interpreted and acted upon by a programmable digital computer (also referred to as a module or a computer program).

Dedication (of Software) — The evaluation of software not developed under utilizing organization existing quality assurance plans and procedures (or not developed under NQA-1 standards). The evaluation determines and asserts the software's compliance with NQA-1 quality standards and its readiness for use in specific applications. (Typically applies to commercially available software.) The utilizing organization reviews the intended software application sufficiently to determine the critical functions that provide evidence of the software's suitability for use. Once the critical functions have been established, methods are defined to verify critical function adequacy and provide verifiable acceptance criteria. Acceptable dedication methods are implemented and required documentation is prepared.

Design Requirements — Description of the methodology, assumptions, functional requirements, and technical requirements for a software system.

Error — A condition deviating from an established base line, including deviations from the current approved computer program and its baseline requirements. [NQA-1]

Executable Code — The user form of a computer code. For programs written in a compilable programming language, the compiled and loaded program. For programs written in an interpretable programming language, the source code.

Firmware — The combination of a hardware device and computer instructions and data that reside as read-only software on that device. [IEEE Standard 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology]

Gap Analysis — Evaluation of the Software Quality Assurance attributes of specific computer software against identified criteria.

Nuclear Facility — A reactor or a nonreactor nuclear facility where an activity is conducted for or on behalf of DOE and includes any related area, structure, facility, or activity to the extent necessary to ensure proper implementation of the requirements established by 10 CFR 830. [10 CFR 830]

Object Code — A computer code in its compiled form. This applies only to programs written in a compilable programming language.

Operating Environment — A collection of software, firmware, and hardware elements that provide for the execution of computer programs. [NQA-1]

Safety Analysis and Design Software – Computer software that is not part of a structure, system, or component (SSC) but is used in the safety classification, design, and analysis of nuclear facilities to: ensure the proper accident analysis of nuclear facilities; ensure the proper analysis and design of safety SSCs; and, ensure the proper identification, maintenance, and operation of safety SSCs. [DOE O 414.1B]

Safety Analysis Software Group (SASG) — A group of technical experts formed by the Deputy Secretary in October 2000 in response to Technical Report 25 issued by the Defense Nuclear Facilities Safety Board (DNFSB). This group was responsible for determining the safety analysis and instrument and control (I&C) software needs to be fixed or replaced, establishing plans and cost estimates for remedial work, providing recommendations for permanent storage of the software and coordinating with the Nuclear Regulatory Commission on code assessment as appropriate.

Safety-Class Structures, Systems, and Components (SC SSCs) — SSCs, including portions of process systems, whose preventive and mitigative function is necessary to limit radioactive hazardous material exposure to the public, as determined from the safety analyses. [10 CFR 830]

Safety-Significant Structures, Systems, and Components (SS SSCs) — SSCs which are not designated as safety-class SSCs, but whose preventive or mitigative function is a major contributor to defense in depth and/or worker safety as determined from safety analyses. [10 CFR 830] As a general rule of thumb, SS SSC designations based on worker safety are limited to those systems, structures, or components whose failure is estimated to result in prompt worker fatalities, serious injuries, or significant radiological or chemical exposure to workers. The term serious injuries, as used in this definition, refers to medical treatment for immediately life-threatening or permanently disabling injuries (e.g., loss of eye, loss of limb). The general rule of thumb cited above is neither an evaluation guideline nor a quantitative criterion. It represents a lower threshold of concern for which an SS SSC designation may be warranted. Estimates of worker consequences for the purpose of SS SSC designation are not intended to require detailed analytical modeling. Consideration should be based on engineering judgment of possible effects and the potential added value of SS SSC designation. [DOE G 420.1-1]

Safety Software — Includes both safety system software, and safety analysis and design software. [DOE O 414.1B]

Safety Structures, Systems, and Components (SSCs) — The set of safety-class SSCs and safety-significant SSCs for a given facility. [10 CFR 830]

Safety System Software — Computer software and firmware that performs a safety system function as part of a structure, system, or component (SSC) that has been functionally classified as Safety Class (SC) or Safety Significant (SS). This also includes computer software such as human-machine interface software, network interface software, programmable logic controller (PLC) programming language software, and safety management databases that

are not part of an SSC but whose operation or malfunction can directly affect SS and SC SSC function. [DOE O 414.1B]

Software — Computer programs, operating systems, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. [IEEE Standard 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology]

Software design requirements The activities that begin with the decision to develop a software product and end when the software is delivered. The software development cycle typically includes the following activities:

Software design
Implementation
Test, and sometimes:
Installation. [NQA-1]

Software Engineering — The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software; also: the study of these applications. [NQA-1]

Source Code — A computer code in its originally coded form, typically in text file format. For programs written in a compilable programming language, the uncompiled program.

System Software — Software designed to enable the operation and maintenance of a computer system and its associated computer programs. [NQA-1]

Test Plan (Procedure) — A document that describes the approach to be followed for testing a system or component. Typical contents identify the items to be tested, tasks to be performed, and responsibilities for the testing activities. [NQA-1]

Testing — An element of verification for the determination of the capability of an item to meet specified requirements by subjecting the item to a set of physical, chemical, environmental, or operating conditions. [NQA-1]

Toolbox Codes — A small number of standard computer models (codes) supporting DOE safety analysis, having widespread use, and of appropriate qualification that are maintained, managed, and distributed by a central source. Toolbox codes meet minimum quality assurance criteria. They may be applied to support 10 CFR 830 DSAs provided the application domain and input parameters are valid. In addition to public domain software, commercial or proprietary software may also be considered. In addition to safety analysis software, design codes may also be included if there is a benefit to maintain centralized control of the codes [modified from DOE N 411.1].

User Manual — A document that presents the information necessary to employ a system or component to obtain desired results. Typically described are system or component capabilities, limitations, options, permitted inputs, expected outputs, possible error messages, and special instructions. Note: A user manual is distinguished from an operator manual when a distinction is made between those who operate a computer system (mounting tapes, etc.) and those who use the system for its intended purpose. Syn: User Guide. [IEEE 610-12]

Validation – 1. The process of testing a computer program and evaluating the results to ensure compliance with specified requirements [ANSI/ANS-10.4-1987].
2.The process of determining the degree to which a model is an accurate representation of the real-world from the perspective of the intended uses of the model [Department of Defense Directive 5000.59, *DoD Modeling and Simulation (M&S) Management*].

Verification – 1. The process of evaluating the products of a software development phase to provide assurance that they meet the requirements defined for them by the previous phase [ANSI/ANS-10.4-1987].
2. The process of determining that a model implementation accurately represents the developer's conceptual description and specifications [Department of Defense Directive 5000.59, *DoD Modeling and Simulation (M&S) Management*].

7.0 References

Note: The references listed below may not have been used directly in the gap analysis. However, they were used to provide a context for performing the overall code evaluation.

CFR Code of Federal Regulations (10 CFR 830). 10 CFR 830, Nuclear Safety Management Rule.

DNFSB Defense Nuclear Facilities Safety Board, (2000). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, Technical Report DNFSB/TECH-25, (January 2000).

DNFSB Defense Nuclear Facilities Safety Board, (2002). *Recommendation 2002-1, Quality Assurance for Safety-Related Software*, (September 2002).

DOE, U.S. Department of Energy (2000a). *Appendix A, Evaluation Guideline,* DOE-STD-3009-94, *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports* (January 2000).

DOE, U.S. Department of Energy (2000b). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, DOE Response to TECH-25, Letter and Report, (October 2000).

DOE, U.S. Department of Energy (2002). *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports*, DOE-HDBK-3010-94, Change Notice 2 (April 2002).

DOE, U.S. Department of Energy (2003a). *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).

DOE, U.S. Department of Energy (2003b). *Designation of Initial Safety Analysis Toolbox Codes*, Letter, (March 28, 2003).

DOE, U.S. Department of Energy (2003c). *Assessment Criteria and Guidelines for Determining the Adequacy of Software Used in the Safety Analysis and Design of Defense Nuclear Facilities*, Report, CRAD-4.2.4-1, Rev 0, (August 27 2003).

DOE, U.S. Department of Energy (2003d). *Software Quality Assurance Improvement Plan: Format and Content For Code Guidance Reports*, Revision A (draft), Report, (August 2003).

DOE, U.S. Department of Energy (2003e). *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes*, Revision 1, (November 2003).

DOE, U.S. Department of Energy (2004). *MACCS2 Computer Code Application Guidance for Documented Safety Analysis*, (May 2004).

Mills, M.T. (1987). "Modeling the Release and Dispersion of Toxic Combustion Products from Chemical Fires," paper presented at the International Conference on Vapor Cloud Modeling, November 2–4, 1987, Cambridge, MA. (Obtained from the author at ENSR Consulting and Engineering, Acton, MA.)

SNL (1986). *Sandia Software Guidelines: Volume 3: Standards, Practices, and Conventions.* Sandia National Laboratories, Albuquerque, NM, SAND85-2346.

SNL (1987). *Sandia Software Guidelines: Volume 1: Software Quality Planning.* Sandia National Laboratories, Albuquerque, NM, SAND85-2344.

SNL (1989). *Sandia Software Guidelines: Volume 5: Tools, Techniques, and Methodologies.* Sandia National Laboratories, Albuquerque, NM, SAND85-2348.

SNL (1992). *Sandia Software Guidelines: Volume 4: Configuration Management.* Sandia National Laboratories, Albuquerque, NM, SAND85-2347.

SNL (1995). *Sandia Software Guidelines: Volume 2: Documentation.* Sandia National Laboratories, Albuquerque, NM, SAND85-2345.

Appendices

Appendix	Subject
A	Software Information Template
B	Operation and Maintenance Criterion

APPENDIX A. SOFTWARE INFORMATION TEMPLATE

The following is a condensed version of the information request sent to the MACCS2 code developer in October 2003.

Information Form

Development and Maintenance of Designated Safety Analysis Toolbox Codes

The following summary information in Table 2 should be completed to the level that is meaningful – enter N/A if not applicable. See Appendix A for an example of the input to the table prepared for the MACCS2 code.

Table 2. Summary Description of Subject Software

Table 2. Summary Description of Subject Software	
Type	**Specific Information**
Code Name	
Version of the Code	
Developing Organization and Sponsor Information	
Auxiliary Codes	
Software Platform/Portability	
Coding and Computer(s)	
Technical Support Point of Contact	
Code Procurement Point of Contact	
Code Package Label/Title	
Contributing Organization(s)	
Recommended Documentation - Supplied with Code Transmittal upon Distribution or Otherwise Available	1. 2. 3. 4. 5.
Input Data/Parameter Requirements	

Table 2. Summary Description of Subject Software	
Type	**Specific Information**
Summary of Output	
Nature of Problem Addressed by Software	
Significant Strengths of Software	
Known Restrictions or Limitations	
Preprocessing (set-up) time for Typical Safety Analysis Calculation	
Execution Time	
Computer Hardware Requirements	
Computer Software Requirements	
Other Versions Available	

Table 3. Point of Contact for Form Completion

Individual(s) completing this information form: Name: Organization: Telephone: Email: Fax:	

1. Software Quality Assurance Plan

The software quality assurance plan for your software may be either a standalone document, or embedded in other documents, related procedures, QA assessment reports, test reports, problem reports, corrective actions, supplier control, and training package.

1.a For this software, identify the governing Software Quality Assurance Plan (SQAP)? [Please submit a PDF of the SQAP, or send hard copy of the SQAP[9]]

1.b What software quality assurance industry standards are met by the SQAP?

1.c What federal agency standards were used, if any, from the sponsoring organization?

1.d Has the SQAP been revised since the current version of the Subject Software was released? If so, what was the impact to the subject software?

1.e Is the SQAP proceduralized in your organization? If so, please list the primary procedures that provide guidance.

Guidance for SQA Plans:

Requirement 2 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a)
ASME NQA-1 2000 Section 200
IEEE Standard 730, *IEEE Standard for Software Quality Assurance Plans.*
IEEE Standard 730.1, *IEEE Guide for Software Quality Assurance Planning.*

2. Software Requirements Description

The software requirements description (SRD) should contain functional and performance requirements for the subject software. It may be contained in a standalone document or embedded in another document, and should address functionality, performance, design constraints, attributes and external interfaces.

2.a For this software, was a software requirements description documented with the software sponsor? [If available, please submit a PDF of the Software Requirements Description, or include hard copy with transmittal of SQAP]

[9] Notify Kevin O'Kula of your intent to send hard copies of requested reports and shipping will be arranged.

2.b If a SRD was not prepared, are there written communications that indicate agreement on requirements for the software? Please list other sources of this information if it is not available in one document.

Guidance for Software Requirements Documentation:

Requirement 5 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 401
IEEE Standard 830, *Software Requirements Specifications*

3. Software Design Documentation

The software design documentation (SDD) depicts how the software is structured to satisfy the requirements in the software requirements description. It should be defined and maintained to ensure that software will serve its intended function. The SDD for the subject software may be contained in a standalone document or embedded in another document.

The SDD should provide the following:

- Description of the major components of the software design as they relate to the software requirements,
- Technical description of the software with respect to the theoretical basis, mathematical model, control flow, data flow, control logic, and data structure,
- Description of the allowable or prescribed ranges of inputs and outputs,
- Design described in a manner suitable for translating into computer coding, and
- Computer program listings (or suitable references).

3.a For the subject software, was a software design document prepared, or were its constituents parts covered elsewhere? [If available, please submit a PDF of the Software Design Document, or include hard copy with transmittal of SQAP]

3.b If the intent of the SDD information is satisfied in other documents, provide the appropriate references (document number, section, and page number).

Guidance for Software Design Documentation:

Requirement 6 - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
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*
IEEE Standard 1012, *IEEE Standard for Software Verification and Validation;*
IEEE Standard 1012a, *IEEE Standard for Software Verification and Validation – Supplement to 1012*

4. Software User Documentation

Software User Documentation is necessary to assist the user in installing, operating, managing, and maintaining the software, and to ensure that the software satisfies user requirements. At minimum, the documentation should describe:

- The user's interaction with the software
- Any required training
- Input and output specifications and formats, options
- Software limitations
- Error message identification and description, including suggested corrective actions to be taken to correct those errors, and
- Other essential information for using the software.

4.a **For the subject software, has Software User Documentation been prepared, or are its constituents parts covered elsewhere?** [If available, please submit a PDF of the Software User Documentation, or include a hard copy with transmittal of SQAP]

4.b **If the intent of the Software User Documentation information is satisfied in other documents, provide the appropriate references (document number, section, and page number).**

4.c **Training – How is training offered in correctly running the subject software? Complete the appropriate section in the following:**

Type	Description	Frequency of training
Training Offered to User Groups as Needed		
Training Sessions Offered at Technical Meetings or Workshops		
Training Offered on Web or Through Video Conferencing		
Other Training Modes		
Training Not Provided		

Type	Description	Frequency of training

Guidance for Software User Documentation:

Requirement 9 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 203
IEEE Standard 1063, *IEEE Standard for Software User Documentation*

5. Software Verification &Validation Documentation (Includes Test Reports)

Verification and Validation (*V&V*) documentation should confirm that a software V&V process has been defined, that V&V has been performed, and that related documentation is maintained to ensure that:

(a) The software adequately and correctly performs all intended functions, and
(b) The software does not perform any unintended function.

The software V&V documentation, either as a standalone document or embedded in other documents and should describe:

- The tasks and criteria for verifying the software in each development phase and validating it at completion,
- Specification of the hardware and software configurations pertaining to the software V&V
- Traceability to both software requirements and design
- Results of the V&V activities, including test plans, test results, and reviews (also see 5.b below)
- A summary of the status of the software's completeness
- Assurance that changes to software are subjected to appropriate V&V,
- V&V is complete, and all unintended conditions are dispositioned before software is approved for use, and
- V&V performed by individuals or organizations that are sufficiently independent.

5.a **For the subject software, identify the V&V Documentation that has been prepared.**
[If available, please submit a PDF of the Verification and Validation Documentation, or include a hard copy with transmittal of SQAP]

5.b **If the intent of the V&V Documentation information is satisfied in one or more other documents, provide the appropriate references (document number, section, and page number). For example, a "Test Plan and Results" report, containing a plan for software testing, the test results, and associated reviews may be published separately.**

5.c **Testing of software: What has been used to test the subject software?**

☐ Experimental data or observations
☐ Standalone calculations
☐ Another validated software
☐ Software is based on previously accepted solution technique

Provide any reports or written documentation substantiating the responses above.

Guidance for Software Verification & Validation, and Testing Documentation:

Requirement 6 – *Design Phase* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
Requirement 8 – *Testing Phase* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
Requirement 10 – *Acceptance Test* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 402 (Note: Some aspects of verification may be handled as part of the Design Phase).
ASME NQA-1 2000 Section 404 (Note: Aspects of validation may be handled as part of the Testing Phase).
IEEE Standard 1012, *IEEE Standard for Software Verification and Validation;*
IEEE Standard 1012a, *IEEE Standard for Software Verification and Validation – Supplement to 1012*
IEEE Standard 829, *IEEE Standard for Software Test Documentation.*
IEEE Standard 1008, *Software Unit Testing*

6. Software Configuration Management (SCM)

A process and related documentation for SCM should be defined, maintained, and controlled.

The appropriate documents, such as project procedures related to software change controls, should verify that a software configuration management process exists and is effective.

The following points should be covered in SCM document(s):

- A Software Configuration Management Plan, either in standalone form or embedded in another document,
- Configuration management data such as software source code components, calculational spreadsheets, operational data, run-time libraries, and operating systems,
- A configuration baseline with configuration items that have been placed under configuration control,
- Procedures governing change controls,
- Software change packages and work packages to demonstrate that (1) possible impacts of software modifications are evaluated before changes are made, (2) various software system products are examined for consistency after changes are made, and (3) software is tested according to established standards after changes have been made.

6.a **For the subject software, has a Software Configuration Management Plan been prepared, or are its constituent parts covered elsewhere?** [If available, please submit a PDF of the Software Configuration Management Plan and related procedures, or include hard copies with transmittal of SQAP].

6.b **Identify the process and procedures governing control and distribution of the subject software with users.**

6.c **Do you currently interact with a software distribution organization such as the Radiation Safety Information Computational Center (RSICC)?**

6.d A Central Registry organization, under the management and coordination of the Department of Energy's Office of Environment, Safety and Health (EH), will be responsible for the long-term maintenance and control of the safety analysis toolbox codes for DOE safety analysis applications. Indicate any questions, comments, or concerns on the Central Registry's role and the maintenance of the subject software.

Guidance for Software Configuration Management Plan Documentation:

Requirement 12 – *Configuration Control* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 203
IEEE Standard 828, *IEEE Standard for Software Configuration Management Plans.*

7. Software Problem Reporting and Corrective Action

Software problem reporting and corrective action documentation help ensure that a formal procedure for problem reporting and corrective action development for software errors and failures is established, maintained, and controlled.

A Software Error Notification and Corrective Action Report, procedure, or similar documentation, should be implemented to report, track, and resolve problems or issues identified in both software items, and in software development and maintenance processes. Documentation should note specific organizational responsibilities for implementation. Software problems should be promptly reported to affected organizations, along with corrective actions. Corrective actions taken ensure that:

- Problems are identified, evaluated, documented, and, if required, corrected,
- Problems are assessed for impact on past and present applications of the software by the responsible organization,
- Corrections and changes are executed according to established change control procedures, and
- Preventive actions and corrective actions results are provided to affected organizations.

Identify documentation specific to the subject software that controls the error notification and corrective actions. [If available, please submit a PDF of the Error Notification and Corrective Action Report documentation for the subject software (or related procedures). If this is not available, include hard copies with transmittal of SQAP].

7.aProvide examples of problem/error notification to users and the process followed to address the deficiency. Attach files as necessary.

7.bProvide an assessment of known errors or defects in the subject software and the planned action and time frame for correction.

Category of Error or Defect	Corrective Action	Planned schedule for correctic
Major		

Minor		

7.cIdentify the process and procedures governing communication of errors/defects related to the subject software with users.

Guidance for Error/Defect Reporting and Corrective Action Documentation:

Requirement 13 – *Error Impact* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 204
IEEE Standard 1063, *IEEE Standard for Software User Documentation*

8. Resource Estimates

If one or more plans, documents, or sets of procedures identified in parts one (1) through seven (7) do not exist, please provide estimates of the resources (full-time equivalent (40-hour) weeks, FTE-weeks) and the duration (months) needed to meet the specific SQA requirement.

Enter estimate in Table 4 only if specific document has not been prepared, or requires revision.

Table 4. Resource and Schedule for SQA Documentation

Plan/Document/Procedure	Resource Estimate (FTE-weeks)	Duration of Activity (months)
1. Software Quality Assurance Plan		
2. Software Requirements Document		
3. Software Design Document		
4. Test Case Description and Report		
5. Software Configuration and Control		
6. Error Notification and Corrective Action Report		
7. User's Instructions (User's Manual)		
8. Other SQA Documentation		

Comments or Questions:

9. **Software Upgrades**

Describe modifications planned for the subject software.

Technical Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

User Interface Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Software Engineering Improvements

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Other Planned Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Thank you for your input to the SQA upgrade process. Your experience and insights are critical towards successfully resolving the issues identified in DNFSB Recommendation 2002-1.

APPENDIX B. OPERATION AND MAINTENANCE CRITERION

B.1 Topical Area Assessment: Operation and Maintenance Phase

This area corresponds to the requirement entitled Operation and Maintenance in Table 3-3 of DOE (2003c).

B.1.1 Criterion Specification and Results

Table B-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table B-1 — Subset of Criteria for Operations and Maintenance Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
B.1	During this phase, software shall be controlled to remove latent errors (corrective maintenance), to respond to new or revised requirements (enhancement), or to adapt the software to changes in the operating environment (adaptive maintenance). Software modifications shall be approved, documented, verified and validated, and controlled in accordance with the related life cycle phases.	No.	A written procedure for corrective and adaptive maintenance was not available.
B.2	The validation of modifications shall be subject to selective regression testing to detect errors introduced during the modification of software or operating system components to verify that the modifications have not caused unintended adverse effects and to verify that the modified software still meets its specified requirements.	Partial.	Discussions with MACCS2 consultant indicate that some regression testing was performed. It is not clear if this activity was performed sporadically in the second phase of MACCS2 development or followed a regular schedule.
B.3	Test cases shall be developed and documented to permit confirmation of acceptable performance of the software in the environment in which the software is used. Test cases shall be run whenever the software is installed on a different computer, or when significant hardware or operating system configuration changes are made.	Partial.	Test cases were formally drawn up for Version 1.12 of MACCS2.
B.4	Periodic in-use manual or automatic self-check in-use tests shall be prescribed and performed for those	N/A.	This evaluation criterion was interpreted to be applicable mostly for process control software. It was not

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	computer programs where computer program errors, data errors, computer hardware failures, or instrument drift can affect required performance.		addressed here due to the safety analysis nature of the MACCS2 software.

B.1.2 Sources and Method of Review

This review was based on information contained in East (1998b) and Bixler (2000). It also includes discussions with SNL code developers and David Chanin. Both discussions occurred in January 2004.

B.1.3 Software Quality-Related Issues or Concerns

While some maintenance activities such as configuration control and a software reporting system were in place during the development of MACCS2 Version 1.12, other activities falling under Operations and Maintenance appear to have been performed. However, there is no written record for confirmation purposes.

B.1.4 Recommendations

Of the four criteria evaluated for this requirement, one (1) is not met, two (2) are partially met, and one is judged as not applicable. Thus the requirement is not met.

It is advised that MACCS2 developer consider Operations and Maintenance processes as a tool to check software change effects. Specifically, a process should describe removal of latent errors (corrective maintenance) and adaptation of the software to changes in the operating environment (adaptive maintenance). Regression testing procedures should be developed to ensure that software modifications do not introduce unintended adverse effects. Test case recommendations are made in the body of this report.

SEPARATION

PAGE

Defense Nuclear Facilities Safety Board Recommendation 2002-1
Software Quality Assurance Improvement Plan
Commitment 4.2.1.3:

Software Quality Assurance Improvement Plan: GENII Gap Analysis

Final Report

U.S. Department of Energy
Office of Environment, Safety, and Health
1000 Independence Ave., S.W.
Washington, DC 20585-2040

May 2004

INTENTIONALLY BLANK

FOREWORD

This document provides an evaluation of the Software Quality Assurance (SQA) attributes of GENII, a radiological dispersion computer code, relative to established requirements. This evaluation, a "gap analysis", is performed to meet commitment 4.2.1.3 of the Department of Energy's Implementation Plan to resolve SQA issues identified in the Defense Nuclear Facilities Safety Board Recommendation 2002-1. Both versions of the GENII code (1.485 and 2.0) are addressed.

Suggestions for corrections or improvements to this document should be addressed to –

Chip Lagdon
EH-31/GTN
U.S. Department of Energy
Washington, D.C. 20585-2040
Phone (301) 903-4218
Email: chip.lagdon@eh.doe.gov

INTENTIONALLY BLANK

REVISION STATUS

Page/Section	Revision	Change
1. Entire Document	1. Interim Report	1. Original Issue
2. Entire Document	2. Final Report, May 3, 2004	2. Updated all sections per review comments.

INTENTIONALLY BLANK

CONTENTS

Section			Page
FOREWORD			**III**
REVISION STATUS			**V**
EXECUTIVE SUMMARY			**XIII**
1.0	**INTRODUCTION**		**1-1**
	1.1	BACKGROUND: OVERVIEW OF DESIGNATED TOOLBOX SOFTWARE IN THE CONTEXT OF 10 CFR 830	1-1
	1.2	EVALUATION OF TOOLBOX CODES	1-2
	1.3	USES OF THE GAP ANALYSIS	1-2
	1.4	SCOPE	1-3
	1.5	PURPOSE	1-3
	1.6	METHODOLOGY FOR GAP ANALYSIS	1-3
	1.7	SUMMARY DESCRIPTION OF SOFTWARE BEING REVIEWED	1-5
2.0	**ASSESSMENT SUMMARY RESULTS**		**2-7**
	2.1	CRITERIA MET	2-7
	2.2	EXCEPTIONS TO REQUIREMENTS	2-7
	2.3	AREAS NEEDING IMPROVEMENT	2-7
	2.4	AREAS NOT ASSESSED AND ANY LIMITATIONS OF GAP ANALYSIS	2-8
	2.5	CONCLUSION REGARDING SOFTWARE'S ABILITY TO MEET INTENDED FUNCTION	2-8
3.0	**LESSONS LEARNED**		**3-1**
4.0	**ASSESSMENT DETAILED RESULTS**		**4-1**
	4.1	TOPICAL AREA 1 ASSESSMENT: SOFTWARE CLASSIFICATION	4-2
		4.1.1 Criterion Specification and Result	*4-2*
		4.1.2 Sources and Method of Review	*4-3*
		4.1.3 Software Quality-Related Issues or Concerns	*4-3*
		4.1.4 Other Areas for Improvement	*4-3*
		4.1.5 Recommendations	*4-3*
	4.2	TOPICAL AREA 2 ASSESSMENT: SQA PROCEDURES AND PLANS	4-3
		4.2.1 Criterion Specification and Result	*4-3*
		4.2.2 Sources and Method of Review	*4-6*
		4.2.3 Software Quality-Related Issues or Concerns	*4-6*
		4.2.4 Other Areas for Improvement	*4-6*
		4.2.5 Recommendations	*4-6*
	4.3	TOPICAL AREA 3 ASSESSMENT: REQUIREMENTS PHASE	4-6
		4.3.1 Criterion Specification and Result	*4-6*
		4.3.2 Sources and Method of Review	*4-8*
		4.3.3 Software Quality-Related Issues or Concerns	*4-8*
		4.3.4 Other Areas for Improvement	*4-8*
		4.3.5 Recommendations	*4-8*

4.4 TOPICAL AREA 4 ASSESSMENT: DESIGN PHASE 4-8
4.4.1 Criterion Specification and Result *4-8*
4.4.2 Sources and Method of Review *4-14*
4.4.3 Software Quality-Related Issues or Concerns *4-14*
4.4.4 Other Areas for Improvement *4-14*
4.4.5 Recommendations *4-14*
4.5 TOPICAL AREA 5 ASSESSMENT: IMPLEMENTATION PHASE 4-14
4.5.1 Criterion Specification and Result *4-14*
4.5.2 Sources and Method of Review *4-16*
4.5.3 Software Quality-Related Issues or Concerns *4-16*
4.5.4 Other Areas for Improvement *4-16*
4.5.5 Recommendations *4-16*
4.6 TOPICAL AREA 6 ASSESSMENT: TESTING PHASE 4-16
4.6.1 Criterion Specification and Result *4-16*
4.6.2 Sources and Method of Review *4-18*
4.6.3 Software Quality-Related Issues or Concerns *4-19*
4.6.4 Other Areas for Improvement *4-19*
4.6.5 Recommendations *4-19*
4.7 TOPICAL AREA 7 ASSESSMENT: USER INSTRUCTIONS 4-19
4.7.1 Criterion Specification and Result *4-19*
4.7.2 Sources and Method of Review *4-21*
4.7.3 Software Quality-Related Issues or Concerns *4-21*
4.7.4 Other Areas for Improvement *4-22*
4.7.5 Recommendations *4-22*
4.8 TOPICAL AREA 8 ASSESSMENT: ACCEPTANCE TEST 4-22
4.8.1 Sources and Method of Review *4-24*
4.8.2 Software Quality-Related Issues or Concerns *4-24*
4.8.3 Other Areas for Improvement *4-24*
4.8.4 Recommendations *4-24*
4.9 TOPICAL AREA 9 ASSESSMENT: CONFIGURATION CONTROL 4-25
4.9.1 Criterion Specification and Result *4-25*
4.9.2 Sources and Method of Review *4-26*
4.9.3 Software Quality-Related Issues or Concerns *4-26*
4.9.4 Other Areas for Improvement *4-26*
4.9.5 Recommendations *4-26*
4.10 TOPICAL AREA 10 ASSESSMENT: ERROR IMPACT 4-26
4.10.1 Criterion Specification and Result *4-26*
4.10.2 Sources and Method of Review *4-27*
4.10.3 Software Quality-Related Issues or Concerns *4-27*
4.10.4 Other Areas for Improvement *4-28*
4.10.5 Recommendations *4-28*
4.11 TRAINING PROGRAM ASSESSMENT 4-28

5.0 CONCLUSION **5-1**

6.0 ACRONYMS AND DEFINITIONS **6-1**

7.0 REFERENCES **7-1**

APPENDIX A. — COMMUNICATIONS WITH OTHERS A-1

APPENDIX B. — GENII BENCHMARKING AND V&V B-1

PUBLICATIONS ON GENII VERIFICATION AND VALIDATION B-1
ADDITIONAL GENII BENCHMARKING AND COMPARISONS B-2
SUMMARY OF DEVELOPER/USER TESTING AND PEER REVIEW OF GENII FOR WHICH DOCUMENTATION IS AVAILABLE B-2

INTENTIONALLY BLANK

TABLES

	Page
Table 1-1 — Plan for SQA Evaluation of Existing Safety Analysis Software	1-3
Table 1-2 — Software Documentation Reviewed for GENII	1-5
Table 2-1 — Summary of Important Exceptions, Reasoning, and Suggested Remediation for GENII 2.0	2-7
Table 2-2 — Summary of Important Recommendations for GENII	2-8
Table 3-1 — Lessons Learned	3-1
Table 4.0-1 — Cross-Reference of Requirements with Subsection and Entry from DOE (2003e)	4-1
Table 4.1-1 — Subset of Criteria for Software Classification Topic and Results	4-2
Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results	4-3
Table 4.2-2 — Recommendations for SQA Procedures and Plans Topic	4-6
Table 4.3-1 — Subset of Criteria for Requirements Phase Topic and Results	4-7
Table 4.3-2 — Recommendations for Requirements Phase Topic	4-8
Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results	4-8
Table 4.4-2 — Recommendations for Design Phase Topic	4-14
Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results	4-15
Table 4.5-2 — Recommendations for Implementation Phase Topic	4-16
Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results	4-17
Table 4.6-2 — Recommendations for Testing Phase Topic	4-19
Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results	4-19
Table 4.7-2 — Recommendations for User Instructions Topic	4-22
Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results	4-23
Table 4.8-2 — Recommendations for Acceptance Test Topic	4-25
Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results	4-25
Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results	4-26
Table 4.10-2 — Recommendations for Error Impact Topic	4-28

FIGURES

Page

Figure 4-1 — Error reporting / update request form for GENII 1.485 4-5

Software Quality Assurance Improvement Plan: GENII Gap Analysis

EXECUTIVE SUMMARY

The Defense Nuclear Facilities Safety Board (DNFSB) issued Recommendation 2002-1 on *Quality Assurance for Safety-Related Software* in September 2002 (DNFSB 2002). The Recommendation identified a number of quality assurance issues for software used in the 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 high-use, Software Quality Assurance (SQA)-compliant safety analysis codes is one of the major improvement actions discussed in the *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities* (DOE 2003a). A DOE safety analysis toolbox would contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed, and maintained for DOE-broad safety basis applications.

The GENII software, for radiological dispersion and consequence analysis, is one of the codes designated for the toolbox. To determine the actions needed to bring the GENII code into compliance with the SQA qualification criteria, and develop an estimate of the resources required to perform the upgrade, the Implementation Plan has committed to sponsoring a code-specific gap analysis document. The gap analysis evaluates the software quality assurance attributes of GENII against identified criteria.

The balance of this document provides the outcome of the GENII gap analysis compliant with NQA-1-based requirements as contained in U.S. Department of Energy, *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes*, (DOE, 2003e). For GENII 1.485, of the ten general topical quality areas that were evaluated for software developers, nine met the criteria fully, and one failed to meet the criteria. For GENII 2.0, of the ten general topical quality areas, two met the criteria fully, five met the criteria partially, and three failed to meet the criteria. Recommendations are given for each of the topical areas in Section 4.0. The GENII code was evaluated to determine if the code, as it currently stands, meets the intended function for the code in the context as described in the scope of this gap analysis. When the code is run for the intended applications, as detailed in the code guidance document, *GENII Computer Code Application Guidance for Documented Safety Analysis*, (DOE 2004), it is judged that GENII 1.485 will meet its intended function, but GENII 2.0 will not. Therefore, only GENII 1.485 can be recommended for DSA use at this time. Note, however, that the GENII guidance document (DOE 2004) specifies that GENII 1.485 should be run either on a DOS-based computer, or in a DOS window of a Windows-95 or -98 based computer, not in a DOS window of a Windows XP based computer where problems may be encountered.

It is estimated that nearly ten full-time equivalent (FTE) months would be required to perform all SQA upgrade tasks identified in Section 4.0 of this report for GENII 2.0.

Of the ten primary SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), nine requirements are met at an acceptable level for GENII 1.485, (items 1-9). Improvement actions are recommended for GENII 1.485 to fully meet the requirement for *Error Impact* (item 10). For GENII 2.0, of the ten primary SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), two requirements are met at an acceptable level, i.e., *Software Classification* (1) and *Configuration Control* (9). Improvement actions are recommended for

GENII 2.0 to fully meet the requirement for five that are partially met, i.e., *SQA Procedures and Plans* (2), *Requirements Phase* (3), *Design Phase* (4), *Implementation Phase* (5), and *User Instructions* (7) and for the remaining three, *Testing Phase* (6), *Acceptance Test* (8), and *Error Impact* (10). This evaluation outcome is deemed acceptable because: (1) GENII 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 GENII is limited to those analytic applications for which the software is intended.

By order of priority, it is recommended that GENII software improvement actions be taken, especially:

1. correct known defects
2. upgrade user technical support activities
3. provide training on a regular basis, and
4. revise software documentation.

Performing these four primary actions should satisfactorily improve the SQA compliance status of GENII relative to the primary evaluation criteria cited in this report.

A new software baseline set of documents is recommended for GENII 2.0 to demonstrate completion of item 4 (above), revise software documentation. The list of baseline documents for revision includes:

1. Software Quality Assurance Plan
2. Software Model Description, including, but not limited to,
    a. Software Requirements
    b. Software Design
3. User's Manual, including, but not limited to,
    a. User Instructions
    b. Test Case Description and Report
    c. Software Configuration and Control
4. Error Notification and Corrective Action Procedure.

While completion of the GENII 2.0 development is encouraged, current DOE DSA support should be through the earlier code version, GENII 1.485, run on a DOS-based computer or a Windows-95 or -98 based computer. No evidence was found of software-induced errors in GENII 1.485 that have led to non-conservatisms in nuclear facility operations or in the identification of facility controls.

1.0 Introduction

This document reports on the results of a gap analysis for the GENII computer code. Both versions of the code (1.485 and 2.0) are considered. The intent of the gap analysis is to determine the actions needed to bring the specific software into compliance with established Software Quality Assurance (SQA) criteria. A secondary aspect of this report is to develop an estimate of the level of effort required to upgrade GENII based on the gap analysis results.

1.1 Background: Overview of Designated Toolbox Software in the Context of 10 CFR 830

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 (DNFSB, 2000). 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.

While progress was made in resolving several of the issues raised in TECH-25, the DNFSB issued Recommendation 2002-1 on Quality Assurance for Safety-Related Software in September 2002. 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, the quality of the software used to design or develop safety-related controls, and the proficiency of personnel using the software. The Recommendation identified a number of quality assurance issues for software used in the 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 high-use, SQA-compliant safety analysis codes is one of the major commitments contained in the March 2003 *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities* (IP). In time, the DOE safety analysis toolbox will contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed and maintained for DOE-broad safety basis applications.

Six computer codes, including ALOHA (chemical release dispersion/consequence analysis), CFAST (fire analysis), EPIcode (chemical release dispersion/consequence analysis), GENII (radiological dispersion/consequence analysis), MACCS2 (radiological dispersion/consequence analysis), and MELCOR (leak path factor analysis), were designated by DOE for the toolbox (DOE/EH, 2003). It is found that these codes provide generally recognized and acceptable approaches for modeling source term and consequence phenomenology, and can be applied as appropriate to support accident analysis in Documented Safety Analyses (DSAs).

As one of the designated toolbox codes, GENII will likely require some degree of quality assurance improvement before meeting current SQA standards. The analysis documented herein is an evaluation of GENII, both versions 1.485 and 2.0, relative to current software quality assurance criteria. It assesses the extent of the deficiencies, or gaps, to provide DOE and the software developer the extent to which minimum upgrades are needed. The overall assessment is therefore termed a "gap" analysis.

1.2 Evaluation of Toolbox Codes

The quality assurance criteria identified in later sections of this report are defined as the set of established requirements, or bases, by which to evaluate each designated toolbox code. This gap analysis evaluation, is commitment 4.2.1.3 in the IP:

> Perform a SQA evaluation to the toolbox codes to determine the actions needed to bring the codes into compliance with the SQA qualification criteria, and develop a schedule with milestones to upgrade each code based on the SQA evaluation results.

This process is a prerequisite step for software improvement. It will allow DOE to determine the current limitations and vulnerabilities of each code as well as help define and prioritize the steps required for improvement.

Early in the SQA evaluation program, it was anticipated that each toolbox code owner would provide input information on the SQA programs, processes, and procedures used to develop their software. However, most of the designated toolbox software, including GENII, was developed without complete conformance to software quality standards. Furthermore, many of the software developer organizations cannot confirm that key processes were followed. Therefore, most of the SQA evaluation has been preceded with reconstructing software development processes based on anecdotal evidence and limited, supporting documentation.

For independence reasons, the gap analysis is performed by a SQA evaluator not affiliated with the GENII development program. While independent of the code developer, the SQA evaluators responsible for GENII are knowledgeable in the use of the software for accident analysis applications, and understand current software development standards.

1.3 Uses of the Gap Analysis

The gap analysis will provide information to DOE, code developers, and code users.

DOE will see the following benefits:

- Estimates of the resources required to perform modifications to designated toolbox codes
- Basis for schedule and prioritization to upgrade each designated toolbox code.

Each code developer will be provided:

- Information on areas where software quality assurance improvements are needed to comply with industry SQA standards and practices
- Specific areas for improvement for guiding development of new versions of the software.

DOE safety analysts and code users will benefit from:

- Improved awareness of the strengths, limits, and vulnerable areas of each computer code
- Recommendations for code use in safety analysis application areas.

1.4 Scope

The gap analysis is applicable to the GENII code, one of the six designated toolbox codes for safety analysis. While GENII is the subject of the current report, other safety analysis software considered for the toolbox in the future may be evaluated with the same process applied here. The template outlined in this document is applicable to analytical software as long as the primary criteria are ASME NQA-1, 10 CFR 830, and related DOE directives discussed in DOE (2003e).

1.5 Purpose

The purpose of this report is to document the gap analysis performed on the GENII code as part of DOE's implementation plan on SQA improvements.

1.6 Methodology for Gap Analysis

The gap analysis for GENII is based on the criteria as described in *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes* (DOE 2003e). The overall methodology for the gap analysis is summarized in Table 1-1. The gap analysis utilizes ten of the fourteen topical areas listed in DOE (2003e) related to software quality assurance to assess the GENII software. The ten areas are those particularly applicable to the software development, specifically: (1) Software Classification, (2) SQA Procedures/Plans, (5) Requirements Phase, (6) Design Phase, (7) Implementation Phase, (8) Testing Phase, (9) User Instructions, (10) Acceptance Test, (12) Configuration Control, and (13) Error Impact. Each area, or requirement, is assessed individually in Section 4.

Requirements 3 (Dedication), 4 (Evaluation), 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 evaluating the areas listed above. Requirement 11 (Operation and Maintenance) is only partially applicable to software development, and is interpreted to be applicable mostly to the software user organization.

Table 1-1 — Plan for SQA Evaluation of Existing Safety Analysis Software[1]

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 of DOE (2003e).
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.,

[1] Originally documented as Table 2-2 in DOE (2003e).

Table 1-1 — Plan for SQA Evaluation of Existing Safety Analysis Software (continued)

PHASE	Procedure
	• 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: • SQA 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.
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.

1.7 Summary Description of Software Being Reviewed

The gap analysis was performed on both versions of the GENII code (i.e., Version 1.485 [Napier, 1988a, 1988b, 1988c] and Version 2.0 [Napier, 1995, 2002a, 2002b, 2003]). Although the earlier version (1.485) is the one recommended for use in current DSAs, the later version (2.0) is also evaluated, because the improvements recommended here, if implemented, would allow it to be used in DSAs in the future. In the following discussion, RSICC refers to the Radiation Safety Information Computational Center at Oak Ridge, TN.

The set of documents reviewed as part of the gap analysis are listed in Table 1-2.

Table 1-2 — Software Documentation Reviewed for GENII

No.	Information	
1.	Reference:	B. A. Napier, R. A. Peloquin, D. L. Strenge, and J. V. Ramsdell, *GENII – The Hanford Environmental Radiation Dosimetry Software System. Volume 1: Conceptual Representation.* PNL-6584, December 1988. (Napier, 1988a)
	Remarks:	Documentation provided by RSICC in .pdf format
2.	Reference:	B. A. Napier, R. A. Peloquin, D. L. Strenge, and J. V. Ramsdell, *GENII – The Hanford Environmental Radiation Dosimetry Software System. Volume 2: User's Manual,* PNL-6584, November 1988. (Napier, 1988b)
	Remarks:	Documentation provided by RSICC in .pdf format
3.	Reference:	B. A. Napier, R. A. Peloquin, D. L. Strenge, and J. V. Ramsdell, *GENII – The Hanford Environmental Radiation Dosimetry Software System. Volume 3: Code Maintenance Manual,* PNL-6584, September 1988. (Napier, 1988c) Only the table of contents is available (included as part of the .pdf file of Volumes 1 and 2). Bruce Napier has one of the few copies of the entire document (Volume 3), which is about 1,500 pages long, but a copy was not available for this gap analysis.
	Remarks:	Table of contents in .pdf format provided by RSICC.
4.	Reference:	B. A. Napier, J. V. Ramsdell, and D. L. Strenge, *Software Requirements Specifications for Hanford Environmental Dosimetry Coordination Project*, Draft Report, prepared for review by the EPA Office of Radiation and Indoor Air, May 1995. (Napier, 1995)
	Remarks:	Documentation provided by Bruce Napier.
5.	Reference:	B. A. Napier, *GENII Version 2 User's Guide* (Napier, 2002a)
	Remarks:	Downloaded from PNNL website
6.	Reference:	B. A. Napier, D. L. Strenge, J. V. Ramsdell, Jr., P. W. Eslinger, and C. Fosmire, *GENII Version 2 Software Design Document* (Napier, 2002b)
	Remarks:	Downloaded from PNNL website
7.	Reference:	B. A. Napier, *GENII Version 2 Example Calculation Descriptions* (Napier, 1999a)
	Remarks:	Documentation on CD from EFCOG training class, June 1999
8.	Reference:	B. A. Napier and L. Staven, *GENII Version 2 Training Power Point Slides* (Napier, 1999b)
	Remarks:	Documentation on CD from EFCOG training class, June 1999

Table 1-2 — Software Documentation Reviewed for GENII (continued)

No.	Information	
9.	Reference:	B. A. Napier, *Getting Started with GENII Version 2* (Napier, 2003)
	Remarks:	Downloaded from EPA/NESHAPs website
10.	Reference:	B. A. Napier, E-mail communications with K. R. O'Kula and Vern Peterson
	Remarks:	Provided in Appendix A
11.	Reference:	W. E. Joyce, Telephone conversation with V. L. Peterson
	Remarks:	Provided in Appendix A
12.	Reference:	Publications supporting GENII Benchmarking and V&V
	Remarks:	Provided in Appendix B

2.0 Assessment Summary Results

2.1 Criteria Met

For GENII 1.485, of the applicable ten general topical quality areas, nine met the criteria fully, and one failed to meet the criteria. An exception was found in the area of Error Impact. An area for GENII 1.485 improvement is to create and follow a formal error reporting and corrective action process. For GENII 2.0, of the ten general topical quality areas, two met the criteria fully, five met the criteria partially, and three failed to meet the criteria. Exceptions were found in the areas of Testing Phase, Acceptance Test, Error Impact, and partially in the areas of SQA Procedures and Plans, Requirements Phase, Design Phase, Implementation Phase, and User Instructions.

The areas that should be addressed for improvement actions are listed in Section 2.2 (Exceptions to Requirements). Detail on the evaluation process relative to the requirements, and the criteria applied, are found in Section 4.

2.2 Exceptions to Requirements

Some of the more important exceptions to criteria found for GENII 2.0 are listed below in Table 2-1. No similar list is needed for GENII 1.485. The requirement is given, the reason the requirement was not met is provided, and remedial action(s) are listed to correct the exceptions. The ten criteria evaluated are those predominantly executed by the software developer. However, it is noted that criteria for SQA Procedures/Plan, Testing, Acceptance Test, Configuration Control, and Error Notification also have requirements for the organization implementing the software. These criteria were assessed in the present evaluation only from the code developer perspective.

Table 2-1 — Summary of Important Exceptions, Reasoning, and Suggested Remediation for GENII 2.0

No.	Criterion	Reason Not Met	Remedial Action(s)
1.	Testing Phase	Testing not yet complete	Document all testing of GENII 2.0
2.	Acceptance Test	Testing not yet complete	Develop and document acceptance criteria for GENII 2.0 and document acceptance testing.
3.	Error Impact	A formal error reporting and corrective action procedure is not followed.	Create and follow a formal error reporting and corrective action process (applies to GENII 1.485 as well)

2.3 Areas Needing Improvement

The gap analysis identified a number of improvements that could be made related to the code and its quality assurance. Some of the important ones are listed in Table 2-2.

Table 2-2 — Summary of Important Recommendations for GENII

No.	Recommendation
1.	Establish and follow formal review schedules for GENII 2.0.
2.	Make GENII 2.0 code listings available upon completion and final testing of code.
3.	Correct the user documentation (see Section 4.7.4) and the bugs in the user interface for GENII 2.0 (see Criterion 9.6).
4.	Run GENII 1.485 only on a DOS-based computer, or in a DOS window of a Windows-95 or –98 based computer. Problems may be encountered when GENII 1.485 is run under Windows-XP. It is recommended that GENII 1.485 be recompiled with a FORTRAN compiler that is compatible with Windows-XP, tested, and released as an updated version.
5.	Modify GENII 2.0 to make it easy for the user to determine 95th percentile consequences at the site boundary and at a user-selected collocated worker distance (for example, 100 m).
6.	Assemble the existing "software change packets" for GENII 1.485 into a document to verify that changes to the code followed a logical and verifiable process.

2.4 Areas Not Assessed and Any Limitations of Gap Analysis

All areas were assessed for this gap analysis. Some areas were found to be more difficult to assess than others, depending upon the level of detail provided in the documentation. However, no limitations were imposed on the gap analysis.

2.5 Conclusion Regarding Software's Ability to Meet Intended Function

The GENII code was evaluated to determine if the software, in its current state, meets the intended function in a safety analysis context as assessed in this gap analysis. When the code is run for the intended applications, as detailed in the code guidance document, *GENII Computer Code Application Guidance for Documented Safety Analysis* (DOE 2004), and also utilizing information from documentation available (Table 1-2), it is judged that GENII 1.485 will meet the intended function, but GENII 2.0 will not. Therefore, only GENII 1.485 can be recommended for DSA use at this time. Note, however, that the GENII guidance document (DOE 2004) specifies that GENII 1.485 should be run on a DOS-based computer, or in a DOS window of a Windows-95 or –98 based computer, not in a DOS window of a Windows-XP based computer where problems may be encountered.

The primary remedial actions required for GENII 2.0 include the following:

(1) Modify the software so that the user can determine the 95th percentile doses at the site boundary in all sectors
(2) Improve the user documentation
(3) Create an error-reporting and corrective action procedure, including its documentation
(4) Complete code testing and document it
(5) Create and implement a code maintenance procedure.

3.0 Lessons Learned

Table 3-1 provides a summary of the lessons learned during the performance of the GENII gap analysis.

Table 3-1 — Lessons Learned

No.	Lesson
1.	Changing criteria in SQA standards over the years can render codes non-compliant that were once compliant.
2.	Although the author of a code may intend the code to be compliant with SQA standards, the standards may present sufficient complexity so that some requirements are not met in total.
3.	Development of software that is compliant with SQA standards can be a costly and laborious endeavor, especially if it is back-fit to the software, instead of being a parallel requirement during software development. If funding for the project is meager, SQA will probably not be followed as closely as may have been intended originally. Completion of the code development may take precedence over SQA measures.
4.	Changing sponsors may impact the SQA pedigree of software. This situation can arise especially if more recent software development was driven by other, non-SQA requirements than were present originally. The current version of the code has been developed for Environmental Protection Agency (EPA)/National Emission Standards for Hazardous Air Pollutants (NESHAPS), while original versions of the code were funded out of the PNNL budget.

4.0 Assessment Detailed Results

Ten topical areas, or requirements, are presented in the assessment as listed in Table 4.0-1. Training and Software Improvements (resource estimate) sections follow the 10 topical areas. Included in the software improvements section is an estimate of the resources required to upgrade GENII.

In the tables that follow, the topical areas or requirements are labeled as (1.x, 2.x,...,10.x) with the first value corresponding to the topical area and the second value (x), the sequential table order of each criterion. Four qualitative values shall be used to evaluate whether a specific criterion is met:

- Yes – evidence is available to confirm that the program, practices, and/or procedures followed in developing the version of code satisfy the criterion.
- No – sufficient evidence does not exist to demonstrate that the code meets the criterion
- Partial – some evidence exists that the criterion is met, but has not been finalized or is incomplete
- Uncertain – no basis is available to confirm that the criterion is met.

The overall evaluation for a specific requirement is based on the evaluation of the software against the criteria.

Table 4.0-1 — Cross-Reference of Requirements with Subsection and Entry from DOE (2003e)

Subsection (This Report)	Corresponding Entry Table 3-3 from DOE (2003e)	Requirement	ASME NQA-1 2000 Section/Consensus Standards
4.1	1	Software Classification	ASME NQA-1 2000 Section 200
4.2	2	SQA Procedures/Plans	ASME NQA-1 2000 Section 200; IEEE Std. 730, *IEEE Standard for Software Quality Assurance Plans*
4.3	5	Requirements Phase	ASME NQA-1 2000 Section 401; IEEE Standard 830, *Software Requirements Specifications*
4.4	6	Design Phase	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*
4.5	7	Implementation Phase	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*

Table 4.0-1 — Cross-Reference of Requirements with Subsection and Entry from DOE (2003e) (continued)

Subsection (This Report)	Corresponding Entry Table 3-3 from DOE (2003e)	Requirement	ASME NQA-1 2000 Section/Consensus Standards
4.6	8	Testing Phase	ASME NQA-1 2000 Section 404; IEEE Std. 829, *IEEE Standard for Software Test Documentation;* IEEE Standard 1008, *Software Unit Testing*
4.7	9	User Instructions	ASME NQA-1 2000 Section 203; IEEE Standard 1063, *IEEE Standard for Software User Documentation*
4.8	10	Acceptance Test	ASME NQA-1 2000 Section 404; IEEE Std. 829, *IEEE Standard for Software Test Documentation;* IEEE Standard 1008, *Software Unit Testing*
4.9	12	Configuration Control	ASME NQA-1 2000 Section 405; ASME NQA-1 2000 Section 406
4.10	13	Error Notification	ASME NQA-1 2000 Section 203

4.1 Topical Area 1 Assessment: Software Classification

This area corresponds to the requirement entitled Software Classification in Table 3-3 of DOE (2003e).

4.1.1 Criterion Specification and Result

This topical area is "required" for both GENII 1.485 and 2.0. Table 4.1-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.1-1 — Subset of Criteria for Software Classification Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
1.1	The code developer must provide sufficient information to allow the user to make an informed decision on the classification of the software.	Yes for both	The documentation from the developer makes it clear that both GENII 1.485 and 2.0 are Level B software.

4.1.2 Sources and Method of Review

All of the documentation listed in Table 1-2 has been reviewed with attention to "Software Classification," except for Item 12 (see Appendix B).

4.1.3 Software Quality-Related Issues or Concerns

There are no other SQA-related issues or concerns in "Software Classification."

4.1.4 Other Areas for Improvement

No areas of improvement in "Software Classification" have been noted.

4.1.5 Recommendations

This requirement is met. There are no recommendations related to this Topical Area.

4.2 Topical Area 2 Assessment: SQA Procedures and Plans

This area corresponds to the requirement entitled *SQA Procedures and Plans* in Table 3-3 of the DOE SQA plan (DOE 2003e). It deals with the planning efforts prior to code development.

4.2.1 Criterion Specification and Result

This topical area is "required" for both GENII 1.485 and 2.0. Table 4.2-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
2.1	Procedures/plans for SQA (SQA Plan) have identified organizations responsible for performing work, independent reviews, etc.	Yes for both	Pacific Northwest National Laboratory (PNNL) (formerly Pacific Northwest Laboratory [PNL]) is responsible for performing the work and providing for independent reviews (Napier, 1988a) and Napier (1995)

Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results (continued)

Criterion Number	Criterion Specification	Compliant	Summary Remarks
2.2	Procedures/plans for SQA (SQA Plan) have identified software engineering methods.	Yes for both	The software engineering methods are discussed in Napier (1988a) and Napier (1995)
2.3	Procedures/plans for SQA (SQA Plan) have identified documentation to be required as part of program.	Yes for both	Required documentation is discussed in Napier (1988a) and Napier (1995)
2.4	Procedures/plans for SQA (SQA Plan) have identified standards, conventions, techniques, and/or methodologies that shall be used to guide the software development, methods to ensure compliance with the same.	Yes for both	The standards, conventions, techniques, and/or methodologies that were used to guide code development are discussed in Napier (1988a) and Napier (1995).
2.5	Procedures/plans for SQA (SQA Plan) have identified software reviews and schedule.	Yes for 1.485. No for 2.0.	Napier (1988a) discusses two formal review periods for GENII 1.485. No similar discussion is in the GENII 2.0 documentation.
2.6	Procedures/plans for SQA (SQA Plan) have identified methods for error reporting and corrective actions.	Yes for 1.485. No for 2.0	Napier (1988b) discusses how to report errors and request upgrades. An informal method is used for GENII 2.0.

Additional Detail

The following provides additional detailed explanation on selected criteria in the above table:

Criterion 2.1 — The GENII 1.485 system was developed under the direction of the DOE office at Hanford for use by nuclear safety analysts. Potential user groups were identified and representatives of these groups were then selected to form a committee to specify the software requirements. Other groups were identified to provide reviews of the design and perform independent testing. The documentation describes these groups by their functions and the names of individual members are given in the "Acknowledgements" section. The organization selected to perform the work was the PNL (now PNNL). The GENII 2.0 system was developed with funding from the EPA. It incorporates much of the code developed for GENII 1.485 but was developed for use by the EPA in Environmental Impact Statements (EISs). The various groups for review and testing are mentioned in Napier (1995), which is the SQA plan for GENII 2.0.

Criterion 2.2 — An appendix to the GENII 1.485 volume 1 (Napier, 1988a) is a detailed system-requirements document. In it, software engineering methods are discussed. For GENII 2.0, the system requirements are given in Napier (1995), which discusses software engineering. (However, the word "engineering" is not used in either document.)

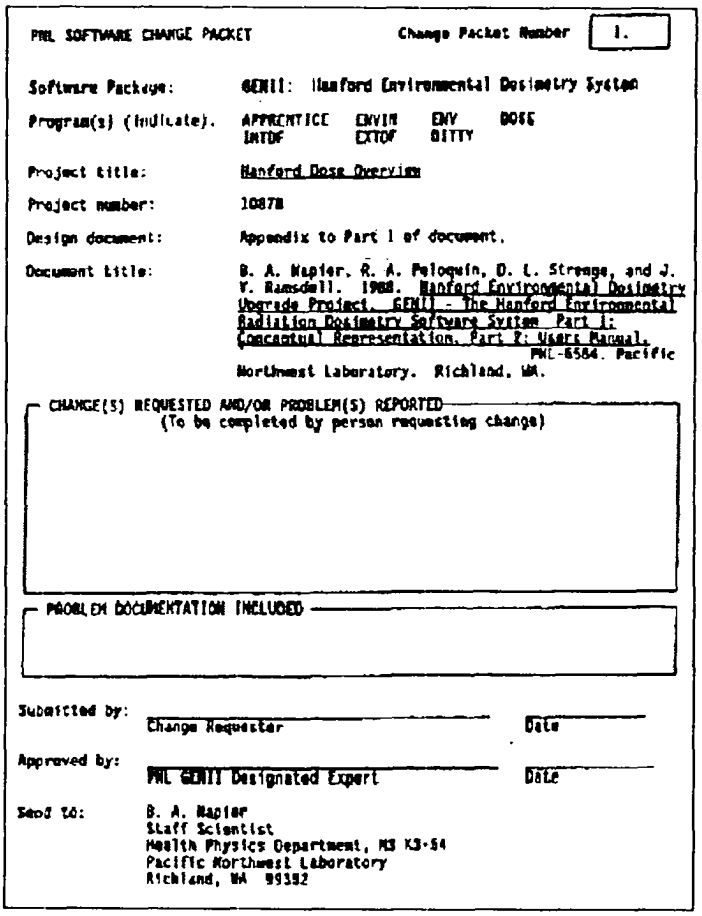
Criterion 2.3 — The GENII 1.485 documentation (Napier, 1988a, 1988b) identified several required documents, including requirements for the overall system, design, implementation, testing, user manual, and maintenance. Likewise, Napier (1995) discusses the planned documentation for GENII 2.0.

Criterion 2.4 — Napier (1988a) and Napier (1995) discuss the standards, conventions, techniques, and/or methodologies to be used to guide code development. Napier (1988a) was

prepared during and after the development of GENII 1.485 and is, thus, more detailed than Napier (1995), which was prepared before the development of GENII 2.0.

Criterion 2.5 -- External peer reviews of GENII 1.485 were conducted during the weeks beginning September 14, 1987 and February 1, 1988. This was followed by a formal acceptance of the code upon completion of the documentation packages for the user. Review schedules are not discussed in the GENII 2.0 documentation.

Criterion 2.6 — A formal error-reporting methodology was used for GENII 1.485. A copy of the reporting form is shown in Figure 4-1. For GENII 2.0, error reporting is informal, as evidenced by e-mail from Napier (see Appendix A) that includes the statement "I only have a few beta users; they let me know when it's broke and I fix it for them."



PNL SOFTWARE CHANGE PACKET Change Packet Number 1.

Software Package: GENII: Hanford Environmental Dosimetry System

Program(s) (indicate). APPRENTICE ENVIN ENV DOSE INTDF EXTDF DITTY

Project title: Hanford Dose Overview

Project number: 10878

Design document: Appendix to Part 1 of document.

Document title: B. A. Napier, R. A. Peloquin, D. L. Strenge, and J. V. Ramsdell. 1988. Hanford Environmental Dosimetry Upgrade Project. GENII - The Hanford Environmental Radiation Dosimetry Software System Part 1: Conceptual Representation. Part 2: Users Manual. PNL-6584. Pacific Northwest Laboratory. Richland, WA.

CHANGE(S) REQUESTED AND/OR PROBLEM(S) REPORTED
(To be completed by person requesting change)

PROBLEM DOCUMENTATION INCLUDED

Submitted by: Change Requester Date

Approved by: PNL GENII Designated Expert Date

Send to: B. A. Napier
Staff Scientist
Health Physics Department, MS K3-54
Pacific Northwest Laboratory
Richland, WA 99352

Figure 4-1 — Error reporting / update request form for GENII 1.485

4.2.2 Sources and Method of Review

All of the documentation listed in Table 1-2 has been reviewed with attention to "SQA Procedures and Plans," except for Item 12 (see Appendix B).

4.2.3 Software Quality-Related Issues or Concerns

Review schedules and a formal error reporting and corrective action methodology needs to be implemented for GENII 2.0.

4.2.4 Other Areas for Improvement

No other areas of improvement are noted.

4.2.5 Recommendations

Recommendations related to this topical area are provide in Table 4.2-2.

Table 4.2-2 — Recommendations for SQA Procedures and Plans Topic

Recom-mendation Number	Relates to Table 4.2-1 Criterion Number(s)	Recommendation	Est. FTE to Complete	Est. Calendar Duration
2.1	2.6	Implement a Formal Error Report (FER) and handling methodology for GENII 2.0. This is not required for GENII 1.485.	One FTE week	Two weeks
2.2	2.5	Establish formal review schedules for GENII 2.0.	One FTE day	One week

4.3 Topical Area 3 Assessment: Requirements Phase

This area corresponds to the requirement entitled *Requirements* Phase in Table 3-3 of the DOE SQA plan (DOE 2003e).

4.3.1 Criterion Specification and Result

This topical area is "required" for both GENII 1.485 and 2.0. Table 4.3-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.3-1 — Subset of Criteria for Requirements Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
3.1	Software requirements for the subject software have been established.	Yes for both	Software Requirements are in: 1.485: Napier (1988a) appendix 2.0: Napier (1995)
3.2	Software requirements are specified, documented, reviewed, and approved.	Yes for both	1.485: Software specifications, review, and approval are in Napier (1988a) and its appendix. 2.0: Requirements in Napier (1995). Review and approval implied by Napier (2002b).
3.3	Requirements define the functions to be performed by the software and provide detail and information necessary to design the software.	Yes for both	Detailed functional requirements are defined in: 1.485: Napier (1988a) appendix 2.0: Napier (1995)
3.4	A Software Requirements Document, or equivalent, defines requirements for functionality, performance, design inputs, design constraints, installation considerations, operating systems (if applicable), and external interfaces necessary to design the software.	Yes for both	Detailed functional requirements are defined in the System Requirements documents: 1.485: Napier (1988a) appendix 2.0: Napier (1995)
3.5	Acceptance criteria are established in the software requirements documentation for each of the identified requirements.	Yes for 1.485. Partial for 2.0	1.485: Napier (1988b, 1988c) 2.0: Acceptance criteria are not specifically described but are implied by testing requirements

Additional Detail

The following provides additional detailed explanation on selected criteria in the above table.

Criteria 3.1 and 3.2 — GENII 1.485 was developed by means of tasks designed to provide a state-of-the-art, technically peer-reviewed, and documented set of programs. The initial task resulted in a system design requirements report, based on input from potential Hanford users, providing general descriptions of the calculations that the final programs must perform. The recommendations of that report formed the basis for the remainder of the tasks, defining the elements that determined the equation formulation and parameter selection tasks (Napier, 1988a). The appendix to that document provides a discussion of SQA issues, including responsible organizations. Napier (1995) provides a similar discussion for GENII 2.0 and states the code was developed in a similar manner. The identified user groups are EPA analysts and contractors.

Criterion 3.5 — Napier (1988b, 1988c) discuss acceptance criteria and testing for GENII 1.485. The GENII 2.0 documentation does not specifically address acceptance criteria but implies their existence by referring to code testing.

4.3.2 Sources and Method of Review

All of the documentation listed in Table 1-2 has been reviewed with attention to "Requirements," except for Item 12 (see Appendix B).

4.3.3 Software Quality-Related Issues or Concerns

The only SQA concern for GENII 2.0 was the lack of specific acceptance criteria. There are no similar concerns for GENII 1.485.

4.3.4 Other Areas for Improvement

No other areas of improvement were noted.

4.3.5 Recommendations

Recommendations related to this topical area are provide in Table 4.3-2.

Table 4.3-2 — Recommendations for Requirements Phase Topic

Recommendation Number	Relates to Table 4.3-1 Criterion Number(s)	Recommendation	Est. FTE to Complete	Est. Calendar Duration
3.1	3.5	Develop and document acceptance criteria for GENII 2.0.	One FTE week	One month

4.4 Topical Area 4 Assessment: Design Phase

This area corresponds to the requirement entitled *Design Phase* in Table 3-3 of the DOE SQA plan (DOE 2003e).

4.4.1 Criterion Specification and Result

This topical area is "graded" for GENII 1.485 and "required" for GENII 2.0. Table 4.4-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.1	The software design was developed, documented, reviewed, and controlled.	Yes for both	1.485: Napier (1988a) provides System Requirements as well as software design. 2.0: Napier (2002b) is the System Design Document

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results (continued)

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.2	Code developer(s) 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.	Yes for both	1.485: Napier (1988a) provides System Requirements as well as software design activities. 2.0: Napier (2002b) is the System Design Document. Pseudo-code listings provided.
4.3	Design presents and documents specification of interfaces, overall structure (control and data flow) and the reduction of the overall structure into physical solutions (algorithms, equations, control logic, and data structures).	Yes for both	1.485: Napier (1988a, b, c) document overall structure, interfaces, control and data flow, and physical solutions. 2.0: Napier (1995, 2002b) document overall structure, interfaces, control and data flow, and physical solutions. Pseudo-code listings are provided. For both, diagrams show the flow of data and logic.
4.4	Design presents and documents that 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.	Yes for both	1.485: Napier (1988ab,c) show that the overall system design accounted for hardware and software interfaces and limitations, including the O/S. 2.0: Napier (1,995, 2002b) provides similar features.
4.5	Design presents and documents that as an integral part of software design, problems are mitigated. These potential problems include external and internal abnormal conditions and events that can affect the computer program.	Yes for 1.485. Partial for 2.0.	1.485: Napier (1988b) provides error-reporting forms to testers and users so that errors can be fixed and users informed. 2.0: the error-reporting is less formal
4.6	A Software Design Document, or equivalent, is available and contains a description of the major components of the software design as they relate to the software requirements.	Yes for both	1.485: Napier (1988a) describes major components of design 2.0: Napier (2002b) is the System Design Document. Pseudo-code listings provided.

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results (continued)

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.7	A Software Design Document, or equivalent, is available and contains 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.	Yes for both	1.485: Napier (1988a) provides the theoretical basis, control logic and flow, data flow and structure, mathematical models, process flow and structure, physical models, and coupling between structure and standards. 2.0: Napier (2002b) provides similar information. Pseudo-code listings are provided.
4.8	A Software Design Document, or equivalent, is available and contains a description of the allowable or prescribed ranges for inputs and outputs.	Yes for both	1.485: Napier (1988a) discusses ranges of input variables and error message generated when out of range. 2.0: Napier (2002b) provides similar information.
4.9	A Software Design Document, or equivalent, is available and contains the design described in a manner that can be translated into code.	Yes for both	1.485: Napier (1988a) and its appendix provide enough detail that the design can be translated into code 2.0: Napier (2002b) provides similar information. Pseudo-code listings are provided.
4.10	A Software Design Document, or equivalent, is available and contains 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.	Yes for both	1.485: Napier (1988a, b, c) discuss testing and the H/W and S/W configurations 2.0: Napier (1995, 2002b) provides similar information.
4.11	The organization responsible for the design identified and documented the particular verification methods to be used and assured that an Independent Review was performed and documented. This review evaluated the technical adequacy of the design approach; assured internal completeness, consistency, clarity, and correctness of the software design; and verified that the software design is traceable to the requirements.	Yes for 1.485. No for 2.0	1.485: Napier (1988a, b, c) states that the code has been thoroughly tested and verified by independent reviewers according to NQA-1 standards. 2.0: Because this code has not been completed in all its aspects, the final testing has not yet been done.

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results (continued)

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.12	The organization responsible for the design assured that the test results adequately demonstrated the requirements were met.	Yes for 1.485. No for 2.0	1.485: Napier (1988a, b, c) states that the code has been thoroughly tested and verified by independent reviewers according to NQA-1 standards. 2.0: Because this code has not been completed in all its aspects, the final testing has not yet been done.
4.13	The Independent Review was performed by competent individual(s) other than those who developed and documented the original design, but who may have been from the same organization.	Yes for 1.485. No for 2.0	1.485: Napier (1988a, b, c) states that the code has been thoroughly tested and verified by independent reviewers according to NQA-1 standards. This includes review by competent, independent individuals. 2.0: Because this code has not been completed in all its aspects, the final testing has not yet been done.
4.14	The results of the Independent Review are documented with the identification of the verifier indicated.	Yes for 1.485. No for 2.0	1.485: The independent reviewers are identified by name in the Acknowledgements section of Napier (1988a,b) 2.0: Because this code has not been completed in all its aspects, the final testing has not yet been done.
4.15	If review alone was not adequate to determine if requirements are met, alternate calculations were used, or tests were developed and integrated into the appropriate activities of the software development cycle.	Yes for both	During code development, extensive manual calculations were made (and archived) to verify proper performance of the code. See final paragraph of this subsection.

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results (continued)

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.16	Software design documentation was completed prior to finalizing the Independent Review.	Yes for both	1.485: Napier (1988a) states that the code has been thoroughly tested and verified by independent reviewers according to NQA-1 standards. This includes completion of S/W design prior to finalizing independent review. 2.0: Napier (2002b), the design document, has been completed. The final independent review has not yet occurred.
4.17	The extent of the Independent Review and the methods chosen are shown to be a function of the following: The importance to safety The complexity of the software The degree of standardization The similarity with previously proven software	Yes for both	These issues are decided by the independent reviewers, not the code developers. Therefore, they are not specifically addressed in the documentation of either version of GENII.

Additional Detail

The following provides additional detailed explanation on selected criteria in the above table:

Criterion 4.1 — The Napier (1988a) appendix, *Hanford Environmental Dosimetry Upgrade Project (HEDUP) Task 02 - System Design Requirements*, is the complete SQA requirements document for GENII 1.485. It includes the following:

1. General computational requirements
2. Computational facilities, hardware, and databases
3. Code language
4. Coding Standard and coding standard tools
5. Input parameters and format:
   - Release category and source term
   - Scenarios
   - Meteorology
   - Environmental transport
   - Exposure pathways
6. Dosimetry specifications
7. Risk assessment calculations
8. Integration of separate codes
9. Customized pathway requirements
10. Specialized scenario requirements
11. Output format

12. Graphics
13. Documentation and instructions
14. Error messages
15. Updates and revisions
16. Security
17. Quality assurance
18. Training

Napier (2002b) is the System Design Document for GENII 2.0. It defines details of the overall structure of the software, the major software components, their data file interfaces, and specific mathematical models to be used. The design represents a translation of the requirements (Napier, 1995) into a description of the software structure, software components, interfaces, and necessary data. The design focuses on the major components and data communication links that are key to the implementation of the software within the operating framework.

Criterion 4.5 — The error reporting forms for GENII 1.485 (see Figure 4-1) provided a formal method of problem mitigation. A similar methodology does not exist for GENII 2.0.

Criterion 4.10 — The hardware requirements for GENII 1.485 are an IBM PC/AT or compatible computer, an 80287 math coprocessor, 640 KB of random access memory, a minimum of 5 MB on-line disk storage, and operating under DOS 3.1 or later (Napier, 1988b). Hardware requirements for GENII 2.0 are Windows® 95, 98, NT, or 2000[2], using Pentium processors, and disk storage in excess of 60 MB. FRAMES and GENII make use of the memory swapping capabilities of Windows, so the programs should run on any Windows-compatible computer. However, they will generally run fastest on machines with 256Mbytes of memory or more (Napier, 2002a). GENII 2.0 will not run in the DOS environment.

Criterion 4.13 — GENII 1.485 has already been thoroughly reviewed and tested and there are no plans to pursue these issues again. GENII 2.0 has been reviewed at PNNL and several EPA clients, and it went through an advisory review with the EPA Science Advisory Board. This board suggested some additional capabilities that have not yet been implemented. The code author developed the code as general-purpose software and "importance to safety" was not an issue in its development. Standardization was an important consideration and was a direct response to the issue of testability and complexity of the older version. GENII 2.0 is very similar to 1.485 but it is not the same and is intended for a different set of users.

In summary, the GENII 1.485 User's Guide (Napier, 1988b), p 5.1, states: "The design process consisted of developing and internally testing software, developing test cases, and documenting software in accordance with the design input. The GENII package has been extensively tested and verified by hand, using the hand calculation worksheets of (the Code Maintenance Manual) and benchmarked against similar Hanford environmental dosimetry programs. A 10-volume set of test documentation is available for review from the authors upon request. The design process concluded with analysis of the final design by means of a Final Internal Development Review (FIDR). Two external peer reviews were held, as described in (the Conceptual Representation volume); these constitute the FIDR for the GENII package."

[2] The documentation from which this sentence was extracted (Napier, 2002a) was written before the advent of Windows XP. Experience shows that GENII 2.0 also runs under Windows XP.

4.4.2 Sources and Method of Review

All of the documentation listed in Table 1-2 has been reviewed with attention to "Design," except for Item 12 (see Appendix B), and several e-mail communications with the code developer (Bruce Napier) have helped to clarify issues.

4.4.3 Software Quality-Related Issues or Concerns

There are no additional SQA related issues or concerns in "Design."

4.4.4 Other Areas for Improvement

No other areas of improvement have been identified.

4.4.5 Recommendations

Recommendations related to this topical area are provided in Table 4.4-2.

Table 4.4-2 — Recommendations for Design Phase Topic

Recommendation Number	Relates to Table 4.4-1 Criterion Number(s)	Recommendation	Est. FTE to Complete	Est. Calendar Duration
4.1	4.5	See recommendation 2.1 on criterion 2.6.		
4.2	4.11, 4.12, 4.13, 4.14	When GENII 2.0 is complete, a comprehensive independent review must be documented to cover all aspects of these items	Two FTE months	Four months

Additional Detail

No additional detail is needed on the above recommendations.

4.5 Topical Area 5 Assessment: Implementation Phase

This area corresponds to the requirement entitled *Implementation Phase* in Table 3-3 of the DOE SQA plan (DOE 2003e).

4.5.1 Criterion Specification and Result

This topical area is "graded" for GENII 1.485 and "required" for GENII 2.0. Table 4.5-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
5.1	The implementation process resulted in software products such as computer program listings and instructions for computer program use.	Yes for 1.485. Partial for 2.0	1.485: Napier (1988c) is the code maintenance manual, containing listings of all source code. Napier (1988b) is the user's manual. 2.0: Napier (2002a) is the user's guide. Program listings are not yet published.
5.2	Implemented software was analyzed to identify and correct errors.	Yes for 1.485. Partial for 2.0.	1.485: An error reporting and corrective action process was used during development. 2.0: Used an informal error reporting process.
5.3	The source code finalized during verification (this phase) was placed under configuration control.	Yes for 1.485. No for 2.0.	1.485: Configuration control was in place during code development. Current configuration control is provided through RSICC, the distributor of the code, who will not release revised code unless tested and verified. 2.0: Code is not yet finalized.
5.4	Documentation during verification included a copy of the software, test case description, and associated criteria that are traceable to the software requirements and design documentation.	Yes for both	Although the documentation reviewed (Table 1-2) does not specifically address the items provided to the testers, the code author affirms that these items were given to them.

Additional Detail

The following provides additional detailed explanation on selected criteria in the above table:

Criterion 5.1 — GENII 2.0 has not been finalized. Code listings should become available after completion and final testing of code.

Criterion 5.2 — See recommendation 2.1 (on Criterion 2.6) for a discussion of this.

Criterion 5.3 — The appendix to Napier (1988a), the system design document, states: "Configuration control shall be a feature of the software to protect the basic code from unauthorized changes. A control mechanism with sign-off procedures shall be implemented to protect the software from unauthorized modifications. Needed changes shall be validated before modification are permitted." Bruce Napier is the current custodian of GENII 1.485 although at times past others had been assigned this duty. The code is distributed through RSICC at Oak Ridge, TN. Together, they provide the current configuration control.

Criterion 5.4 — The code author (Bruce Napier) states (e-mail in Appendix A): "The test cases were generally designed to meet the needs of certain types of calculation, and were done first on the computer (using the code and documentation to run) and then again on the GENII-specific

hand calculation worksheets. The criteria were that the numbers had to match to two significant figures (which is all that the GENII code transfers internally at certain steps)."

4.5.2 Sources and Method of Review

E-mails with the code author addressed some of these issues. In addition, all of the documentation listed in Table 1-2 was reviewed with attention to "Implementation," except for Item 12 (see Appendix B).

4.5.3 Software Quality-Related Issues or Concerns

There are no other SQA-related issues or concerns in "Implementation Phase."

4.5.4 Other Areas for Improvement

No other areas for improvement have been identified.

4.5.5 Recommendations

Recommendations related to this topical area are provide in Table 4.5-2.

Table 4.5-2 — Recommendations for Implementation Phase Topic

Recom-mendation Number	Relates to Table 4.5-1 Criterion Number(s)	Recommendation	Est. FTE to Complete	Est. Calendar Duration
5.1	5.1	Make GENII 2.0 code listings available upon completion and final testing of code.	One FTE week	One month

4.6 Topical Area 6 Assessment: Testing Phase

This area corresponds to the requirement entitled *Testing Phase* in Table 3-3 of the DOE SQA plan (DOE 2003e).

4.6.1 Criterion Specification and Result

This topical area is "required" for both GENII 1.485 and 2.0. Table 4.6-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
6.1	The software was validated by executing test cases.	Yes for 1.485. No for 2.0.	1.485: Code was validated by being thoroughly tested (Napier, 1988a, 1988b). 2.0: Code not yet completed, so testing is not complete.
6.2	Testing demonstrated the capability of the software to produce valid results for test cases encompassing the range of permitted usage defined by the program documentation. Such activities provide evidence to ensure that the software adequately and correctly performed all intended functions and does not perform adverse unintended functions.	Yes for 1.485. No for 2.0.	1.485: Code was thoroughly tested (Napier, 1988a, 1988b). 2.0: Code not yet completed, so testing is not complete.
6.3	Testing demonstrated that the computer program properly handles abnormal conditions and events as well as credible failures appropriate warning or error messages are provided to the user when the code is used improperly (e.g., an input is specified outside acceptable range).	Yes for 1.485. No for 2.0.	1.485: Code was thoroughly tested (Napier, 1988a, 1988b). 2.0: Code not yet completed, so testing is not complete.
6.4	Test Phase documentation includes test procedures or plans and the results of the execution of test cases. The test results documentation demonstrates successful completion of all test cases or the resolution of unsuccessful test cases and provides direct traceability between the test results and specified software requirements.	Yes for 1.485. No for 2.0.	1.485: Code was thoroughly tested (Napier, 1988a, 1988b). 2.0: Code not yet completed, so testing is not complete.

Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results (continued)

Criterion Number	Criterion Specification	Compliant	Summary Remarks
6.5	Test procedures or plans specify the following, as applicable: (1) Required tests and test sequence, (2) Required range of input parameters (3) Identification of the stages at which testing is required, (4) Requirements for testing logic branches, (5) Requirements for hardware integration, (6) Anticipated output values (7) Acceptance criteria, (8) Reports, records, standard formatting, and conventions, (9) Identification of operating environment, support software, software tools or system software, hardware operating system(s) and/or limitations.	Yes for 1.485. No for 2.0.	1.485: Code was thoroughly tested (Napier, 1988a, 1988b). 2.0: Code not yet completed, so testing is not complete.

Additional Detail

The following provides additional detailed explanation on selected criteria in the above table:

Criteria 6.1 – 6.5 — Napier (1988b) states that there is a ten-volume set of test documentation available for inspection by interested parties. These documents are not included in those reviewed here, as they are at the offices at PNNL. The GENII 2.0 User's Guide (Napier, 2002a), in reference to Version 1.485, states: "GENII Version 1 has been included in the International Atomic Energy Agency's VAMP project (VAlidation of Model Predictions - an acronym for the Coordinated Research Program on Validation of Models for the Transfer of Radionuclides in Terrestrial, Urban and Aquatic Environments), an international effort to compare environmental radionuclide transport models with measured environmental data. Results for test scenario CB (based on environmental measurements following the Chernobyl accident) indicated that dose estimates from GENII were comparable to, although slightly higher than, those of other participating models, which is consistent with its primary function as a prospective analysis tool. The models included in the code have been validated to various degrees by additional studies, however these have not been compared directly to output from the code."

4.6.2 Sources and Method of Review

All of the documentation listed in Table 1-2 has been reviewed with attention to "Testing Phase," except for Item 12 (see Appendix B).

4.6.3 Software Quality-Related Issues or Concerns

There are no other SQA-related issues or concerns in "Testing Phase."

4.6.4 Other Areas for Improvement

No other areas of improvement in the "Testing Phase" have been identified.

4.6.5 Recommendations

Recommendations related to this topical area are provide in Table 4.6-2.

Table 4.6-2 — Recommendations for Testing Phase Topic

Recom-mendation Number	Relates to Table 4.6-1 Criterion Number(s)	Recommendation	Est. FTE to Complete	Est. Calendar Duration
6.1	All	Document all testing of GENII 2.0.	Three FTE months	Six months

4.7 Topical Area 7 Assessment: User Instructions

This area corresponds to the requirement entitled *User Instructions* in Table 3-3 of the DOE SQA plan (DOE 2003e).

4.7.1 Criterion Specification and Result

This topical area is "required" for both GENII 1.485 and 2.0. Table 4.7-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
7.1	A description of the model is documented and made available to users.	Yes for both	1.485: Napier, 1988a. 2.0: Napier, 2002b.
7.2	User's manual or guide describes software and hardware limitations and identifies/includes approved operating systems (for cases where source code is provided, applicable compilers should be noted).	Yes for both	1.485: Napier, 1988b. 2.0: Napier, 2002a. Lahey Fortran-77 or F-99 compiler used. Source code in: 1.485: Napier, 1988c. 2.0: Not provided.

Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results (continued)

Criterion Number	Criterion Specification	Compliant	Summary Remarks
7.3	User's manual or guide includes description of the user's interaction with the software.	Yes for both	1.485: Napier, 1988b. 2.0: Napier, 2002a and 2003.
7.4	User's manual or guide includes a description of any required training necessary to use the software.	Yes for 1.485. No for 2.0.	1.485: A required training course is described in the system requirements document, not the user's manual. 2.0: Training is available (e.g., at EFCOG meetings) but it is not described in the User's Manual.
7.5	User's manual or guide includes input and output specifications.	Yes for both	1.485: Napier, 1988b. 2.0: Napier, 2002a.
7.6	User's manual or guide includes a description of user messages initiated because of improper input and how the user can respond.	Yes for both	1.485: Napier, 1988b. 2.0: Napier, 2002a.
7.7	User's manual or guide includes information for obtaining user and maintenance support.	Yes for 1.485. Partial for 2.0.	1.485: Readme.93 file on Distribution Disk 03. 2.0: Napier, 2002a.

Additional Detail

The following provides additional detailed explanation on selected criteria in the above table:

Criterion 7.2 — Both versions of GENII were written and compiled using the Lahey Fortran (F-77 or F-99) software, except for the user interface of GENII 1.485 (Apprentice), which was written using Microsoft QuickBasic. Source code for GENII 1.485 is given in Volume 3 of PNL-6584, *Code Maintenance Manual* (Napier, 1988c). It is also can be found on Distribution Disk02 by double clicking on SOURCE.EXE, which will unpack all the routines, both those in Fortran and those in QuickBasic. Source code is not provided for GENII 2.0.

Criterion 7.4 — The appendix to Napier (1988a), the system requirements document, p A.15, states: "A short training program shall be developed at the completion of the code to instruct potential users on the execution of the code. A detailed stepwise instruction manual shall also be prepared. Training should consist of class sessions and hand-out instructions, with opportunity for hands-on testing of the code." This training was provided on GENII 1.485 after it was released but such training is no longer available. Training for GENII 2.0 has been available at annual EFCOG meetings but there is no guarantee this will continue. Training would be useful for GENII (either version). The intuitive nature of the user interface and the documentation (e.g., Napier, 1988b, 2002a, 2003) is helpful but not enough for a first-time user.

Criterion 7.6 — In GENII 1.485, user input is primarily through the Apprentice program, which prompts the user for input and requires incorrect or incompatible entries to be corrected. Appendix B of the GENII 1.485 User's Manual (Napier, 1988b) gives an extensive discussion of error handling within GENII, not just that of Apprentice. For GENII 2.0, the FRAMES user interface provides error messages when input is incomplete, out of bounds, or conflicting.

However, the current version has bugs. For example, it is possible to be trapped in an unending loop of error messages.

Criterion 7.7 -- The GENII 1.485 User's Manual gives the names of the authors of GENII but not the contact information. The primary contact person is the lead author of the code, Bruce Napier (509-375-3896). In addition, RSICC has provided a "Readme" file with the name and telephone number of a very knowledgeable user of the code (Paul D. Rittman - 509-376-8715), who can also be contacted in case of problems. For GENII 2.0, the FRAMES Constituent Database user interface gives the contact information for the lead author of GENII (Bruce Napier).

4.7.2 Sources and Method of Review

The user's manual for GENII 1.485, *GENII – The Hanford Environmental Radiation Dosimetry Software System. Volume 2: User's Manual* (Napier, 1988b), was reviewed for this Gap Analysis. Section 2 of that document gives the code overview, including user interaction levels and data file descriptions. Section 3 gives specific user instructions for both user interaction levels 0 and 1. Section 4 discusses system requirements and Section 5 discusses quality assurance topics. Appendix A gives an input/output example and Appendix B gives an extensive discussion of error messages. A revision to some of the data files for GENII 1.485 was issued in 1993 and another in 1996, but these did not change the code or its usage.

The User's Guide for GENII 2.0, *GENII Version 2 User's Guide* (Napier, 2002a) and *Getting Started with GENII Version* 2 (Napier, 2003) were reviewed for this Gap Analysis. The User's Guide provides details on all the options available in GENII 2.0, whereas the Getting Started document provides an introduction useful for evaluating simple, but typical, scenarios.

Correspondence (e-mails and telephone conversations) with an expert user of GENII 2.0 and with Bruce Napier has also been reviewed. A condensed version of them is included as Appendix A of this document. The expert user of GENII 2.0 was identified by Bruce Napier as William Joyce[3].

4.7.3 Software Quality-Related Issues or Concerns

An item not discussed in the documentation is memory management. GENII 1.485 was developed in the DOS environment and was expected to be run in that environment. Experience shows that it can be run in a DOS window in the Windows environment[4] but problems may be encountered when runs are made on Windows-XP based computer. This may stem from the fact that memory management is different between DOS and Windows. GENII 1.485 should be recompiled with a Windows-XP compatible FORTRAN compiler and verified to run properly.

The bug in error handling of GENII 2.0 (see Criterion 7.6) needs to be fixed.

---

[3] Mr. Joyce is a Senior Safety Engineer with ATL International, Corp., 20010 Century Blvd, Suite 500, Germantown, MD 20874.

[4] The Radiation Safety Information Computational Center (RSICC) at Oak Ridge verified the performance of GENII 1.485 on a 486 PC under the MS DOS 6.2 and Windows 95 operating systems. Testing conducted during the preparation of this Gap Analysis shows that GENII 1.485 also can be executed in Windows 98SE and but problems were encountered when run under Windows-XP.

4.7.4 Other Areas for Improvement

The GENII 2.0 user guidance (Napier, 2002b, 2003) doesn't always match the operations the user needs to perform. For example, in a number of cases, the instructions say to right-click a button whereas the correct procedure is a left-click. In addition, some of the screens the user sees are not in the same order given in the guidance.

GENII 1.485 can determine 95th percentile consequences in only one direction (sector) at a time. It would be very helpful to the analyst for GENII 1.485 to automatically determine the 95th percentile consequences in every sector at the site boundary and other user-selected distance (such as 100 m). This can be done now only by setting up multiple runs of GENII 1.485. GENII 2.0 cannot determine 95th percentile consequences except perhaps in a manner involving a random sampling of the weather and compiling statistics that would yield 95th percentile values. However, this has not yet been tested.

4.7.5 Recommendations

Recommendations related to this topical area are provide in Table 4.7-2.

Table 4.7-2 — Recommendations for User Instructions Topic

Recommendation Number	Relates to Table 4.7-1 Criterion Number(s)	Recommendation	Est. FTE to Complete	Est. Calendar Duration
7.1	Criterion 7.2	Recompile GENII 1.485 with a Windows-XP compatible compiler and verify that it runs correctly in a Windows environment.	One FTE week	Two weeks
7.2	Criterion 7.5	Correct the user guidance for GENII 2.0.	One FTE week	Two weeks
7.3	Criterion 7.6	The error message-handling problem needs to be fixed.	One FTE week	Two weeks

Additional Detail

Recommendation 7.1 – The estimate of one FTE week is for recompiling and the comparison testing, which would consist of running the same scenarios side by side on DOS-based and Window-based computers. The current version of GENII 1.485 should not be run on Windows-XP based computers.

4.8 Topical Area 8 Assessment: Acceptance Test

This area corresponds to the requirement entitled *Acceptance Test* Table 3-3 of the DOE SQA plan (DOE 2003e). During this phase of the software development, the software becomes part of a system incorporating applicable software components, hardware, and data, and is accepted for use.

During development of the software, the developing organization is responsible for documenting its procedures and acceptance tests it uses. Once the software is released, user organizations need a test

protocol to determine if the software is correctly installed. Implementation for this type of acceptance testing is the responsibility of the user organization.

Criterion Specification and Result

This topical area is "required" for both GENII 1.485 and 2.0. Table 4.8-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
8.1	To the extent applicable to the developer, acceptance testing includes a comprehensive test in the operating environment(s).	Yes for 1.485. No for 2.0.	1.485: Napier (1988b) states that the code was tested on PCs from many manufacturers. 2.0: Acceptance testing is not yet complete but Napier (2002a) states but the test plan has been developed and testing underway.
8.2	To the extent applicable to the developer, acceptance testing was performed prior to approval of the computer program for use.	Yes for 1.485. No for 2.0.	1.485: The code delivered to RSICC for distribution had been tested prior to release. 2.0: Acceptance testing is not yet complete.
8.3	The acceptance testing comprehensively evaluates software performance against specified software requirements. To the extent applicable to the developer, software validation was performed to ensure that the installed software product satisfies the specified software requirements.	Yes for 1.485. No for 2.0.	Both codes were developed under NQA-1 guidelines. This includes testing against software requirements. 1.485: Acceptance testing complete and code in use. 2.0: Acceptance testing is not yet complete.
8.4	Acceptance testing documentation includes results of the execution of test cases for system installation and integration, user instructions (Refer to Requirement 7 above), and documentation of the acceptance of the software for operational use.	Yes for 1.485. No for 2.0.	1.485: Extensive test documentation is available on all aspects of code development. 2.0: Acceptance testing is not yet complete.

Additional Detail

The following provides additional detailed explanation on selected criteria in the above table:

> Criterion 8.1 -- The GENII 1.485 User's Manual (Napier, 1988b), p 4.1, states: "Portions of the GENII Software Package have been tested on a number of IBM-PC/AT compatible machines. Versions of GENII have been established on microcomputers manufactured by GRID, NEC, Hewlett-Packard, and IBM. The IBM machines have included the new PS/2 System 50 and System 80. No machine-based incompatibilities have been found." The GENII 2.0 User Guide (Napier, 2002a), p 6, states: "A comprehensive test plan has been developed and testing is underway."

Criterion 8.2 — The preface to the RSICC distribution package of GENII 1.485 states that the authors of the code affirm that the code was tested prior to submission to RSICC for distribution to users.

Criterion 8.3 — The GENII 2.0 User Guide (Napier, 2002a), pp 5-6 states: "Both GENII versions were developed under QA plans based on the American National Standards Institute (ANSI) standard NQA-1 as implemented in the PNNL Quality Assurance Manual. All steps of the code development have been documented and tested, and hand calculations have verified the code's implementation of major transport and exposure pathways for a subset of the radionuclide library. A collection of hand calculations and other verification activities is available. A comprehensive test plan has been developed and testing is underway." The latter sentence refers to GENII 2.0, not 1.485.

Criterion 8.4 — Napier (1988b) states that there is a ten-volume set of test documentation available for inspection by interested parties.

4.8.1 Sources and Method of Review

All of the documentation listed in Table 1-2 has been reviewed with attention to "Acceptance Test," except for Item 12 (see Appendix B). The list in Appendix B includes a summary of developer/user testing and peer review of GENII for which documentation is available.

4.8.2 Software Quality-Related Issues or Concerns

There are no other SQA-related issues or concerns in "Acceptance Test."

4.8.3 Other Areas for Improvement

No other areas of improvement have been identified.

4.8.4 Recommendations

Recommendations related to this topical area are provide in Table 4.8-2.

Table 4.8-2 — Recommendations for Acceptance Test Topic

Recommendation Number	Relates to Table 4.8-1 Criterion Number(s)	Recommendation	Est. FTE to Complete	Est. Calendar Duration
8.1	All	Complete the documentation of acceptance testing for GENII 2.0	Two FTE months	Four months

4.9 Topical Area 9 Assessment: Configuration Control

This area corresponds to the requirement entitled *Configuration Control* in Table 3-3 of the DOE SQA plan (DOE 2003e).

4.9.1 Criterion Specification and Result

This topical area is "required" for both GENII 1.485 and 2.0. Table 4.9-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
9.1	For the developers, the methods used to control, uniquely identify, describe, and document the configuration of each version or update of a computer program (for example, source, object, and back-up files) and its related documentation (for example, software design requirements, instructions for computer program use, test plans, and results) are described in implementing procedures.	Yes for both	1.485: Configuration control followed PNO-MA-70, the PNL version of the NQA-1 Quality Assurance Manual that existed during development. In addition, a series of "software change packets" have been maintained. 2.0: Formal procedures for configuration control follow the current PNNL "Software Based Management System" (SBMS). Notebooks and backups are also used for this purpose. (See Appendix A.)
9.2	Implementing procedures meet applicable criteria for configuration identification, change control, and configuration status accounting.	Yes for both	See the comments above, for Criterion 9.1.

Additional Detail

The following provides additional detailed explanation on selected criteria in the above table:

> Criteria 9.1 and 9.2 — Configuration control followed/follows procedures formalized in SQA methods used at PNL/PNNL during the development of each version of GENII. These

procedures have evolved over the years, and thus, the procedures used for Version 2.0 are not identical to those used for Version 1.485. The author of the code(s) has kept informal notebooks and copies of earlier versions.

4.9.2 Sources and Method of Review

All of the documentation listed in Table 1-2 has been reviewed with attention to "Configuration Control," except for Item 12 (see Appendix B), as well as e-mails with the code developer.

4.9.3 Software Quality-Related Issues or Concerns

There are no SQA-related issues or concerns in "Configuration Control."

4.9.4 Other Areas for Improvement

No additional areas of improvement in "Configuration Control" have been identified.

4.9.5 Recommendations

There are no recommendations related to this topical area.

4.10 Topical Area 10 Assessment: Error Impact

This area corresponds to the requirement entitled *Error Impact* in Table 3-3 of the DOE SQA plan (DOE 2003e).

4.10.1 Criterion Specification and Result

This topical area is "graded" for both GENII 1.485 and 2.0. Table 4.10-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
10.1	The developing organization's problem reporting and corrective action process addresses the appropriate requirements of its corrective action system and is documented in implementing procedures.	Yes for 1.485. No for 2.0	Napier (1988b) discusses how to report errors and request upgrades. An informal method is used for GENII 2.0. See criterion 2.6.

Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results (continued)

Criterion Number	Criterion Specification	Compliant	Summary Remarks
10.2	The process for evaluating, and documenting whether a reported problem is an error is documented and implemented.	No for both	Not specifically discussed in the documentation reviewed. However, the SQA procedures followed during development (see criterion 9.1) do require problem reporting and documenting.
10.3	The process for disposition of the problem reports, including notification to the originator of the results of the evaluation, is documented and implemented.	No for both	Not specifically discussed in the documentation reviewed. However, the SQA procedures followed during development (see Criterion 9.1) do require proper disposition of problem reports.
10.4	A documented process provides guidance on determining how identified errors relate to appropriate software engineering elements and is implemented.	No for both	Not discussed in the documentation reviewed.
10.5	The process is documented and implemented for determining how an error impacts past and present use of the computer program.	No for both	Not discussed in the documentation reviewed.
10.6	The process is documented and implemented for determining how an error and resulting corrective action impacts previous development activities.	No for both	Not discussed in the documentation reviewed.
10.7	The process is documented and implemented describing how the users are notified of an identified error, its impact; and how to avoid the error, pending implementation of corrective actions.	No for both	Not discussed in the documentation reviewed.

4.10.2 Sources and Method of Review

All of the documentation listed in Table 1-2 has been reviewed with attention to "Error Impact," except for Item 12 (see Appendix B).

4.10.3 Software Quality-Related Issues or Concerns

For users of GENII 2.0 within PNNL, the existing Standards Based Management System (SBMS) process can be followed. There would be no software quality-related issues or concerns for these users. However, for users outside of PNNL, the process of error notification and corrective action needs to be formalized and documented so that users know how to report errors, how PNNL will respond, how PNNL will notify other users of the problem, and how too avoid the problem.

4.10.4 Other Areas for Improvement

No other areas of improvement are noted.

4.10.5 Recommendations

Recommendations related to this topical area are provide in Table 4.10-2.

Table 4.10-2 — Recommendations for Error Impact Topic

Recommendation Number	Relate to Table 4.13-1 Criterion Number(s)	Recommendation	Est. FTE to Complete	Est. Calendar Duration
10.1	All	A formal error reporting and corrective action process needs to be implemented for GENII 1.485 and GENII 2.0 for users outside of PNNL.	One FTE month	Two months

4.11 Training Program Assessment

No regularly scheduled GENII training program is conducted. Training materials for Version 1.485 of GENII are still available, but there have been no requests made to the author (Bruce Napier) to use these for several years.

There have been discussions with the EPA about training on Version 2, and the author has given some Version 2.0 training at recent EPA NESHAPS meetings (held annually). Future training may be provided to the NRC headquarters staff. However, the latter is still in the planning stage.

The last known training to DOE safety analysis community occurred during the 2000 Energy Facility Contractors Group (EFCOG) Safety Analysis Working Group Workshop (April 2000). It is recommended that this forum be explored to provide DOE users with a regular opportunity for GENII training.

5.0 Conclusion

The gap analysis for Version 1.485 and 2.0 of the GENII software, based on a set of requirements and criteria compliant with NQA-1, has been completed. Of the ten primary SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), nine requirements are met at an acceptable level for GENII 1.485, (items 1-9). Improvement actions are recommended for GENII 1.485 to fully meet the requirement for *Error Impact* (item 10). For GENII 2.0, of the ten primary SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), two requirements are met at an acceptable level, i.e., *Software Classification* (1) and *Configuration Control* (9). Improvement actions are recommended for GENII 2.0 to fully meet the requirement for five that are partially met, i.e., *SQA Procedures and Plans* (2), *Requirements Phase* (3), *Design Phase* (4), *Implementation Phase* (5), and *User Instructions* (7) and for the remaining three, *Testing Phase* (6), *Acceptance Test* (8), and *Error Impact* (10). This evaluation outcome is deemed acceptable because: (1) GENII 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 GENII is limited to those analytic applications for which the software is intended.

It was determined that GENII 1.485 code does meet its intended function for use in supporting documented safety analysis, providing it is not used on a Windows-XP based computer. It was determined that GENII 2.0 will not meet its intended function. Therefore, only GENII 1.485 can be recommended for DSA use at this time. As with all safety-related software, users should be aware of current limitations and capabilities of GENII for supporting safety analysis. Informed use of the software can be assisted by the current set of GENII reports (refer to Table 1-2), and the code guidance report for DOE safety analysts, *GENII Computer Code Application Guidance for Documented Safety Analysis*, (DOE, 2004). Furthermore, while SQA improvement actions are recommended for GENII, no evidence has been found of programming, logic, or other types of software errors in GENII that have led to non-conservatisms in nuclear facility operations, or in the identification of facility controls.

By order of priority, it is recommended that GENII software improvement actions be taken, especially:

- correcting know defects
- upgrading user technical support activities
- providing training on a regular basis, and
- revising software documentation.

Performing these four primary actions should satisfactorily improve the SQA compliance status of GENII relative to the primary evaluation criteria cited in this report.

Recommendations are given for each of the topical areas in Section 4.0. It is estimated that nearly ten full-time equivalent (FTE) months would be required to perform all SQA upgrade tasks covered in Section 4.0 for GENII 2.0. Because GENII 1.485 has been in use for many years and the code author does not intend to make any further modifications, no similar estimates need be made. The error-reporting estimate for GENII 2.0 may be applied to GENII 1.485. In order to use GENII 1.485 in all Windows environments, it will be necessary to recompile the code using a Windows-XP compatible compiler. A side-by-side testing on DOS-based and Windows-based computers would then follow this. The GENII 1.485 documentation would not need to be changed if the results were the same but documentation of the results should be included with the RSICC distribution package for GENII 1.485. The recompiled version would have to be given a new number, such as 1.486.

Training opportunities exist for both versions of GENII, but these are not routinely offered. It is recommended that user training for safety analysis applications be conducted formally on at a minimum, an annual basis. Prerequisites for, and core knowledge needed by, the user prior to initiating GENII applications should be documented by the code developer.

While completion of the GENII 2.0 development is encouraged, current DOE DSA support should be through the earlier code version, GENII 1.485. Use of Windows-XP based computers should be avoided for GENII 1.485 until such time that a Windows-XP based version is available.

6.0 Acronyms and Definitions

ACRONYMS:

ALOHA	Areal Locations of Hazardous Atmospheres (designated toolbox software)
ANS	American Nuclear Society
ANSI	American National Standards Institute
ASME	American Society of Mechanical Engineers
CD	Compliance Decision
CFAST	Consolidated Fire and Smoke Transport Model (designated toolbox software)
CFR	Code of Federal Regulations
DNFSB	Defense Nuclear Facilities Safety Board
DoD	Department of Defense
DOE	Department of Energy
DSA	Documented Safety Analysis
EFCOG	Energy Facility Contractors Group
EH	DOE Office of Environment, Safety and Health
EIS	Environmental Impact Statement
EPA	Environmental Protection Agency
EPIcode	Emergency Prediction Information code (designated toolbox software)
FTE	Full-time equivalent
GENII	Generalized Environmental Radiation Dosimetry Software System - Hanford Dosimetry System (Generation II) (designated toolbox software)
IEEE	Institute of Electrical and Electronics Engineers
IP	Implementation Plan
ISO	International Standards Organization
MACCS2	MELCOR Accident Consequence Code System 2 (designated toolbox software)
MELCOR	Methods for Estimation of Leakages and Consequences of Releases (designated toolbox software)
NESHAPS	National Emission Standards for Hazardous Air Pollutants
NNSA	National Nuclear Security Administration
NRC	Nuclear Regulatory Commission
QAP	Quality Assurance Program (alternatively, Plan)
RSICC	Radiation Safety Information Computational Center
SNL	Sandia National Laboratories
SQA	Software Quality Assurance
V&V	Verification and Validation

DEFINITIONS

The following definitions are taken from the Implementation Plan. References in brackets following definitions indicate the original source, not the Implementation Plan.

Acceptance Testing — The process of exercising or evaluating a system or system component by manual or automated means to ensure that it satisfies the specified requirements and to identify differences between expected and actual results in the operating environment. [NQA-1]

Central Registry — An organization designated to be responsible for the storage, control, and long-term maintenance of the Department's safety analysis "toolbox codes." The central registry may also perform this function for other codes if the Department determines that this is appropriate.

Computer Code — A set of instructions that can be interpreted and acted upon by a programmable digital computer (also referred to as a module or a computer program).

Dedication (of Software) — The evaluation of software not developed under utilizing organization existing quality assurance plans and procedures (or not developed under NQA-1 standards). The evaluation determines and asserts the software's compliance with NQA-1 quality standards and its readiness for use in specific applications. (Typically applies to commercially available software.) The utilizing organization reviews the intended software application sufficiently to determine the critical functions that provide evidence of the software's suitability for use. Once the critical functions have been established, methods are defined to verify critical function adequacy and provide verifiable acceptance criteria. Acceptable dedication methods are implemented and required documentation is prepared.

Design Requirements — Description of the methodology, assumptions, functional requirements, and technical requirements for a software system.

Error — A condition deviating from an established base line, including deviations from the current approved computer program and its baseline requirements. [NQA-1]

Executable Code — The user form of a computer code. For programs written in a compilable programming language, the compiled and loaded program. For programs written in an interpretable programming language, the source code.

Firmware — The combination of a hardware device and computer instructions and data that reside as read-only software on that device. [IEEE Standard 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology]

Gap Analysis — Evaluation of the Software Quality Assurance attributes of specific computer software against identified criteria.

Nuclear Facility — A reactor or a nonreactor nuclear facility where an activity is conducted for, or on behalf of, DOE and includes any related area, structure, facility, or activity to the extent necessary to ensure proper implementation of the requirements established by 10 CFR 830. [10 CFR 830]

Object Code — A computer code in its compiled form. This applies only to programs written in a compilable programming language.

Operating Environment — A collection of software, firmware, and hardware elements that provide for the execution of computer programs. [NQA-1]

Safety Analysis and Design Software — Computer software that is not part of a structure, system, or component (SSC) but is used in the safety classification, design, and analysis of nuclear facilities to: ensure the proper accident analysis of nuclear facilities; ensure the proper analysis and design of safety SSCs; and, ensure the proper identification, maintenance, and operation of safety SSCs. [DOE O 414.1B]

Safety Analysis Software Group (SASG) — A group of technical experts formed by the Deputy Secretary in October 2000 in response to Technical Report 25 issued by the Defense Nuclear Facilities Safety Board (DNFSB). This group was responsible for determining the safety analysis and instrument and control (I&C) software needs to be fixed or replaced, establishing plans and cost estimates for remedial work, providing recommendations for permanent storage of the software and coordinating with the Nuclear Regulatory Commission on code assessment as appropriate.

Safety Software — Includes both safety system software, and safety analysis and design software. [DOE O 414.1B]

Safety System Software — Computer software and firmware that performs a safety system function as part of a structure, system, or component (SSC) that has been functionally classified as Safety Class (SC) or Safety Significant (SS). This also includes computer software such as human-machine interface software, network interface software, programmable logic controller (PLC) programming language software, and safety management databases that are not part of an SSC but whose operation or malfunction can directly affect SS and SC SSC function. [DOE O 414.1B]

Software —Computer programs, operating systems, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. [IEEE Standard 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology

Software design requirements The activities that begin with the decision to develop a software product and end when the software is delivered. The software development cycle typically includes the following activities:

Software design
Implementation
Test, and sometimes:
Installation. [NQA-1]

Software Engineering — The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software; also: the study of these applications. [NQA-1]

Source Code — A computer code in its originally coded form, typically in text file format. For programs written in a compilable programming language, the uncompiled program.

System Software — Software designed to enable the operation and maintenance of a computer system and its associated computer programs. [NQA-1]

Test Plan (Procedure) — A document that describes the approach to be followed for testing a system or component. Typical contents identify the items to be tested, tasks to be performed, and responsibilities for the testing activities. [NQA-1]

Testing — An element of verification for the determination of the capability of an item to meet specified requirements by subjecting the item to a set of physical, chemical, environmental, or operating conditions. [NQA-1]

Toolbox Codes — A small number of standard computer models (codes) supporting DOE safety analysis, having widespread use, and of appropriate qualification that are maintained, managed, and distributed by a central source. Toolbox codes meet minimum quality assurance criteria. They may be applied to support 10 CFR 830 DSAs provided the application domain and input parameters are valid. In addition to public domain software, commercial or proprietary software may also be considered. In addition to safety analysis software, design codes may also be included if there is a benefit to maintain centralized control of the codes [modified from DOE N 411.1].

User Manual — A document that presents the information necessary to employ a system or component to obtain desired results. Typically described are system or component capabilities, limitations, options, permitted inputs, expected outputs, possible error messages, and special instructions. Note: A user manual is distinguished from an operator manual when a distinction is made between those who operate a computer system (mounting tapes, etc.) and those who use the system for its intended purpose. Syn: User Guide. [IEEE 610-12]

Validation — 1. The process of testing a computer program and evaluating the results to ensure compliance with specified requirements. [ANSI/ANS-10.4-1987]
2. The process of determining the degree to which a model is an accurate representation of the real-world from the perspective of the intended uses of the model. [Department of Defense Directive 5000.59, DoD Modeling and Simulation (M&S) Management]

Verification —1. The process of evaluating the products of a software development phase to provide assurance that they meet the requirements defined for them by the previous phase. [ANSI/ANS-10.4-1987]
2. The process of determining that a model implementation accurately represents the developer's conceptual description and specifications. [Department of Defense Directive 5000.59, DoD Modeling and Simulation (M&S) Management]

7.0 References

Note: The references listed below may not have been used directly in the gap analysis. However, they were used to provide a context for performing the overall code evaluation.

CFR Code of Federal Regulations (10 CFR 830). 10 CFR 830, Nuclear Safety Management Rule.

DNFSB Defense Nuclear Facilities Safety Board, (2000). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, Technical Report DNFSB/TECH-25, (January 2000).

DNFSB Defense Nuclear Facilities Safety Board, (2002). *Recommendation 2002-1, Quality Assurance for Safety-Related Software,* (September 2002).

DOE, U.S. Department of Energy (2000a). *Appendix A, Evaluation Guideline,* DOE-STD-3009-94, *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports* (January 2000).

DOE, U.S. Department of Energy (2000b). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, DOE Response to TECH-25, Letter and Report, (October 2000).

DOE, U.S. Department of Energy (2002). *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports*, DOE-HDBK-3010-94, Change Notice 2 (April 2002).

DOE, U.S. Department of Energy (2003a). *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).

DOE, U.S. Department of Energy (2003b). *Designation of Initial Safety Analysis Toolbox Codes*, Letter, (March 28, 2003).

DOE, U.S. Department of Energy (2003c). *Assessment Criteria and Guidelines for Determining the Adequacy of Software Used in the Safety Analysis and Design of Defense Nuclear Facilities*, Report, CRAD-4.2.4-1, Rev 0, (August 27 2003).

DOE, U.S. Department of Energy (2003d). *Software Quality Assurance Improvement Plan: Format and Content For Code Guidance Reports*, Revision A (draft), Report, (August 2003).

DOE, U.S. Department of Energy (2003e). *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes*, Revision 1, (November 2003).

DOE, U.S. Department of Energy (2004). *GENII Computer Code Application Guidance for Documented Safety Analysis*, (May 2004).

Napier, 1988a, B. A. Napier, R. A. Peloquin, D. L. Strenge, and J. V. Ramsdell, *GENII – The Hanford Environmental Radiation Dosimetry Software System. Volume 1: Conceptual Representation.* PNL-6584, Pacific Northwest Laboratories, Richland, WA, (December 1988).

Napier, 1988b, B. A. Napier, R. A. Peloquin, D. L. Strenge, and J. V. Ramsdell, *GENII – The Hanford Environmental Radiation Dosimetry Software System. Volume 2: User's Manual*, PNL-6584, Pacific Northwest Laboratories, Richland, WA, (November 1988).

Napier, 1988c, B. A. Napier, R. A. Peloquin, D. L. Strenge, and J. V. Ramsdell, *GENII – The Hanford Environmental Radiation Dosimetry Software System. Volume 3: Code Maintenance Manual*, PNL-6584, Pacific Northwest Laboratories, Richland, WA, (September 1988).

Napier, 1995, B. A. Napier, J. V. Ramsdell, and D. L. Strenge. *Software Requirements Specifications for Hanford Environmental Dosimetry Coordination Project*, May 1995 Draft Report, prepared for review by the EPA Office of Radiation and Indoor Air.

Napier, 1999a, B. A. Napier, *GENII Version 2 Example Calculation Descriptions*, Prepared for U.S. Environmental Protection Agency, January 1999.

Napier, 1999b, B. A. Napier and L. Staven, *GENII Version 2 Training Power Point Slides*, Presented at the Safety Analysis Workshop of the annual meeting of the Energy Facility Contractors Group (EFCOG), June 1999.

Napier, 2002a, B. A. Napier, *GENII Version 2 User's Guide*, Prepared for U.S. Environmental Protection Agency, September 2002.

Napier, 2002b, B. A. Napier, D.L. Strenge, J. V. Ramsdell, Jr., P. W. Eslinger, and C. Fosmire, *GENII Version 2 Software Design Document*, Prepared for U.S. Environmental Protection Agency, November 2002.

Napier, 2003, B. A. Napier, *Getting Started with GENII Version 2*, Prepared for U.S. Environmental Protection Agency, February 2003.

Napier, 2003, B. A. Napier, Communication with K.R. O'Kula.

Appendices

Appendix	Subject
A	COMMUNICATIONS WITH OTHERS
B	GENII BENCHMARKING AND V&V

APPENDIX A.— COMMUNICATIONS WITH OTHERS

Appendix A is provided to supplement and support information found in GENII documentation. Bruce Napier is the code developer and William Joyce is an EPA consultant that has been reviewing GENII 2.0.

1) Differences between Versions 1.485 and 2.0

Response from Napier:
Version 2 is much different than 1.485:

- It uses hourly meteorology, not joint frequency data.
- It is set up for the acute release met model to start at a defined date and time. However, the FRAMES system has a stochastic processor that wraps around all the GENII modules and allows variation in all the input parameters.
- It can be run a few thousand times, varying the start time. This has the effect of building the entire output dose distribution, not just the 95th percentile meteorology.
- Seasonal model taken out only, uses the Fall model.

2) Topics regarding SQA

Response from Napier:

GENII 1.485

This version was developed under the earliest NQA-1 standards (1986 version):

SQA Plan — Exists, out of date. Refers to PNNL manual no longer available. Napier has key chapters though.

Software Requirements Document — Exists, but the one developed was very short, and not nearly as detailed as may now be desired.

Software Design Document —Believe the GENII PNL-6854 Volume I report covers this.

Test Case Description and Report — Ran all modifications against a series of regression tests with known answers. Have an extensive series of documented hand calculation worksheets that give "the right answer." Some documentation is available not in reports.

Software Configuration and Control Document— Hard copies of all the versions from 1.350 (the point at which things were stable) through 1.485, including the software change packets exist. RSICC does code distributions.

Error Notification and Corrective Action Report — Not done now except in extraordinary circumstances.

User's Manual, and other relevant documentation (model description, weekly or monthly reports to code sponsor, etc.). — Believe GENII PNL-6854 Volume 2 report covers this.

No official changes to the code since 1990. Because of compiler and other issues, changes may be difficult now.

GENII V2

GENII Version 2 keeps the name, and a few of the basic algorithms. Almost everything else is new.

The formal QA is weaker than for 1.485.

SQA Plan — Exists, but is very brief.

Software Requirements Document — Exists, reasonably detailed and complete.

Software Design Document —GENII Version 2 Software Design Document available.

Test Case Description and Report — None exists.

Software Configuration and Control Document —Informal notebooks and backups exist.

Error Notification and Corrective Action Report — Limited beta testing is done.

User's Manual, and other relevant documentation (model description, weekly or monthly reports to code sponsor, etc.). — GENII Version 2 Users Guide is available, plus the "Getting Started with GENII" exists.

Was reviewed by the EPA Science Advisory Board (who have a report).

3) Verification Documentation – copy of the software available to testers, test case description available, associated criteria available, software requirements traceability.

Response from Napier:

The test cases were generally designed to meet the needs of certain types of calculation, and were done first on the computer (using the code and documentation to run) and then again on the GENII-specific hand calculation worksheets. The criteria were that the numbers had to match to 2 significant figures (which is all that the GENII code transfers internally at certain steps).

- Verifiers had the software.
- Verifiers had the documentation. The GENII documentation, PNL-6584 Volume 1 contains the Design Requirements as an appendix. Requirements are traceable.
- Verifiers had test case descriptions (or wrote their own).
- Verifiers had criteria.

4) General code use and information

Response from William Joyce:

- GENII 2.0 can't give 95th percentile consequences
- The ten receptor locations in GENII 2.0 are each forced to be at the nearest grid points, which may not be where the user wants them
- GENII 2.0 is meant for EPA NESHAPS *(was not developed specifically for DSA applications)*
- GENII 1.485 was developed in a DOS environment and therefore had to address the memory limit of <640 KB. The Windows memory management system is different and there is a potential that this may lead to problems.

APPENDIX B. — GENII BENCHMARKING AND V&V

(List provided by Bruce Napier)

Publications on GENII Verification and Validation

Johnson, K.A., and M.J. Sowa. 1997. Benchmarking the GENII and RESRAD Computer Codes, Oregon State University Radiation Center, Corvallis, Oregon.

International Atomic Energy Agency. 1995. Validation of Models using Chernobyl Fallout Data from the Central Bohemia Region of the Czech Republic: Scenario CB, IAEA-TECDOC-795, First Report of the VAMP Multiple Pathways Assessment Working Group, International Atomic Energy Agency, Vienna, Austria.

Maheras, S.J. 1995. *GENII Version 1.485* (Software Review), Health Physics, 68, pp. 119-121.

Rittmann, P.D. 1995. Benchmarking of Computer Codes (GENII, PATHRAE, RESRAD) Using Hand Calculations, Westinghouse Hanford Company, Richland, Washington.

Maheras, S.J., P.D. Ritter, P.R. Leonard, and R. Moore. *Benchmarking of the CAP-88 and GENII Computer Codes using 1990 and 1991 Monitored Atmospheric Releases from the Idaho National Engineering Laboratory*, Health Physics, 67, pp. 509-517.

Faillace, E.R., J.J. Cheng, and C. Yu. 1994. RESRAD Benchmarking Against Six Radiation Exposure Pathway Models, ANL/EAD/TM-24, Argonne National Laboratory, Argonne, Illinois.

Preece, A.B. 1993. Use of the GENII Computer Code in a Low-Level Radioactive Waste Disposal Facility Performance Assessment Methodology. Master's Thesis, University of Texas, Austin, Texas.

Seitz, R.R., J.R. Cook, M.I. Wood, P.D. Rittmann, B.A. Napier, and D.W. Wood. 1992. Comparison of Computer Codes and Inputs Used at DOE Sites to Model Intrusion Scenarios, PNL-SA-20502, Pacific Northwest Laboratory. Presented at Waste Management '92, Tucson, Arizona, March 1-5, 1992

Kozak, M.W., M.S.Y. Chu, and P.A. Mattingly. 1990. A Performance Assessment Methodology for Low-Level Waste Facilities, NUREG/CR-5532, Sandia National Laboratories, Albuquerque, New Mexico.

Kozak, M.W., M.S.Y. Chu, P.A. Mattingly, J.D. Johnson, and J.T. McCord. 1990. Background Information for the Development of a Low-Level Waste Performance Assessment Methodology: Identification and Recommendation of Computer Codes, NUREG/CR-5453, Volume 5, Sandia National Laboratories, Albuquerque, New Mexico.

Jaquish, R. E., and B. A. Napier. 1987. "A Comparison of Environmental Radionuclide Concentrations Calculated by a Mathematical Model with Measured Concentrations." PNL-SA-14720. In Proceedings of ANS Topical Conference on Population Exposure from the Nuclear Fuel Cycle. Oak Ridge, Tennessee.

Aaberg, R. L., and B. A. Napier. 1985. Hanford Dose Overview Program: Comparison of AIRDOS-EPA and Hanford Site Dose Codes, PNL-5633, Pacific Northwest Laboratory, Richland, Washington.

Additional GENII Benchmarking and Comparisons

Stull, E. 1990. Comparison of GENII and RSAC-4 for use in the New Production Reactor EIS program.

Ikenberry, T.A. 1990. Demonstration of acceptable accuracy and reproducibility of the HUDU atmospheric dispersion and radiation dose program (benchmark against GENII).

Peterson, V., R. Patlovany, and G. Ennis. 1992. Comparison of MACCS and GENII, EG&G Rocky Flats, Boulder, Colorado.

Sartori, E., A. Curti, L. Riposi, and G. Graziani. 1992. Comparison of GENII and VADOSCA Computer Codes. Nuclear Energy Agency, Commission of the European Communities.

Abbott, M. 1993. MACCS2 benchmarking against GENII, Idaho National Engineering Laboratory, Idaho Falls, Idaho.

Aaberg, R.L. 1993. Comparison of GENII and RSAC-5 for the Tank Waste Remediation System (TWRS). Pacific Northwest National Laboratory, Richland, Washington.

Morris, J., and C. Williams. 1994. Workshop to discuss the use of environmental transport and fate models in the Environmental Restoration part of the DOE Programmatic Environmental Impact Statement (PEIS).

Peer Review of Multimedia Models. 1994. Comparison of MEPAS, MMSOIL, RESRAD, and GENII. M. Small (Carnegie-Mellon), D. Back (HydroGeologic), R. Charbeneau (U. Texas), C. Chein (duPont), Y. Cohen (U. California), T. Gallagher (HydroQual), M. Kavanaugh (Montgomery Watson), J. Mauro (SC&A), E. Makhlouf (Montgomery Watson), and B. Weiss (U. Rochester).

Seitz, R., P.D. Rittmann, J. Cook, and M. Wood. 1994. Comparison of PATHRAE and GENII, DOE Performance Assessment Task Team.

Summary of Developer/User Testing and Peer Review of GENII for which Documentation is Available

Baker, D. 1987. Review of HEDUP documentation and QA.

Carter, M.(Georgia Institute of Technology), K. Eckerman (Oak Ridge National Laboratory), J. Johnson (Chalk River Laboratory). 1987. External peer review panel.
R.Gray. 1988. Extramural panel review of GENII.

Napier, B.A. 1990. GENII "Conversion Testing, Verification, and Validation of Software" plan listing 42 tests performed as of 2/7/1989.

Rhoads, K. 1990. Review of acute dispersion calculation GENII Version 1.449.

Winter, R., G. Anast, H. Avci, M. Biggerston, D. Smith. 1990. Informal review of GENII verification and validation. Argonne National Laboratory/ DOE-HQ.

Nelson, I.C., L.H. Sawyer, T.A. Ikenberry. 1990. Hand Calculations performed on GENII to support NPR-EIS program.

Sawyer, L.H., T.A. Ikenberry. 1991. Hand calculations performed to support acute models in GENII.

Cammann, J. and P.D. Rittmann. 1990. Revisions to GENII dose increment libraries.

Peloquin, R.A., 1994. GENII Hand Calculation Worksheets, version of February 2, 1994.

SEPARATION

PAGE

Defense Nuclear Facilities Safety Board Recommendation 2002-1
Software Quality Assurance Improvement Plan
Commitment 4.2.1.3:

Software Quality Assurance Improvement Plan: ALOHA Gap Analysis

Final Report

U.S. Department of Energy
Office of Environment, Safety and Health
1000 Independence Ave., S.W.
Washington, DC 20585-2040

May 2004

INTENTIONALLY BLANK

FOREWORD

This report documents the outcome of an evaluation of the Software Quality Assurance (SQA) attributes of the chemical source term and atmospheric dispersion computer code, ALOHA 5.2.3, relative to established requirements. This evaluation, a "gap analysis", is performed to meet commitment 4.2.1.3 of the Department of Energy's Implementation Plan to resolve SQA issues identified in the Defense Nuclear Facilities Safety Board Recommendation 2002-1.

Suggestions for corrections or improvements to this document should be addressed to –

Chip Lagdon
EH-31/GTN
U.S. Department of Energy
Washington, D.C. 20585-2040
Phone (301) 903-4218
Email: chip.lagdon@eh.doe.gov

INTENTIONALLY BLANK

REVISION STATUS

Page/Section	Revision	Change
1. Entire Document	1. Interim Report	1. Original Issue
1. Entire Document	1. Final Report, May 3, 2004	1. Updated all sections per review comments.

INTENTIONALLY BLANK

CONTENTS

Section			Page
FOREWORD			III
REVISION STATUS			V
EXECUTIVE SUMMARY			XIII
1.0	INTRODUCTION		1-1
	1.1	BACKGROUND: OVERVIEW OF DESIGNATED TOOLBOX SOFTWARE IN THE CONTEXT OF 10 CFR 830	1-1
	1.2	EVALUATION OF TOOLBOX CODES	1-2
	1.3	USES OF THE GAP ANALYSIS	1-2
	1.4	SCOPE	1-2
	1.5	PURPOSE	1-2
	1.6	METHODOLOGY FOR GAP ANALYSIS	1-2
	1.7	SUMMARY DESCRIPTION OF SOFTWARE BEING REVIEWED	1-4
2.0	ASSESSMENT SUMMARY RESULTS		2-1
	2.1	CRITERIA MET	2-1
	2.2	EXCEPTIONS TO REQUIREMENTS	2-1
	2.3	AREAS NEEDING IMPROVEMENT	2-2
	2.4	CONCLUSION REGARDING CODES ABILITY TO MEET INTENDED FUNCTION	2-4
3.0	LESSONS LEARNED		3-1
4.0	DETAILED RESULTS OF THE ASSESSMENT PROCESS		4-1
	4.1	TOPICAL AREA 1 ASSESSMENT: SOFTWARE CLASSIFICATION	4-1
		4.1.1 Criterion Specification and Result	*4-1*
		4.1.2 Sources and Method of Review	*4-2*
		4.1.3 Software Quality-Related Issues or Concerns	*4-2*
		4.1.4 Recommendations	*4-2*
	4.2	TOPICAL AREA 2 ASSESSMENT: SQA PROCEDURES AND PLANS	4-3
		4.2.1 Criterion Specification and Result	*4-3*
		4.2.2 Sources and Method of Review	*4-4*
		4.2.3 Software Quality-Related Issues or Concerns	*4-4*
		4.2.4 Recommendations	*4-4*
	4.3	TOPICAL AREA 3 ASSESSMENT: REQUIREMENTS PHASE	4-4
		4.3.1 Criterion Specification and Result	*4-4*
		4.3.2 Sources and Method of Review	*4-5*
		4.3.3 Software Quality-Related Issues or Concerns	*4-5*
		4.3.4 Recommendations	*4-5*
	4.4	TOPICAL AREA 4 ASSESSMENT: DESIGN PHASE	4-6
		4.4.1 Criterion Specification and Result	*4-6*
		4.4.2 Sources and Method of Review	*4-8*
		4.4.3 Software Quality-Related Issues or Concerns	*4-8*
		4.4.4 Recommendations	*4-8*
	4.5	TOPICAL AREA 5 ASSESSMENT: IMPLEMENTATION PHASE	4-8
		4.5.1 Criterion Specification and Result	*4-8*
		4.5.2 Sources and Method of Review	*4-10*

4.5.3 Software Quality-Related Issues or Concerns *4-10*
4.5.4 Recommendations *4-10*
4.6 TOPICAL AREA 6 ASSESSMENT: TESTING PHASE 4-10
4.6.1 Criterion Specification and Result *4-10*
4.6.2 Sources and Method of Review *4-13*
4.6.3 Software Quality-Related Issues or Concerns *4-14*
4.6.4 Recommendations *4-14*
4.7 TOPICAL AREA 7 ASSESSMENT: USER INSTRUCTIONS 4-14
4.7.1 Criterion Specification and Result *4-14*
4.7.2 Sources and Method of Review *4-15*
4.7.3 Software Quality-Related Issues or Concerns *4-15*
4.7.4 Recommendations *4-15*
4.8 TOPICAL AREA 8 ASSESSMENT: ACCEPTANCE TEST 4-15
4.8.1 Criterion Specification and Result *4-15*
4.8.2 Sources and Method of Review *4-16*
4.8.3 Software Quality-Related Issues or Concerns *4-16*
4.8.4 Recommendations *4-16*
4.9 TOPICAL AREA 9 ASSESSMENT: CONFIGURATION CONTROL 4-16
4.9.1 Criterion Specification and Result *4-17*
4.9.2 Sources and Method of Review *4-17*
4.9.3 Software Quality-Related Issues or Concerns *4-17*
4.9.4 Recommendations *4-17*
4.10 TOPICAL AREA 10 ASSESSMENT: ERROR IMPACT 4-17
4.10.1 Criterion Specification and Result *4-17*
4.10.2 Sources and Method of Review *4-19*
4.10.3 Software Quality-Related Issues or Concerns *4-19*
4.10.4 Recommendations *4-19*
4.11 TRAINING PROGRAM ASSESSMENT 4-19
4.12 SOFTWARE IMPROVEMENTS 4-19

5.0 CONCLUSION 5-1

6.0 ACRONYMS AND DEFINITIONS 6-1

7.0 REFERENCES 7-1

APPENDIX A. — SOFTWARE INFORMATION TEMPLATE 2

INTENTIONALLY BLANK

TABLES

	Page
Table 1-1. – Plan for SQA Evaluation of Existing Safety Analysis Software	1-3
Table 1-2 — Summary Description of ALOHA Software	1-5
Table 1-3 — Software Documentation Reviewed for ALOHA	1-7
Table 2-1 — Summary of Important Exceptions, Reasoning, and Suggested Remediation	2-1
Table 2-2 — Summary of Important Recommendations for ALOHA	2-2
Table 4.0-1. — Cross-Reference of Requirements with Subsection and Entry from DOE (2003e)	4-1
Table 4.1-1 — Subset of Criteria for Software Classification Topic and Results	4-2
Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results	4-3
Table 4.3-1 — Subset of Criteria for Requirements Phase Topic and Results	4-4
Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results	4-6
Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results	4-8
Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results	4-10
Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results	4-14
Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results	4-15
Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results	4-17
Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results	4-18

INTENTIONALLY BLANK

FIGURES

Page

None

Software Quality Assurance Improvement Plan: ALOHA Gap Analysis

EXECUTIVE SUMMARY

The Defense Nuclear Facilities Safety Board (DNFSB) issued Recommendation 2002-1 on *Quality Assurance for Safety-Related Software* in September 2002 (DNFSB 2002). The Recommendation identified a number of quality assurance issues for software used in the 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 high-use, Software Quality Assurance (SQA)-compliant safety analysis codes is one of the major improvement actions discussed in the *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities*. A DOE safety analysis toolbox would contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed and maintained for DOE-broad safety basis applications.

The ALOHA 5.2.3 software for chemical source term and atmospheric dispersion and consequence analysis, is one of the codes designated for the toolbox. To determine the actions needed to bring the ALOHA 5.2.3 code into compliance with the SQA qualification criteria, and develop an estimate of the resources required to perform the upgrade, the Implementation Plan has committed to sponsoring a code-specific gap analysis document. The gap analysis evaluates the software quality assurance attributes of ALOHA 5.2.3 against identified criteria.

The balance of this document provides the outcome of the ALOHA gap analysis compliant with NQA-1-based requirements as contained in U.S. Department of Energy, *Software Quality Assurance Plan and Criteria for Safety Analysis Toolbox Codes*, (DOE, 2003e). Of the ten SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), two requirements are met at acceptable level, i.e., *Classification* (1) and *User Instructions* (7). A third requirement, *Configuration Control* (9), is partially met. Improvement actions are recommended for ALOHA to fully meet *Configuration Control* (9) criteria and the remaining seven requirements. This evaluation outcome is deemed acceptable because: (1) ALOHA 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 ALOHA is limited to those analytic applications for which the software is intended.

Suggested remedial actions for this software would warrant upgrading software documents. The complete list of revised 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 Manual.

As part of this effort, the draft National Oceanic and Atmospheric Administration (NOAA) theoretical description memorandum for ALOHA 5.0 (Reynolds, 1992), which is the main source of information for technical information, should be updated for recent upgrades, technically reviewed, and issued as final.

It is estimated that a concentrated program to upgrade the SQA pedigree of ALOHA to be compliant with the ten criteria discussed here would require fourteen to sixteen full-time equivalent (FTE)-months. Technical review of the chemical databases associated with this software is assumed to have been performed, and is not included in the level-of-effort estimate.

A new version of ALOHA, namely ALOHA 5.3, was released in March 2004 just prior to the issuance of this report. It is recommended that this version be evaluated relative to the software improvement and baseline document recommendations, as well as the full set of SQA criteria discussed in this report. If this version is found to be satisfactory, it should replace version 5.2.3 as the designated version of the software for the toolbox.

It was determined that the ALOHA 5.2.3 code does meet its intended function for use in supporting documented safety analysis. However, as with all safety-related software, users should be aware of current limitations and capabilities of the software for supporting safety analysis. Informed use of the code can be assisted by appropriate use of current ALOHA documentation prepared by NOAA and the ALOHA code guidance report for DOE safety analysts, *ALOHA Computer Code Application Guidance for Documented Safety Analysis*, (DOE, 2004). Furthermore, while SQA improvement actions are recommended for ALOHA, no evidence has been found of programming, logic, or other types of software errors in ALOHA 5.2.3 that have led to non-conservatisms in nuclear facility operations, or in the identification of facility controls.

INTENTIONALLY BLANK

1.0 Introduction

This document reports on the results of a gap analysis for Version 5.2.3 of the ALOHA computer code. The intent of the gap analysis is to determine the actions needed to bring the designated software into compliance with established Software Quality Assurance (SQA) criteria. A secondary aspect of this report is to develop an estimate of the level of effort required to upgrade each code based on the gap analysis results

1.1 Background: Overview of Designated Toolbox Software in the Context of 10 CFR 830

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* (DNFSB, 2000). 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.

While progress was made in resolving several of the issues raised in TECH-25, the DNFSB issued Recommendation 2002-1 on *Quality Assurance for Safety-Related Software* in September 2002. 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, the quality of the software used to design or develop safety-related controls, and the proficiency of personnel using the software. The Recommendation identified a number of quality assurance issues for software used in the 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 high-use, SQA-compliant safety analysis codes is one of the major commitments contained in the March 2003 *Implementation Plan for Recommendation 2002-1 on Quality Assurance for Safety Software at Department of Energy Nuclear Facilities* (IP). In time, the DOE safety analysis toolbox will contain a set of appropriately quality-assured, configuration-controlled, safety analysis codes, managed and maintained for DOE-broad safety basis applications.

Six computer codes, including ALOHA (chemical release dispersion/consequence analysis), CFAST (fire analysis), EPIcode (chemical release dispersion/consequence analysis), GENII (radiological dispersion/consequence analysis), MACCS2 (radiological dispersion/consequence analysis), and MELCOR (leak path factor analysis), were designated by DOE for the toolbox (DOE/EH, 2003). It is found that this software provides generally recognized and acceptable approaches for modeling source term and consequence phenomenology, and can be applied as appropriate to support accident analysis in Documented Safety Analyses (DSAs).

As one of the designated toolbox codes, ALOHA Version 5.2.3, is likely to require some degree of quality assurance improvement before meeting current SQA standards. The analysis of this document evaluates ALOHA Version 5.2.3 relative to current software quality assurance criteria. It assesses the extent of the deficiencies, or gaps, to provide DOE and the software developer the extent to which minimum upgrades are needed. The overall assessment is therefore termed a "gap" analysis.

1.2 Evaluation of Toolbox Codes

The quality assurance criteria identified in later sections of this report are defined as the set of established requirements, or basis, by which to evaluate each designated toolbox code. This evaluation process, a gap analysis, is commitment 4.2.1.3 in the IP:

> Perform a SQA evaluation to the toolbox codes to determine the actions needed to bring the codes into compliance with the SQA qualification criteria, and develop a schedule with milestones to upgrade each code based on the SQA evaluation results.

This process is a prerequisite step for software improvement. It will allow DOE to determine the current limitations and vulnerabilities of each code as well as help define and prioritize the steps required for improvement.

Ideally, each toolbox code owner will provide complete information on the SQA programs, processes, and procedures used to develop their software. However, the gap analysis itself will be performed by a SQA evaluator. The SQA evaluator is independent of the code developer, but knowledgeable in the use of the software for accident analysis applications and current software development standards.

1.3 Uses of the Gap Analysis

The gap analysis provides key information to DOE, code developers, and code users.

DOE obtains the following benefits:

- Estimate of the resources required to perform modifications to designated toolbox codes
- Basis for schedule and prioritization to upgrade each designated toolbox code.

Each code developer is provided:

- Information on areas where software quality assurance improvements are needed to comply with industry SQA standards and practices
- Specific areas for improvement to guide development of new versions of the software.

DOE safety analysts and code users benefit from:

- Improved awareness of the strengths, limits, and vulnerable areas of each computer code
- Recommendations for code use in safety analysis application areas.

1.4 Scope

This analysis is applicable to the ALOHA 5.2.3 code, one of the six designated toolbox codes for safety analysis. While ALOHA 5.2.3 is the subject of the current report, other safety analysis software considered for the toolbox in the future may be evaluated with the same process applied here. The template outlined here is applicable for any analytical software as long as the primary criteria are ASME NQA-1, 10 CFR 830, and related DOE directives discussed in DOE (2003e).

1.5 Purpose

The purpose of this report is to document the gap analysis performed on the ALOHA 5.2.3 code as part of DOE's implementation plan on SQA improvements.

1.6 Methodology for Gap Analysis

The gap analysis for ALOHA 5.2.3 is based on the plan and criteria described in *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes* (DOE 2003e). The overall methodology for the gap analysis is summarized in Table 1-1. The gap analysis reported here utilizes ten of the fourteen topical areas listed in DOE (2003e) related to software quality assurance to assess the

quality of the ALOHA 5.2.3 code. The ten areas are those particularly applicable to the software development, specifically: (1) Software Classification, (2) SQA Procedures/Plans, (5) Requirements Phase, (6) Design Phase, (7) Implementation Phase, (8) Testing Phase, (9) User Instructions, (10) Acceptance Test, (12) Configuration Control, and (13) Error Impact. Each area, or requirement, is assessed individually in Section 4.

Requirements 3 (Dedication), 4 (Evaluation), 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 evaluating the areas listed above. Requirement 11 (Operation and Maintenance) is only partially applicable to software development, and is interpreted to be applicable mostly to the software user organization.

An information template was transmitted to the Safety Analysis Software Developers on 20 October 2003 to provide basic information as input to the gap analysis process (O'Kula, 2003). The core section of the template is attached as Appendix A to the present report. While the ALOHA software developers did not provide a written response using the template, they provided information intermittently through less formal means.

Table 1-1. – Plan for SQA Evaluation of Existing Safety Analysis Software[1]

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.

[1] Originally documented as Table 2-2 in DOE (2003e).

Phase	Procedure
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.
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.

1.7 Summary Description of Software Being Reviewed

The gap analysis was performed on version 5.2.3 of the Areal Locations of Hazardous Atmospheres (ALOHA) code (NOAA, 1999a) as this was the current version during the course of the evaluation.[2] ALOHA 5.2.3 was released in 1999. ALOHA is a public domain code that is part of a system of software that is known as the Computer-Aided Management of Emergency Operations (CAMEO) that was developed to plan for and respond to chemical emergencies. It is also widely used throughout the DOE complex for safety analysis applications.

Specifically, ALOHA performs calculations for source terms and downwind concentrations. Source term calculations determine the rate at which the chemical material is released to the atmosphere, release duration, and the physical form of the chemical upon release. The analyst specifies the chemical and then characterizes the initial boundary conditions of the chemical with respect to the environment through the source configuration input. The ALOHA code allows for the source to be defined in one of four ways (i.e., direct source, puddle source, tank source, or pipe source) in order to model various accident scenarios. The source configuration input is used to either specify the chemical source term or to provide ALOHA with the necessary information and data to calculate transient chemical release rates and physical state of the chemical upon release.

The ALOHA code considers two classes of atmospheric transport and dispersion based upon the assumed interaction of the released cloud with the atmospheric wind flow.

[2] A new version of ALOHA, namely ALOHA 5.3, was released in March 2004 just prior to the issuance of this report.

- For airborne releases in which the initial chemical cloud density is less than or equal to that of the ambient air, ALOHA treats the released chemical as neutrally buoyant.
- Alternatively, if the density of the initial chemical cloud is greater than that of the ambient air, then the possibility exists for either neutrally buoyant or dense-gas type of atmospheric transport and dispersion.

In addition to the source term and downwind concentration calculations, ALOHA allows for the specification of concentration limits for the purpose of consequence assessment (e.g., assessment of human health risks from contaminant plume exposure). ALOHA refers to these concentration limits as level-of-concern (LOC) concentrations. Safety analysis work uses the emergency response planning guidelines (ERPGs) and temporary emergency exposure limits (TEELs) for assessing human health effects for both facility workers and the general public (Craig, 2001). While ERPGs and TEELs are not explicitly a part of the ALOHA 5.2.3 chemical database[3], ALOHA 5.2.3 allows the user to input an ERPG or TEEL value as the LOC concentration.

A brief summary of ALOHA that was supplied code developer is summarized in Table 1-2.

Table 1-2 — Summary Description of ALOHA Software

Type	Specific Information
Code Name	ALOHA (Areal Locations of Hazardous Atmospheres)
Version of the Code	Version 5.2.3
Developing Organization and Sponsor Information	DOC/NOAA/NOS Office of Response and Restoration And EPA Office of Emergency Prevention, Preparedness, and Response
Auxiliary Codes	Codes ALOHA is a standalone program but can be used in conjunction with CAMEO and MARPLOT. For more information, see http://response.restoration.noaa.gov
Software Platform/Portability	Available for Macintosh computers running OS 8, OS 9, or OS X; Available for any personal computer that runs Windows 98, 2000, NT, XP, or ME operating systems.
Coding and Computer(s)	C Code
Technical Support Point of Contact	Robert Jones NOAA/ORR 7600 Sand Point Way, Seattle, WA 98115 206-526-4278 Robert.jones@noaa.gov
Code Procurement Point of Contact	A self-extracting installer can be downloaded from: http://www.epa.gov/ceppo/cameo/aloha.htm Mark W Miller DOC/NOAA/NOS/ORR 7600 Sand Point Way, Seattle, WA 98115 206-526-6272 mark.w.miller@noaa.gov

[3] The ALOHA 5.2.3 chemical database incorporates two sets of concentration limits that are used in the chemical industry to address worker safety issues: (1) immediately dangerous to life or health (IDLH) and (2) threshold limit value – time weighted average (TLV-TWA). ALOHA 5.3, which was released in March 2004 just prior to the issuance of this report, does include TEELs and ERPGs.

Type	Specific Information
Code Package Label/Title	aloha.exe – Windows alohains.sit.hqx - Macintosh
Contributing Organization(s)	DOC/NOAA/NOS Office of Response and Restoration and EPA Office of Emergency Prevention Preparedness and Response
Recommended Documentation - Supplied with Code Transmittal upon Distribution or Otherwise Available	1. The ALOHA manual is a 1.5 MB PDF file (aloha.pdf) that can be downloaded directly from http://www.epa.gov/ceppo/cameo/aloha.htm
Input Data/Parameter Requirements	The location and chemical must be selected from scrolling lists. In some cases, the user must specify the concentration level to be displayed. Wind speed, direction, ground roughness, cloud cover, humidity, air temperature, and inversion height must be selected. The inputs needed to specify the source strength depend upon the scenario chosen; the simplest is the direct source and requires the mass or volume release rate.
Summary of Output	Output is provided in text and graphical form, including - rate at which the pollutant is entering the atmosphere as a function of time - indoor and outdoor concentrations as a function of time at a user-defined location - spatial distribution corresponding to the condition that the maximum concentration exceeds a user-specified level of concern
Nature of Problem Addressed by Software	ALOHA provides conservative estimates of the spatial distribution of the peak concentration of a pollutant following an acute release. To accomplish this, ALOHA contains an extensive database of chemical properties, models for estimating the amount of material entering the atmosphere for a wide range of scenarios, and Gaussian and dense gas (based on DEGADIS) dispersion models.
Significant Strengths of Software	ALOHA contains an extensive database of chemical properties so no additional information beyond the chemical identity is required. ALOHA has submodels for estimating the amount of pollutant entering the atmosphere (source strength). ALOHA has a dispersion model capable of accounting for the gravity effects on dense gas dispersion. ALOHA displays uncertainty associated with wind direction. ALOHA's interface is designed to assist users by including intelligent default entries where appropriate, reasonableness checks for input and context sensitive helps which include data entry guidance.
Known Restrictions or Limitations	ALOHA is designed to estimate the airborne concentration of pollutants over a relatively short time, one hour, and short spatial extent, 10 kilometers. With this restriction, the use of steady-state meteorology is acceptable. ALOHA does not account for steering by local topography, particulates, or reactions (including fire).
Preprocessing (set-up) time for Typical Safety Analysis Calculation	5 - 15 minutes
Execution Time	1 – 10 seconds
Computer Hardware Requirements	Any computer capable of running the operating systems noted above can run ALOHA.

Type	Specific Information
Computer Software Requirements	None
Other Versions Available	N/A
Individual(s) completing this information form: Name: Organization: Telephone: Email: Fax:	 Mark W Miller DOC/NOAA/NOS/ORR 206-526-6272 mark.w.miller@noaa.gov 206-526-6329

The set of documents reviewed as part of the gap analysis are listed in Table 1-3.

Table 1-3 — Software Documentation Reviewed for ALOHA

No.	Reference
1.	*ALOHA User's Manual* (NOAA, 1999a)
2.	*ALOHA 5.2.3 Online Help* (NOAA, 1999b)
3.	*ALOHA Theoretical Description* (Reynolds, 1992) – Draft document
4.	*ALOHA User's and ARCHIE: A Comparison*, Report No. HAZMAT 93-2 (M. Evans, 1993)
5.	*Quality Assurance of ALOHA* (M. Evans, 1994) – Draft document
6.	http://www.nwn.noaa.gov/sites/hazmat/cameo/alotech/quality.html
7.	http://www.nwn.noaa.gov/sites/hazmat/cameo/aloha.html
8.	http://www.epa.gov/ceppo/cameo/instruct.htm
9.	http://response.restoration.noaa.gov/cameo/aloha.html
10.	http://response.restoration.noaa.gov/cameo/alohafaq/history.html

2.0 Assessment Summary Results

2.1 Criteria Met

Of the ten general topical quality areas assessed in the gap analysis, two satisfactorily met the criteria. The analysis found that the ALOHA 5.2.3 SQA program, in general, met criteria for *Software Classification* and *User Instructions*, which refer to Requirements 1 and 7, respectively. A third topical area, *Configuration Control*, partially met criteria, but it and the remaining seven topical quality areas were judged either not wholly compliant with the SQA criteria, and/or lacked documentation to confirm compliance. The eight areas that should be addressed for improvement actions are listed in Section 2.2 (Exceptions to Requirements). Details on the evaluation process relative to the requirements and the criteria applied, are found in Section 4.

2.2 Exceptions to Requirements

Exceptions to criteria found for ALOHA 5.2.3 are listed below in Table 2-1. The requirement is given, the reason the requirement was not met is provided, and action(s) are listed to correct the exceptions. The ten criteria evaluated are those predominantly executed by the software developer. However, it is noted that criteria for SQA Procedures/Plan, Testing, Acceptance Test, Configuration Control, and Error Notification also have requirements for the organization implementing the software. These criteria were assessed in the present evaluation only from the code developer perspective.

Table 2-1 — Summary of Important Exceptions, Reasoning, and Suggested Remediation

No.	Criterion	Reason Not Met	Remedial action(s)
1.	SQA Procedures/Plans (Section 4.2)	SQA Plans and Procedures were not available for the gap analysis.	SQA Plans and Procedures should be developed and made available for review.
2.	Requirements Phase (Section 4.3)	A Software Requirements Document does not exist for review. Thus, it was necessary to infer requirements from draft model description and user guidance documents.	A Software Requirements Document should be prepared and made available for review.
3.	Design Phase (Section 4.4)	A Software Design Document does not exist for review. Thus, it was necessary to infer the intent of the design from draft model description and user guidance documents.	A Software Design Document should be prepared and made available for review. As part of this effort, the draft NOAA theoretical description memorandum for ALOHA 5.0 (Reynolds, 1992), which is the main source for technical information, should be updated for recent upgrades, technically reviewed, and issued as final.
4	Implementation Phase (Section 4.5)	Documentation to support the implementation is lacking.	A verifiable, written set of SQA plans and procedures including

No.	Criterion	Reason Not Met	Remedial action(s)
			implementation, test case descriptions, and associated criteria related to design should be made available.
5.	Testing Phase (Section 4.6)	A Software Testing Report Document does not exist for review. The documentation of results from validation and benchmark activities are incomplete and in the form of summaries that are found at ALOHA websites.	A Software Testing Report Document should be prepared and made available for review.
6	Acceptance Test (Section 4.8)	A verifiable, written set of SQA plans and procedures, which would include acceptance testing documentation, is lacking.	Documented acceptance testing should be developed.
7	Configuration Control (Section 4.9)	A Configuration Control process is in place at NOAA, but limited documentation was forwarded to allow a gap analysis to be performed.	While a Configuration Control process is apparently functional at NOAA, written documentation should be prepared and made available for review.
8.	Error Notification (Section 4.10)	An Error Notification and Corrective Action Report does not exist for review.	While a Software Problem Reporting system is apparently in place, written documentation should be provided to the Central Registry for verification of its effectiveness.

2.3 Areas Needing Improvement

The gap analysis identified a number of improvements that could be made related to the code and its quality assurance. These recommendations are listed in Table 2-2.

Table 2-2 — Summary of Important Recommendations for ALOHA

No.	Recommendation
1.	Correct a reported IDLH bug (e-mail to Mark Miller at NOAA on 11/13/2003). The footprint information gives results for the distance that corresponds to the maximum threat zone for IDHL. When the centerline concentration output is requested at this distance, the concentration results are expected to be the IDLH concentration or very close to it. This is not always the case. (Note: The footprint information output seems to be the source of problem, and neither footprint output or IDLH data are typically not used in DSA applications.)

No.	Recommendation
2.	Provide method to write-protect the Chemical Library. In previous versions of ALOHA, the Chemical Library was protected from inadvertent changes by requiring the use of another program, ChemManager. In the current version, this is not the case; permanent changes may be made within ALOHA code itself. This allows any user to permanently change the chemical library. This is especially problematic, in that users have previously been allowed to make changes knowing that they could not alter the chemical library itself. Allowing some method of protecting the chemical library would be beneficial. Although this can be done within the operating system itself by write protecting the ChemLib file, not all users will be knowledgeable enough to know this, and not all installations will write protect the file.
3.	Add capability to model release durations that are greater than one hour and downwind distances that are greater than 10 km. Although we recognize the purpose of this limitation, for safety analysis purposes, it is standard procedure to model releases using persistent meteorology and a straight-line Gaussian plume to a receptor at the site boundary. As many DOE sites are quite large (hundreds of square miles), this forces an analyst to use another tool to perform the same task. Rather than increasing the limit, we would rather it be removed altogether. While this may allow for unrealistic real-time use, it is typically required for bounding consequence calculations.
4.	Add capability to output consequences for multiple receptors in a single ALOHA run. DSA analyses may need a set of several receptors (e.g., 30m, 100m, 500m, 1km etc.) for which consequences must be determined for every postulated accident scenario. Having the ability to get this output without having to perform a run for each receptor would save time and money on performance and review, and decrease the size of documents. In tandem with the above request, the ability to output a graph of concentration versus centerline distance would be helpful, especially for elevated releases in which the maximum downwind concentration is desired and the distance where this occurs cannot be known apriori.
5. a.) b.)	Add capabilities to facilitate evaporation calculations for chemicals that are not part of ALOHA's library: Add capability to directly input vapor pressure rather than the only option being for ALOHA to calculate it from chemical properties. Occasionally, releases must be modeled for chemicals that are not in ALOHA's library. For some chemicals, though not all physical property data needed by ALOHA to calculate the vapor pressure is available, the vapor pressures themselves are available. It would be helpful if a vapor pressure could be directly entered and used by ALOHA to calculate an evaporative source term. Add capability so a simpler evaporation model as an option to use (one that did not require quite so much physical property data) when insufficient physical property data is known to use the ALOHA evaporation model. The uncertainty in the release quantity is usually far greater than that in the calculation of evaporative source term so the loss of accuracy would not normally be a problem.
6.	Add capability to read from a file of hourly meteorological data over a one-year period, calculate consequences for each hourly entry, and output the 50th and 95th percentile results.

No.	Recommendation
7.	Add capability for ALOHA either to use other sets of dispersion coefficients in addition to the two that are currently available (rural or urban) or to make user-specified adjustments to the dispersion coefficients as noted below:
a.)	Add capability to use the surface roughness input to adjust the rural vertical dispersion coefficient when the input value is greater than 3 cm and less than 100 cm. This will allow more accurate modeling for the majority sites that have surface roughness characteristics that fall in between the two extremes of flat grassland and an urban environment.
b.)	Add capability to adjust the horizontal dispersion coefficients for averaging time to account for specific exposure times that associated with toxic exposure guidelines of interest.
8.	Add capability to model dry deposition. A simple point depletion model could serve this purpose.
9.	For puddle modeling, allow model to calculate surface area from input of volume (or mass) and puddle depth. When using the code for planning rather than for response, this would be more useful than the current options of inputting the area or diameter, then the volume, depth, or mass.
10.	Add explosion modeling capability. A number of DOE sites have begun to look at explosive dispersal of toxicological material. It would be useful to be able to use the Gaussian plume model of ALOHA to estimate downwind concentrations.
11.	Reword or remove from the initial screen, the limitation on modeling particulates. As dispersion of small (respirable) particles is similar to that of gases, ALOHA is often used in the DOE complex to model respirable aerosols, including powders. The wording of this limitation, for some customers, unnecessarily calls into question this practice.
12	Update, technically review, and issue as final the draft NOAA theoretical description memorandum for ALOHA 5.0 that is the main source of information for technical information (Reynolds, 1992).
13.	Add capability to use long filenames for ALOHA save files.

2.4 Conclusion Regarding Codes Ability to Meet Intended Function

The ALOHA 5.2.3 code was evaluated to determine if the software in its current state meets the intended function in a safety analysis context as assessed in this gap analysis. When the code is run for the intended applications as detailed in the code guidance document, *ALOHA Computer Code Application Guidance for Documented Safety Analysis*, (DOE 2004), it is judged that it will meet its intended function.

3.0 Lessons Learned

Additional opportunities and venues should be sought for training and user qualification on safety analysis software. This is a long-term recommendation for ALOHA and other designated software for the DOE toolbox.

4.0 Detailed Results of the Assessment Process

Ten topical areas, or requirements, are presented in the assessment as listed in Table 4.0-1. Training and Software Improvements sections follow the ten topical areas. Included in the software improvements section is an estimate of the resources required to upgrade ALOHA.

In the tables that follow, criteria and recommendations are labeled as (1.x, 2.x, ...10.x) with the first value (1., 2., ...) corresponding to the topical area and the second value (x), the sequential table order.

Table 4.0-1. — Cross-Reference of Requirements with Subsection and Entry from DOE (2003e)

Subsection (This Report)	Corresponding Entry Table 3-3 from DOE (2003e) No.	Requirement
4.1	1	Software Classification
4.2	2	SQA Procedures/Plans
4.3	5	Requirements Phase
4.4	6	Design Phase
4.5	7	Implementation Phase
4.6	8	Testing Phase
4.7	9	User Instructions
4.8	10	Acceptance Test
4.9	12	Configuration Control
4.10	13	Error Notification

4.1 Topical Area 1 Assessment: Software Classification

This area corresponds to the requirement entitled Software Classification in Table 3-2 of (DOE 2003e).

4.1.1 Criterion Specification and Result

Table 4.1-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Sufficient documentation is provided with software transmittal to make an informed determination of the classification of the software. A user of the ALOHA software for safety analysis applications would be expected to interpret the information on the software in light of the requirements for atmospheric dispersion and consequence analysis discussed in Appendix A to DOE-STD-3009-94 to decide on an appropriate safety classification. For most organizations, the safety class or safety significant classification, or Level B in the classification hierarchy discussed in DOE (2003e), would be selected.

Table 4.1-1 — Subset of Criteria for Software Classification Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
1.1	The code developer must provide sufficient information to allow the user to make an informed decision on the classification of the software.	Yes.	It is concluded that sufficient information is provided with the documentation that is transmitted with the software for the user to make an informed determination of the classification of the software. For most DSA applications, the safety class or safety significant classification, or Level B in the classification hierarchy discussed in DOE (2003e), would be selected, which by definition relate to 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 ➢ Whose results are used to make decisions that could result in death or serious injury or are part of the evaluation in accident analyses.

4.1.2 Sources and Method of Review

Documentation that was distributed with the software package (plus information on ALOHA websites) and additional documentation that was supplied by the code developers for this effort were used as the bases for response to this requirement.

4.1.3 Software Quality-Related Issues or Concerns

There are no SQA issues or concerns relative to this requirement.

4.1.4 Recommendations

No recommendations are provided at this time.

4.2 Topical Area 2 Assessment: SQA Procedures and Plans

This area corresponds to the requirement entitled SQA Procedures and Plans in Table 3-3 of (DOE 2003e).

From the limited information received from the software developers, formal, published SQA procedures and plans were not developed. While it is possible that most elements of a compliant SQA program were followed in the development of ALOHA 5.2.3, the lack of written documentation prevents an independent evaluator from making a definitive confirmation. Based on discussions with the code developer, organizational management of the ALOHA 5.2.3 development probably ensured that some, maybe many, elements of a compliant SQA program were fulfilled in an informal manner.

4.2.1 Criterion Specification and Result

Table 4.2-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.2-1 — Subset of Criteria for SQA Procedures and Plans Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
2.1	Procedures/plans for SQA (**SQA Plan**) have identified organizations responsible for performing work; independent reviews, etc.	Uncertain.	A verifiable, complete written set of SQA plans and procedures is lacking for ALOHA. Based on discussions with the code developer, organizational management of the ALOHA development probably ensured that some elements of a compliant SQA program were fulfilled in an informal manner.
2.2	Procedures/plans for SQA (**SQA Plan**) have identified software engineering methods.	Uncertain.	See Criterion 2.1 summary remarks.
2.3	Procedures/plans for SQA (**SQA Plan**) have identified documentation to be required as part of program.	Uncertain.	See Criterion 2.1 summary remarks.
2.4	Procedures/plans for SQA (**SQA Plan**) have identified standards, conventions, techniques, and/or methodologies that shall be used to guide the software development, methods to ensure compliance with the same.	Uncertain.	See Criterion 2.1 summary remarks.
2.5	Procedures/plans for SQA (**SQA Plan**) have identified software reviews and schedule.	Uncertain.	See Criterion 2.1 summary remarks.
2.6	Procedures/plans for SQA (**SQA Plan**) have identified methods for error reporting and corrective actions.	Uncertain.	See Criterion 2.1 summary remarks.

4.2.2 Sources and Method of Review

Documentation that was distributed with the software package (plus information on ALOHA websites) and additional documentation that was supplied by the code developers for this effort were used as the primary bases for response to this requirement.

4.2.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures for ALOHA should be addressed.

4.2.4 Recommendations

Recommendations related to this topical area are provided as follows:

- It is recommended that a SQA plan be developed to provide a framework for configuration control, code maintenance, and support of future upgrades.

4.3 Topical Area 3 Assessment: Requirements Phase

This area corresponds to the requirement entitled Requirements Phase in Table 3-3 of (DOE 2003e).

4.3.1 Criterion Specification and Result

Table 4.3-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.3-1 — Subset of Criteria for Requirements Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
3.1	Software requirements for the subject software have been established.	Yes.	Implicitly fulfilled. The ALOHA program was developed to provide emergency response personnel and emergency planners with a software tool to evaluate downwind concentrations from the atmospheric release of toxic substances. It is a widely used computer code, which demonstrates that it serves the needs of many analysts. The code is regularly upgraded to improve capabilities. Specific requirements can be inferred from various ALOHA documents.
3.2	Software requirements are specified, documented, reviewed and approved.	No.	Software requirements have not been formally established. A verifiable, written set of SQA plans and procedures, which would include software requirements, is lacking for ALOHA.
3.3	Requirements define the functions to be performed by the software and provide detail and information necessary to design the software.	Partial.	Information sources for the technical details of the ALOHA algorithms are given in the ALOHA User's manual (NOAA, 1999a), the online help with

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			ALOHA 5.2.3 (NOAA, 1999b), a NOAA report (Evans, 1993) and a draft NOAA theoretical description memorandum (for ALOHA 5.0) (Reynolds, 1992). Information from ALOHA websites is also available. The ALOHA code uses the well-established models, such as the Gaussian puff and plume models. The draft NOAA theoretical description memorandum (for ALOHA 5.0) comprehensively documents these models (Reynolds, 1992). The document, however, is in draft form and should be updated to reflect upgrades that have been made over the past ten years.
3.4	A **Software Requirements Document**, or equivalent defines requirements for functionality, performance, design inputs, design constraints, installation considerations, operating systems (if applicable), and external interfaces necessary to design the software.	Partial.	The online user's documentation implicitly states requirements. The user's documentation also addresses, at least partially, installation, operating systems, external interfaces (e.g., MARPLOT) and design inputs.
3.5	Acceptance criteria are established in the software requirements documentation for each of the identified requirements.	No.	See Criterion 3.2 summary remarks.

4.3.2 Sources and Method of Review

Documentation that was distributed with the software package (plus information on ALOHA websites) and additional documentation that was supplied by the code developers for this effort were used as the primary bases for response to this requirement. The draft NOAA theoretical description memorandum (for ALOHA 5.0) is the main source of information for technical information (Reynolds, 1992).

4.3.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which would include written software requirements, for ALOHA should be addressed.

4.3.4 Recommendations

Recommendations related to this topical area are provided as follows:

- Formal documentation of the software requirements as in intended in ALOHA 5.2.3 will be needed for ALOHA to meet all prerequisites for the DOE toolbox.
- The draft NOAA theoretical description memorandum (for ALOHA 5.0) is the main source of information for technical information (Reynolds, 1992). It should be updated for recent upgrades, technically reviewed, and issued as final.

4.4 Topical Area 4 Assessment: Design Phase

This area corresponds to the requirement entitled Design Phase in Table 3-3 of (DOE 2003e).

4.4.1 Criterion Specification and Result

Table 4.4-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.4-1 — Subset of Criteria for Design Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
4.1	The software design was developed, documented, reviewed and controlled.	Partial.	Elements of this criterion may be inferred from documentation.
4.2	Code developer(s) 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.	Partial.	Design may be inferred from final software product, but design document was not made available for review.
4.3	The following design should be present and documented: specification of interfaces, overall structure (control and data flow) and the reduction of the overall structure into physical solutions (algorithms, equations, control logic, and data structures).	Partial.	Elements of this criterion may be inferred from documentation.
4.4	The following design should be present and documented: 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.	Partial.	Elements of this criterion may be inferred from documentation.
4.5	The following design should be present and documented: evidence of measures to mitigate the consequences of software design problems. These potential problems include external and internal abnormal conditions and events that can affect the computer program.	Not applicable to non-process, instrumentation and control software.	None.
4.6	A Software Design Document, or equivalent, is available and contains a description of the major components of the software design as they relate to the software requirements.	Partial.	Elements of this criterion may be inferred from documentation. A verifiable, written set of SQA plans and procedures, which would include software design documentation, is lacking for ALOHA.
4.7	A Software Design Document, or equivalent, is available and contains a technical description of the software with respect to the theoretical basis,	Partial.	See Criterion 4.6 summary remarks.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	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.		
4.8	A Software Design Document, or equivalent, is available and contains a description of the allowable or prescribed ranges for inputs and outputs.	Yes.	The ALOHA user documentation contains this information.
4.9	A Software Design Document, or equivalent, is available and contains the design described in a manner that can be translated into code.	Partial.	See Criterion 4.6 summary remarks.
4.10	A Software Design Document, or equivalent, is available and contains 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.	Partial.	See Criterion 4.6 summary remarks.
4.11	The organization responsible for the design identified and documented the particular verification methods to be used and assured that an Independent Review was performed and documented. This review evaluated the technical adequacy of the design approach; assured internal completeness, consistency, clarity, and correctness of the software design; and verified that the software design is traceable to the requirements.	Partial.	While some elements of this criterion may have been met informally per discussions with the software developer, there is no written documentation that allows confirmation.
4.12	The organization responsible for the design assured that the test results adequately demonstrated that the requirements were met.	Partial.	See Criterion 4.6 summary remarks.
4.13	The Independent Review (IR) was performed by competent individual(s) other than those who developed and documented the original design, but who may have been from the same organization.	Partial.	Significant review (see Criterion 4.5) was performed. Documentation of reviewer qualifications and independence is lacking.
4.14	The results of the IR are documented with the identification of the verifier indicated.	Partial.	See Criterion 4.13 summary remarks.
4.15	If review alone was not adequate to determine if requirements are met,	Partial.	See Criterion 4.5 summary remarks.

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	alternate calculations were used, or tests were developed and integrated into the appropriate activities of the software development cycle.		
4.16	Software design documentation was completed prior to finalizing the Independent Review.	Partial.	It appears that some reviews were conducted in parallel with design documentation preparation or before preparation of its equivalent,
4.17	The extent of the IR and the methods chosen are shown to be a function of: ➢ The importance to safety, ➢ The complexity of the software, ➢ The degree of standardization, and ➢ The similarity with previously proven software.	Partial.	Elements of this criterion may be inferred from documentation. Integrated documentation of the design requirements is lacking, as is documentation of the review details and bases.

4.4.2 Sources and Method of Review

Documentation that was distributed with the software package (plus information on ALOHA websites) and additional documentation that was supplied by the code developers for this effort were used as the primary bases for response to this requirement. The draft NOAA theoretical description memorandum (for ALOHA 5.0) is the main source of information for technical information (Reynolds, 1992).

4.4.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which would include software design documentation, for ALOHA should be addressed.

4.4.4 Recommendations

Recommendations related to this topical area are provided as follows:

- Formal documentation of the software design as in intended in ALOHA 5.2.3 will be needed for ALOHA to meet all prerequisites for the DOE toolbox.
- The draft NOAA theoretical description memorandum (for ALOHA 5.0) is the main source of information for technical information (Reynolds, 1992). It should be updated for recent upgrades, technically reviewed, and issued as final.

4.5 Topical Area 5 Assessment: Implementation Phase

This area corresponds to the requirement entitled Implementation Phase in Table 3-3 of (DOE 2003e).

4.5.1 Criterion Specification and Result

Table 4.5-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.5-1 — Subset of Criteria for Implementation Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
5.1	The implementation process resulted in	Yes.	User guide, draft technical

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	software products such as computer program listings and instructions for computer program use.		description report as well as web-posted information, and executable file demonstrate that the essential features of this criterion are met.
5.2	Implemented software was analyzed to identify and correct errors.	Partial.	Practical steps were taken by the code developers to identify problems through the use of fractional factorial designs, which ALOHA draft documentation describes as "a method of experimental design commonly used in industrial research." Using this method, ALOHA output was systematically compared against that of reference models to "identify and eliminate code errors, to find flaws in model algorithms, to identify aspects of the model requiring additional evaluation, to ensure that all substantial deviations of ALOHA's estimates from predictions made by similar models are the result of intended differences in algorithms" (Evans, 1994). Thus, documentation, especially the quality assurance draft report (Evans, 1994), supports partial satisfaction of essential features of this criterion in general for ALOHA development, but does specifically address ALOHA 5.2.3, which was released post-1994. This document should be updated as necessary, reviewed, and issued in final form.
5.3	The source code finalized during verification (this phase) was placed under configuration control.	Partial.	Discussions with the code developers indicate that software is managed by the program manager (currently Mark Miller) and under the direct control of the project manager (currently Jerry Muhasky). The project manager implements all the changes to the source code and

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			maintains the code on his computer, using password protection to control access. The computer is backed up on a NOAA server as well as CD copies that are stored in secure off site locations. The code developers indicate that there are plans to formally document configuration control procedures.
5.4	Documentation during verification included a copy of the software, test case description and associated criteria that are traceable to the software requirements and design documentation.	Partial.	The user's manual includes four sample problems that can serve as test cases. Guidance is given for each required input for each test case. Results are also given for each test case that can be compared to user-generated results. Not possible to trace to requirements and design descriptions since these are lacking documentation.

4.5.2 Sources and Method of Review

Documentation that was distributed with the software package (plus information on ALOHA website) and additional documentation that was supplied by the code developers for this effort were used as the primary bases for response to this requirement.

4.5.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which would include test case descriptions as well as software requirements and design documentation, for ALOHA should be addressed.

4.5.4 Recommendations

Recommendations related to this topical area are provided as follows:

- Formal documentation of the implication process as it relates to ALOHA 5.2.3 will be needed for ALOHA to meet all prerequisites for the DOE toolbox.

4.6 Topical Area 6 Assessment: Testing Phase

This area corresponds to the requirement entitled Testing Phase in Table 3-3 of (DOE 2003e).

4.6.1 Criterion Specification and Result

Table 4.6-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.6-1 — Subset of Criteria for Testing Phase Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
6.1	The software was validated by executing test cases.	Partial.	Documentation, especially the quality assurance draft

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			report (Evans, 1994), supports partial satisfaction of this criterion. Only partial credit is given since the document is draft form and only addresses early development of ALOHA. This document should be updated as necessary, reviewed, and issued in final form.
6.2	Testing demonstrated the capability of the software to produce valid results for test cases encompassing the range of permitted usage defined by the program documentation. Such activities provide evidence to ensure that the software adequately and correctly performed all intended functions and does not perform adverse unintended functions.	Partial.	Draft ALOHA documentation suggests that essential features of this criterion have been partially met informally (Evans, 1994). Only partial credit is given since the document is draft form and only addresses early development of ALOHA. Practical steps were taken by the code developers to identify problems through the use of fractional factorial designs. Using this method, ALOHA output was systematically compared against that of reference models for various combinations of input values and the results analyzed to ensure that all substantial deviations of ALOHA's estimates from predictions made by similar models were the result of intended differences in algorithms (Evans, 1994). Thus, documentation, especially the quality assurance draft report (Evans, 1994), supports partial satisfaction of essential features of this criterion. This document should be updated as necessary, reviewed, and issued in final form.
6.3	Testing demonstrated that the computer program properly handles abnormal conditions and events as well as credible	Partial.	While there is no formal documentation that addresses this issue completely, the

Criterion Number	Criterion Specification	Compliant	Summary Remarks
	failures.		user's manual does address warning messages that ALOHA provides when the user enters an input outside the allowable range or when the user-specified inputs define conditions that may result in phenomena that is outside of the capabilities of ALOHA. An example of the latter case occurs when the user selects an air- or water-reactive chemical for analysis. ALOHA informs the user of the type of reaction and expected reaction products. For example, sulfur trioxide reacts with water to form sulfuric acid and heat. ALOHA does not account for resulting phenomena such as buoyancy from the heat.
6.4	Testing demonstrated that the computer program does not perform adverse unintended functions.	Partial.	See Criterion 6.2 summary remarks.
6.5	Test Phase activities were performed to assure adherence to requirements, and to assure that the software produces correct results for the test case specified. Acceptable methods for evaluating adequacy of software test case results included: (1) analysis with computer assistance; (2) other validated computer programs; (3) experiments and tests; (4) standard problems with known solutions; (5) confirmed published data and correlations.	Partial.	Documentation, especially the quality assurance draft report (Evans, 1994), supports the satisfaction of essential features of this criterion. The results of comparisons of ALOHA predictions against field results as well as other computer codes are presented. This document should be updated as necessary, reviewed, and issued in final form. Documentation of requirements is lacking.
6.6	Test Phase documentation includes test procedures or plans and the results of the execution of test cases. The test results documentation demonstrates successful completion of all test cases or the resolution of unsuccessful test cases and provides direct traceability between the test results and specified software requirements	Partial.	Significant testing on ALOHA has been performed as discussed in the summary remarks of Criterion 6.2 and Criterion 6.5. However, successful resolution of unsuccessful cases is not possible to verify, nor is traceability between test

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			results and software requirements.
6.7	Test procedures or plans specify the following, as applicable: (1) required tests and test sequence, (2) required range of input parameters, (3) identification of the stages at which testing is required, (4) requirements for testing logic branches, (5) requirements for hardware integration, (6) anticipated output values, (7) acceptance criteria, (8) reports, records, standard formatting, and conventions, (9) identification of operating environment, support software, software tools or system software, hardware operating system(s) and/or limitations.	Partial.	Significant testing on ALOHA has been performed as discussed in the summary remarks of Criterion 6.2 and Criterion 6.5. No comprehensive detailed record of test procedures and plans was available. It can be inferred that this criterion was partially met.

Additional Detail

The following provides additional detailed explanation on selected criteria in the above table:

Criterion 6.1 — Details on the comparisons with field data are summarized below (Evans, 1994).

- Source term prediction for non-boiling pool evaporation – All ALOHA predictions were within 42% of measured evaporation rates.
- Source term prediction for liquefied propane – About 83% of ALOHA predictions were within a factor of two of measured vaporization rates.
- Atmospheric transport and dispersion predictions with Gaussian model – ALOHA predictions of mean downwind concentrations were on average 142% of the measured field data. ALOHA tended to underestimate concentrations at distances of 200 meters or more and overestimate concentrations closer in.
- Atmospheric transport and dispersion predictions with dense-gas model – ALOHA predictions were not compared directly with field measurements, but compared with results from the DEGADIS model that was calibrated to 12 trials from field experiments (Spicer, 1989). ALOHA predictions of mean downwind concentrations were on average 107% of DEGADIS predictions, and about 70% of DEGADIS predictions were within a factor of two of measured field concentrations.
- Atmospheric transport and dispersion predictions with dense-gas model for hydrogen fluoride (HF) releases – ALOHA predictions were not compared directly with field measurements, but compared with results from the DEGADIS model that was calibrated to 12 trials from field experiments (Spicer, 1989). ALOHA predictions of mean downwind concentrations were on average 48% of the measured field data.

4.6.2 Sources and Method of Review

Documentation that was distributed with the software package (plus information on ALOHA websites) and additional documentation that was supplied by the code developers for this effort were used as the primary bases for response to this requirement.

4.6.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which includes test reports, for ALOHA should be addressed.

4.6.4 Recommendations

Recommendations related to this topical area are provided as follows:

- It is recommended that benchmark comparisons and validation cases be updated and formally documented (current documentation is in the form of draft summary document that is dated 1994).
- It is recommended that formal test report documentation be established for future upgrades to the code.

4.7 Topical Area 7 Assessment: User Instructions

This area corresponds to the requirement entitled User Instructions in Table 3-3 of (DOE 2003e).

4.7.1 Criterion Specification and Result

Table 4.7-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.7-1 — Subset of Criteria for User Instructions Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
7.1	A description of the model is documented and made available to users.	Partial.	The draft NOAA theoretical description memorandum (for ALOHA 5.0) is the main source of information for technical information (Reynolds, 1992). It should be updated for recent upgrades, technically reviewed, and issued as final. Currently, this draft NOAA theoretical description memorandum is not readily available.
7.2	User's manual or guide describes software and hardware limitations and identifies includes approved operating systems (for cases where source code is provided, applicable compilers should be noted).	Yes.	(NOAA, 1999a; NOAA, 1999b)
7.3	User's manual or guide includes description of the user's interaction with the software.	Yes.	(NOAA, 1999a; NOAA, 1999b)
7.4	User's manual or guide includes a description of any required training necessary to use the software.	Yes.	The user's manual does not state the need for any required general training. The inference can be made that formal training, while recommended, may not be required for general use of the code. The user's manual and web-posted information provide amble guidance, address specific issues that an analyst is likely to encounter, and cover worked sample problems. In addition, a fair amount of training material is posted on the

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			EPA website (http://www.epa.gov/ceppo/cameo/instruct.htm). The user's manual does advise that a very specific type of ALOHA calculation, namely the calculation of dose from a chemical exposure, only be performed by someone trained in toxicology.
7.5	User's manual or guide includes input and output specifications.	Yes.	(NOAA, 1999a; NOAA, 1999b)
7.6	User's manual or guide includes a description of user messages initiated as a result of improper input and how the user can respond.	Yes.	(NOAA, 1999a; NOAA, 1999b)
7.7	User's manual or guide includes information for obtaining user and maintenance support.	Yes.	(NOAA, 1999a; NOAA, 1999b)

4.7.2 Sources and Method of Review

Documentation that was distributed with the software package (plus information on ALOHA websites) and additional documentation that was supplied by the code developers for this effort were used as the primary bases for response to this requirement.

4.7.3 Software Quality-Related Issues or Concerns

There are no SQA issues or concerns relative to this requirement.

4.7.4 Recommendations

Recommendations related to this topical area are provided as follows:

- The draft NOAA theoretical description memorandum (for ALOHA 5.0) is the main source of information for technical information on the models (Reynolds, 1992). It should be updated for recent upgrades, technically reviewed, and issued as final.

4.8 Topical Area 8 Assessment: Acceptance Test

This area corresponds to the requirement entitled Acceptance Test Table 3-3 of (DOE 2003e). During this phase of the software development, the software becomes part of a system incorporating applicable software components, hardware, and data and is accepted for use. Much of this testing is the burden of the user organization, but the developing organization shoulders some responsibility.

4.8.1 Criterion Specification and Result

Table 4.8-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.8-1 — Subset of Criteria for Acceptance Test Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks

Criterion Number	Criterion Specification	Compliant	Summary Remarks
8.1	To the extent applicable to the developer, acceptance testing includes a comprehensive test in the operating environment(s).	Uncertain.	A verifiable, written set of SQA plans and procedures, which would include acceptance-testing documentation, is lacking for ALOHA.
8.2	To the extent applicable to the developer acceptance testing was performed prior to approval of the computer program for use.	Uncertain.	See Criterion 8.1 summary remarks.
8.3	The acceptance testing comprehensively evaluates software performance against specified software requirements. To the extent applicable to the developer software validation was performed to ensure that the installed software product satisfies the specified software requirements.	Uncertain.	See Criterion 8.1 summary remarks.
8.4	Acceptance-testing documentation includes results of the execution of test cases for system installation and integration, user instructions (Refer to Requirement 7 above), and documentation of the acceptance of the software for operational use.	Partial.	The user's manual includes four sample problems that can serve as test cases. Guidance is given for each required input for each test case. Results are also given for each test case that can be compared to user-generated results. These cases can be viewed as providing users and user groups with a mechanism for deciding if the ALOHA software is correctly installed and functioning properly.

4.8.2 Sources and Method of Review

Documentation that was distributed with the software package (plus information on ALOHA websites) and additional documentation that was supplied by the code developers for this effort were used as the primary bases for response to this requirement.

4.8.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which include acceptance-testing documentation for ALOHA should be addressed.

4.8.4 Recommendations

Recommendations related to this topical area are provided as follows:

- Formal documentation of the acceptance testing process as it relates to ALOHA 5.2.3 will be needed for ALOHA to meet all prerequisites for the DOE toolbox.

4.9 Topical Area 9 Assessment: Configuration Control

This area corresponds to the requirement entitled Configuration Control in Table 3-3 of (DOE 2003e).

4.9.1 Criterion Specification and Result

Table 4.9-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.9-1 — Subset of Criteria for Configuration Control Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
9.1	For the developers, the methods used to control, uniquely identify, describe, and document the configuration of each version or update of a computer program (for example, source, object, back-up files) and its related documentation (for example, software design requirements, instructions for computer program use, test plans, and results) are described in implementing procedures.	Partial.	Discussions with the code developers indicate that software is managed by the program manager (currently Mark Miller) and under the direct control of the project manager (currently Jerry Muhasky). The project manager implements all the changes to the source code and maintains the code on his computer, using password protection to control access. The computer is backed up on a NOAA server as well as CD copies that are stored in secure off site locations. The code developers indicate that there are plans to formally document configuration control procedures.
9.2	Implementing procedures meet applicable criteria for configuration identification, change control and configuration status accounting.	Partial.	See Criterion 9.1 summary remarks.

4.9.2 Sources and Method of Review

The requirement was assessed largely through discussions with the code developers.

4.9.3 Software Quality-Related Issues or Concerns

There are no substantive SQA issues or concerns relative to this requirement.

4.9.4 Recommendations

Recommendations related to this topical area are provided as follows:

- It is recommended that the code developers follow through with plans to formally document configuration control procedures.

4.10 Topical Area 10 Assessment: Error Impact

This area corresponds to the requirement entitled Error Impact in Table 3-3 of (DOE 2003e).

4.10.1 Criterion Specification and Result

Table 4.10-1 lists the subset of criteria reviewed for this topical area and summarizes the findings.

Table 4.10-1 — Subset of Criteria for Error Impact Topic and Results

Criterion Number	Criterion Specification	Compliant	Summary Remarks
10.1	The problem reporting and corrective action process used by the software developing organization addresses the appropriate requirements of the developing organization's corrective action system, and are documented in implementing procedures.	Partial.	NOAA controls the error notification and corrective actions process. An error notification and corrective action document was not available, making a thorough evaluation not possible.
10.2	Method(s) for documenting (Error Notification and Corrective Action Report), evaluating, and correcting software problems describe the evaluation process for determining whether a reported problem is an error.	Partial.	Upgrades are made to code has errors are discovered, frequently by users. The program manager evaluates whether a reported error reflects an error in the code. An error notification and corrective action document was not available, making a thorough evaluation not possible.
10.3	Method(s) for documenting (Error Notification and Corrective Action Report), evaluating, and correcting software problems define the responsibilities for disposition of the problem reports, including notification to the originator of the results of the evaluation..	Partial.	If the program manager determines that an error exists in the code, the project manager responsible for the portion of the code in question as well as the originator of the error report are notified. An error notification and corrective action document was not available, making a thorough evaluation not possible.
10.4	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the error relates to appropriate software engineering elements.	Uncertain.	An error notification and corrective action document was not available, making a thorough evaluation not possible.
10.5	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the error impacts past and present use of the computer program	Partial	The lead programmer/project manager makes changes in the source code if required to address the error and incorporates the new source code into the next release. If the error represents a safety issue then a new release is made as soon as possible. Copies of the incrementally changed code are kept as historical documentation. An error notification and corrective action document was not

Criterion Number	Criterion Specification	Compliant	Summary Remarks
			available, making a thorough evaluation not possible.
10.6	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the corrective action impacts previous development activities	Partial.	See Criterion 10.5 summary remarks.
10.7	When a problem is determined to be an error, then action to document, evaluate and correct, as appropriate, is provided for handling how the users are notified of the identified error, its impact; and how to avoid the error, pending implementation of corrective actions.	Uncertain.	An error notification and corrective action document was not available, making a thorough evaluation not possible.

4.10.2 Sources and Method of Review

The requirement was assessed largely through discussions with the code developers. If a user detects a problem with the ALOHA software they can report these problems to the development team through the following methods: a problem reporting form at http://www.epa.gov/ceppo/cameo/bugform.htm, or contacting the CAMEO Specialist (ORR.cameo@noaa.gov) from http://response.restoration.noaa.gov/cameo/intro.html

4.10.3 Software Quality-Related Issues or Concerns

Lack of a verifiable, written set of SQA plans and procedures, which includes error notification and corrective action report, for ALOHA should be addressed.

4.10.4 Recommendations

Recommendations related to this topical area are provided as follows:

- Formal documentation of the error notification and corrective action process as it relates to ALOHA 5.2.3 will be needed for ALOHA to meet all prerequisites for the DOE toolbox.

4.11 Training Program Assessment

The software developer's does not have a published training program available for review. However, discussions with the software developer indicate that there is an active and frequent training program presented nationally on ALOHA/CAMEO.

Discussions are ongoing for the software developer to provide training at the Energy Facility Contractors Group (EFCOG) conferences. The winter session is during the Safety Basis Subgroup meeting and the summer session is the larger Safety Analysis Working Group, and historically has included training workshops.

4.12 Software Improvements

A new version of ALOHA, namely ALOHA 5.3, was released in March 2004 just prior to the issuance of this report. The main changes to the program are as follows according to the code developer:

- Windows source code was updated to a 32-bit application
- footprint output (i.e., concentration contour plot) can now be displayed with up to three level-of-concern concentrations) simultaneously
- evaporation algorithm was updated

- capability to model the evaporation from puddles of five aqueous chemical solutions was added
- chemical library was updated

In general according to the code developers, upgrades to ALOHA occur when features are requested by an outside agency (i.e. EPA), or internal discussion sees the benefit of a feature, and then the following steps are implemented. Once funding has been allocated to accomplish the upgrade, the ALOHA program manager assigns the upgrade to the personnel best suited to handle it. Once the team member has completed the development of the upgrade, the project manager ensures the source code is updated. Testing of the new version is completed and documented before the software version is updated and ready for release over the Internet.

It is estimated that a concentrated program to upgrade the SQA pedigree of ALOHA to be compliant with the ten criteria discussed here would require fourteen to sixteen full-time equivalent (FTE)-months. Technical review of the chemical databases associated with this software is assumed to have been performed, and is not included in the level-of-effort estimate.

5.0 Conclusion

The gap analysis for Version 5.2.3 of the ALOHA software, based on a set of requirements and criteria compliant with NQA-1, has been completed. Of the ten SQA requirements for existing software at the Level B classification (important for safety analysis but whose output is not applied without further review), two requirements are met at acceptable level, i.e., *Classification* (1) and *User Instructions* (7). A third requirement, *Configuration Control* (9), is partially met. Improvement actions are recommended for ALOHA to fully meet *Configuration Control* (9) criteria and the remaining seven requirements. This evaluation outcome is deemed acceptable because: (1) ALOHA 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 ALOHA is limited to those analytic applications for which the software is intended.

Suggested remedial actions for this software would warrant upgrading software documents. The complete list of revised 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 Manual.

As part of this effort, the draft NOAA theoretical description memorandum for ALOHA 5.0 (Reynolds, 1992), which is the main source of information for technical information, should be updated for recent upgrades, technically reviewed, and issued as final.

It is estimated that a concentrated program to upgrade the SQA pedigree of ALOHA to be compliant with the ten criteria discussed here would require fourteen to sixteen full-time equivalent (FTE)-months. Technical review of the chemical databases associated with this software is assumed to have been performed, and is not included in the level-of-effort estimate.

A new version of ALOHA, namely ALOHA 5.3, was released in March 2004 just prior to the issuance of this report. It is recommended that this version be evaluated relative to the software improvement and baseline document recommendations, as well as the full set of SQA criteria discussed in this report. If this version is found to be satisfactory, it should replace version 5.2.3 as the designated version of the software for the toolbox.

It was determined that the ALOHA 5.2.3 code does meet its intended function for use in supporting documented safety analysis. However, as with all safety-related software, users should be aware of current limitations and capabilities of the software for supporting safety analysis. Informed use of the code can be assisted by appropriate use of current ALOHA documentation prepared by NOAA and the ALOHA code guidance report for DOE safety analysts, *ALOHA Computer Code Application Guidance for Documented Safety Analysis*, (DOE, 2004). Furthermore, while SQA improvement actions are recommended for ALOHA, no evidence has been found of programming, logic, or other types of software errors in ALOHA 5.2.3 that have led to non-conservatisms in nuclear facility operations, or in the identification of facility controls.

6.0 Acronyms and Definitions

ACRONYMS:

AEC	Atomic Energy Commission
ALOHA	Areal Locations of Hazardous Atmospheres (designated toolbox software)
ANS	American Nuclear Society
ANSI	American National Standards Institute
ASME	American Society of Mechanical Engineers
CCPS	Center for Chemical Process Safety
CD	Compliance Decision
CFAST	Consolidated Fire and Smoke Transport Model (designated toolbox software)
CFD	Computational Fluid Dynamics
CFR	Code of Federal Regulations
CSARP	Cooperative Severe Accident Research Program
DCF	Dose Conversion Factor
DIR	Defect Investigation Report
DNFSB	Defense Nuclear Facilities Safety Board
DoD	Department of Defense
DOE	Department of Energy
DSA	Documented Safety Analysis
EFCOG	Energy Facility Contractors Group
EH	DOE Office of Environment, Safety and Health
EIA	Electronic Industries Alliance
EM	DOE Office of Environmental Management
EPIcode	Emergency Prediction Information code (designated toolbox software)
EPRI	Electric Power Research Institute
FTE	Full-time equivalent
GENII	Generalized Environmental Radiation Dosimetry Software System - Hanford Dosimetry System (Generation II) (designated toolbox software)
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IP	Implementation Plan
ISO	International Organization for Standardization
MACCS2	MELCOR Accident Consequence Code System 2 (designated toolbox software)
MELCOR	Methods for Estimation of Leakages and Consequences of Releases (designated toolbox software)
NNSA	National Nuclear Security Administration
NRC	Nuclear Regulatory Commission
OCRWM	Office of Civilian Radioactive Waste Management
PSA	Probabilistic Safety Analysis (or Assessment)
QAP	Quality Assurance Program (alternatively, Plan)
RSICC	Radiation Safety Information Computational Center
SNL	Sandia National Laboratories
SQA	Software Quality Assurance
SRS	Savannah River Site
V&V	Verification and Validation
WSRC	Westinghouse Savannah River Company

DEFINITIONS:

The following definitions are taken from the Implementation Plan. References in brackets following definitions indicate the original source, when not the Implementation Plan.

Acceptance Testing — [NQA-1] The process of exercising or evaluating a system or system component by manual or automated means to ensure that it satisfies the specified requirements and to identify differences between expected and actual results in the operating environment.

Central Registry — An organization designated to be responsible for the storage, control, and long-term maintenance of the Department's safety analysis "toolbox codes." The central registry may also perform this function for other codes if the Department determines that this is appropriate.

Classification (Level of Software) — Determination of the level of software quality assurance associated with a computer code commensurate with the importance of the software application. For the toolbox codes, classification level is determined as described in Appendix A of: "Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes".

Commercial Grade Item — An item satisfying a), b), and c) below:

(a) Not subject to design or specification requirements that are unique to nuclear facilities;
(b) Used in applications other than nuclear facilities;
(c) Ordered from the manufacturer/supplier on the basis of specifications set forth in the manufacturer's published product description (for example, catalog). [IEEE Std. 7-4.3.2-1993]

Computer Code — A set of instructions that can be interpreted and acted upon by a programmable digital computer (also referred to as a module or a computer program).

Configuration Item — A collection of hardware or software elements treated as a unit for the purpose of configuration control. [NQA-1]

Configuration Management —The process that controls the activities, and interfaces, among design, construction, procurement, training, licensing, operations, and maintenance to ensure that the configuration of the facility is established, approved and maintained. (Software specific): The process of identifying and defining the configuration items in a system (i.e., software and hardware), controlling the release and change of these items throughout the system's life cycle, and recording and reporting the status of configuration items and change requests. [NQA-1]

Control Point —A point in the software life cycle at which specified agreements or control (typically a test or review) are applied to the software configuration items being developed, e.g., an approved baseline or release of a specified document or computer program. [NQA-1]

Commercial Grade Dedication —A process of evaluating (which includes testing) and accepting commercial grade items to obtain adequate confidence of their suitability for safety application. [IEEE Std. 7-4.3.2-1993]

Data Library — A data file for use with an executable code that is created and maintained by the controlling organization and is not intended for modification by the user.

Dedication (of Software) — The evaluation of software not developed under utilizing organization existing QA plans and procedures (or not developed under NQA-1 standards). The evaluation determines and asserts the software's compliance with NQA-1 quality standards and its readiness for use in specific applications. (Typically applies to commercially available software.) The utilizing organization reviews the intended software application sufficiently to determine the critical functions that provide evidence of the software's suitability for use. Once the critical functions have been established, methods are defined to verify critical function adequacy and provide verifiable acceptance criteria. Acceptable dedication methods are implemented and required documentation is prepared.

Design Requirements — Description of the methodology, assumptions, functional requirements, and technical requirements for a software system.

Discrepancy — The failure of software to perform according to its documentation.

Error —A condition deviating from an established base line, including deviations from the current approved computer program and its baseline requirements. [NQA-1]

Executable Code — The user form of a computer code. For programs written in a compilable programming language, the compiled and loaded program. For programs written in an interpretable programming language, the source code.

Firmware — The combination of a hardware device and computer instructions and data that reside as read-only software on that device. [IEEE Standard 610.12-1990]

Gap Analysis — Evaluation of the Software Quality Assurance attributes of specific computer software against identified criteria.

Independent Verification and Validation (IV&V) — Verification and validation performed by an organization that is technically, managerially, and financially independent of the development organization.

Nuclear Facility — A reactor or a nonreactor nuclear facility where an activity is conducted for or on behalf of DOE and includes any related area, structure, facility, or activity to the extent necessary to ensure proper implementation of the requirements established by 10 CFR 830. [10 CFR 830]

Object Code — A computer code in its compiled form. This applies only to programs written in a compilable programming language.

Operating Environment — A collection of software, firmware, and hardware elements that provide for the execution of computer programs. [NQA-1]

Safety Analysis and Design Software — Computer software that is not part of a structure, system, or component (SSC) but is used in the safety classification, design, and analysis of nuclear facilities to ensure proper accident analysis of nuclear facilities; proper analysis and design of safety SSCs; and proper identification, maintenance, and operation of safety SSCs.

Safety Analysis Software Group (SASG) — A group of technical experts formed by the Deputy Secretary in October 2000 in response to Technical Report 25 issued by the Defense

Nuclear Facilities Safety Board (DNFSB). This group was responsible for determining the safety analysis and instrument and control (I&C) software needs to be fixed or replaced, establishing plans and cost estimates for remedial work, providing recommendations for permanent storage of the software and coordinating with the Nuclear Regulatory Commission on code assessment as appropriate.

Safety-Class Structures, Systems, and Components (SC SSCs) — SSCs, including portions of process systems, whose preventive and mitigative function is necessary to limit radioactive hazardous material exposure to the public, as determined from the safety analyses. [10 CFR 830]

Safety-Significant Structures, Systems, and Components (SS SSCs) — SSCs which are not designated as safety-class SSCs, but whose preventive or mitigative function is a major contributor to defense in depth and/or worker safety as determined from safety analyses. [10 CFR 830] As a general rule of thumb, SS SSC designations based on worker safety are limited to those systems, structures, or components whose failure is estimated to result in prompt worker fatalities, serious injuries, or significant radiological or chemical exposure to workers. The term serious injuries, as used in this definition, refers to medical treatment for immediately life-threatening or permanently disabling injuries (e.g., loss of eye, loss of limb). The general rule of thumb cited above is neither an evaluation guideline nor a quantitative criterion. It represents a lower threshold of concern for which an SS SSC designation may be warranted. Estimates of worker consequences for the purpose of SS SSC designation are not intended to require detailed analytical modeling. Consideration should be based on engineering judgment of possible effects and the potential added value of SS SSC designation. [DOE G 420.1-1]

Safety Software — Includes both safety system software and safety analysis and design software.

Safety Structures, Systems, and Components (SSCs) — The set of safety-class SSCs and safety-significant SSCs for a given facility. [10 CFR 830]

Safety System Software — Computer software and firmware that performs a safety system function as part of a structure, system, or component (SSC) that has been functionally classified as Safety Class (SC) or Safety Significant (SS). This also includes computer software such as human-machine interface software, network interface software, programmable logic controller (PLC) programming language software, and safety management databases that are not part of an SSC but whose operation or malfunction can directly affect SS and SC SSC function.

Sample Input — Input data for a designated sample problem which is maintained by the controlling organization for distribution to users.

Software — Computer programs, operating systems, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. [IEEE Std. 610.12-1990]

Software Design Verification —The process of determining if the product of the software design activity fulfills the software design requirements. [NQA-1]

Software Development Cycle —The activities that begin with the decision to develop a software product and end when the software is delivered. The software development cycle typically includes the following activities:

(a) Software design requirements;

(b) Software design;
(c) Implementation;
(d) Test; and sometimes
(e) Installation. [NQA-1]

Software Engineering — The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software; also: the study of these applications. [NQA-1]

Software Life Cycle —The activities that comprise the evolution of software from conception to retirement. The software life cycle typically includes the software development cycle and the activities associated with operation, maintenance, and retirement. [NQA-1]

Source Code — A computer code in its originally coded form, typically in text file format. For programs written in a compilable programming language, the uncompiled program.

System Software —Software designed to enable the operation and maintenance of a computer system and its associated computer programs. [NQA-1]

Test Case —A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement. [NQA-1]

Test Case Input — Input data for a test case used to verify a modification to a module or a data library.

Test Plan (Procedure) —A document that describes the approach to be followed for testing a system or component. Typical contents identify the items to be tested, tasks to be performed, and responsibilities for the testing activities. [NQA-1]

Testing —An element of verification for the determination of the capability of an item to meet specified requirements by subjecting the item to a set of physical, chemical, environmental, or operating conditions. [NQA-1]

Testing (Software) —The process of

(a) Operating a system (i.e., software and hardware) or system component under specified conditions;

(b) Observing and recording the results; and

(c) Making an evaluation of some aspect of the system (i.e., software and hardware) or system component; in order to verify that it satisfies specified requirements and to identify errors. [NQA-1]

Toolbox Codes — A small number of standard computer models (codes) supporting DOE safety analysis, having widespread use, and of appropriate qualification that are maintained, managed, and distributed by a central source. Toolbox codes meet minimum quality assurance criteria. They may be applied to support 10 CFR 830 DSAs provided the application domain and input parameters are valid. In addition to public domain software, commercial or proprietary software may also be considered. In addition to

safety analysis software, design codes may also be included if there is a benefit to maintain centralized control of the codes [modified from DOE N 411.1].

User Manual — A document that presents the information necessary to employ a system or component to obtain desired results. Typically described are system or component capabilities, limitations, options, permitted inputs, expected outputs, possible error messages, and special instructions. Note: A user manual is distinguished from an operator manual when a distinction is made between those who operate a computer system (mounting tapes, etc.) and those who use the system for its intended purpose. Syn: User Guide. [IEEE 610-12]

Validation — Assurance that a model as embodied in a computer code is a correct representation of the process or system for which it is intended. This is usually accomplished by comparing code results to either physical data or a validated code designed to perform the same type of analysis. [IEEE-610.12]: The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements. Contrast with: **verification.**

Verification — Assurance that a computer code correctly performs the operations specified in a numerical model or the options specified in the user input. This is usually accomplished by comparing code results to a hand calculation or an analytical solution or approximation. [IEEE-610.12]: (1) The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. Contrast with: **validation**. (2) Formal proof of program correctness.

7.0 References

CFR Code of Federal Regulations (10 CFR 830). 10 CFR 830, Nuclear Safety Management Rule.

DNFSB Defense Nuclear Facilities Safety Board, (2000). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, Technical Report DNFSB/TECH-25, (January 2000).

DNFSB Defense Nuclear Facilities Safety Board, (2002). *Recommendation 2002-1, Quality Assurance for Safety-Related Software,* (September 2002).

DOE, U.S. Department of Energy (2000a). *Appendix A, Evaluation Guideline,* DOE-STD-3009-94, *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports* (January 2000).

DOE, U.S. Department of Energy (2000b). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, DOE Response to TECH-25, Letter and Report, (October 2000).

DOE, U.S. Department of Energy (2002). *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports*, DOE-HDBK-3010-94, Change Notice 2 (April 2002).

DOE, U.S. Department of Energy (2003a). *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).

DOE, U.S. Department of Energy (2003b). *Designation of Initial Safety Analysis Toolbox Codes*, Letter, (March 28, 2003).

DOE, U.S. Department of Energy (2003c). *Assessment Criteria and Guidelines for Determining the Adequacy of Software Used in the Safety Analysis and Design of Defense Nuclear Facilities*, Report, CRAD-4.2.4-1, Rev 0, (August 27 2003).

DOE, U.S. Department of Energy (2003d). *Software Quality assurance Improvement Plan: Format and Content For Code Guidance Reports*, Revision A (draft), Report, (August 2003).

DOE, U.S. Department of Energy (2003e). *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes*, Revision 1, (November 2003).

DOE, U.S. Department of Energy (2004). ALOHA Computer Code Application Guidance for Documented Safety Analysis, (May 2004).

M. Evans (1993). *ALOHA User's and ARCHIE: A Comparison*, Report No. HAZMAT 93-2, Office of Ocean and Resources Conservation and Assessment of the National Oceanic and Atmospheric Administration (NOAA), Seattle, WA.

M. Evans (1994). *Quality Assurance of ALOHA*, Draft Report, Office of Ocean and Resources Conservation and Assessment of the National Oceanic and Atmospheric Administration (NOAA), Seattle, WA.

FEMA (1989). *Handbook of Chemical Hazard Analysis Procedures*, (ARCHIE Manual), Federal Emergency Management Agency (FEMA), U. S. Department of Transportation (USDOT), and U.S. Environmental Protection Agency (USEPA) (1989), Washington, D.C.

A. D. Little (1988). *CHEMS-PLUS (Enhanced Chemical Evaluation Hazard Evaluation Methodologies) Reference Manual*, Version 1.0, Cambridge, MA.

NOAA (1998). *ALOHA Quality Assurance*, National Oceanic and Atmospheric Administration (NOAA), http://www.nwn.noaa.gov/sites/hazmat/cameo/alotech/quality.html.

NOAA (1999a) and EPA. *ALOHA User's Manual*, Office of Response and Restoration of the National Oceanic and Atmospheric Administration (NOAA) and Chemical Emergency Preparedness and Prevention Office (CEPPO) of the U.S. Environmental (EPA), Seattle, WA.

NOAA (1999b) and EPA. *ALOHA 5.2.3 Online Help*, Office of Response and Restoration of the National Oceanic and Atmospheric Administration (NOAA) and Chemical Emergency Preparedness and Prevention Office (CEPPO) of the U.S. Environmental (EPA), Seattle, WA.

R. M. Reynolds (1992). *ALOHA Theoretical Description, Draft Technical Memorandum*, NOS ORCA-65 Hazardous Materials Response and Assessment Division (HMRAD) of the National Oceanic and Atmospheric Administration (NOAA), Seattle, WA.

T. Spicer and J. Havens (1989). User's Guide for the DEGADIS 2.1 Dense Gas Dispersion Model, U.S. EPA, EPA-450/4-89-019, Cincinnati.

Appendices

Appendix	Subject
A	Software Information Template

APPENDIX A.— SOFTWARE INFORMATION TEMPLATE

Information Form

Development and Maintenance of Designated Safety Analysis Toolbox Codes

The following summary information in Table 2 should be completed to the level that is meaningful – enter N/A if not applicable. See Appendix A for an example of the input to the table prepared for the MACCS2 code.

Table 2. Summary Description of Subject Software

Table 2. Summary Description of Subject Software	
Type	**Specific Information**
Code Name	
Version of the Code	
Developing Organization and Sponsor Information	
Auxiliary Codes	
Software Platform/Portability	
Coding and Computer(s)	
Technical Support Point of Contact	
Code Procurement Point of Contact	
Code Package Label/Title	
Contributing Organization(s)	
Recommended	1.

Table 2. Summary Description of Subject Software	
Type	**Specific Information**
Documentation - Supplied with Code Transmittal upon Distribution or Otherwise Available	2. 3. 4. 5.
Input Data/Parameter Requirements	
Summary of Output	
Nature of Problem Addressed by Software	
Significant Strengths of Software	
Known Restrictions or Limitations	
Preprocessing (set-up) time for Typical Safety Analysis Calculation	
Execution Time	
Computer Hardware Requirements	
Computer Software Requirements	
Other Versions Available	

Table 3. Point of Contact for Form Completion

Individual(s) completing this information form: Name: Organization: Telephone: Email: Fax:	

1. Software Quality Assurance Plan

The software quality assurance plan for your software may be either a standalone document, or embedded in other documents, related procedures, QA assessment reports, test reports, problem reports, corrective actions, supplier control, and training package.

1.a For this software, identify the governing Software Quality Assurance Plan (SQAP)? [Please submit a PDF of the SQAP, or send hard copy of the SQAP[4]]

1.b What software quality assurance industry standards are met by the SQAP?

1.c What federal agency standards were used, if any, from the sponsoring organization?

1.d Has the SQAP been revised since the current version of the Subject Software was released? If so, what was the impact to the subject software?

1.e Is the SQAP proceduralized in your organization? If so, please list the primary procedures that provide guidance.

Guidance for SQA Plans:

Requirement 2 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a)
ASME NQA-1 2000 Section 200

[4] Notify Kevin O'Kula of your intent to send hard copies of requested reports and shipping will be arranged.

IEEE Standard 730, *IEEE Standard for Software Quality Assurance Plans.*
IEEE Standard 730.1, *IEEE Guide for Software Quality Assurance Planning.*

2. Software Requirements Description

The software requirements description (SRD) should contain functional and performance requirements for the subject software. It may be contained in a standalone document or embedded in another document, and should address functionality, performance, design constraints, attributes and external interfaces.

2.a **For this software, was a software requirements description documented with the software sponsor?** [If available, please submit a PDF of the Software Requirements Description, or include hard copy with transmittal of SQAP]

2.b **If a SRD was not prepared, are there written communications that indicate agreement on requirements for the software? Please list other sources of this information if it is not available in one document.**

Guidance for Software Requirements Documentation:

Requirement 5 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 401
IEEE Standard 830, *Software Requirements Specifications*

3. Software Design Documentation

The software design documentation (SDD) depicts how the software is structured to satisfy the requirements in the software requirements description. It should be defined and maintained to ensure that software will serve its intended function. The SDD for the subject software may be contained in a standalone document or embedded in another document.

The SDD should provide the following:

- Description of the major components of the software design as they relate to the software requirements,
- Technical description of the software with respect to the theoretical basis, mathematical model, control flow, data flow, control logic, and data structure,
- Description of the allowable or prescribed ranges of inputs and outputs,
- Design described in a manner suitable for translating into computer coding, and
- Computer program listings (or suitable references).

3.a **For the subject software, was a software design document prepared, or were its constituents parts covered elsewhere?** [If available, please submit a PDF of the Software Design Document, or include hard copy with transmittal of SQAP]

3.b If the intent of the SDD information is satisfied in other documents, provide the appropriate references (document number, section, and page number).

Guidance for Software Design Documentation:

Requirement 6 - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
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*
IEEE Standard 1012, *IEEE Standard for Software Verification and Validation;*
IEEE Standard 1012a, *IEEE Standard for Software Verification and Validation - Supplement to 1012*

4. Software User Documentation

Software User Documentation is necessary to assist the user in installing, operating, managing, and maintaining the software, and to ensure that the software satisfies user requirements. At minimum, the documentation should describe:

- The user's interaction with the software
- Any required training
- Input and output specifications and formats, options
- Software limitations
- Error message identification and description, including suggested corrective actions to be taken to correct those errors, and
- Other essential information for using the software.

4.a For the subject software, has Software User Documentation been prepared, or are its constituents parts covered elsewhere? [If available, please submit a PDF of the Software User Documentation, or include a hard copy with transmittal of SQAP]

4.b If the intent of the Software User Documentation information is satisfied in other documents, provide the appropriate references (document number, section, and page number).

4.c Training – How is training offered in correctly running the subject software? Complete the appropriate section in the following:

Type	Description	Frequency of training
Training Offered to		

Type	Description	Frequency of training
User Groups as Needed		
Training Sessions Offered at Technical Meetings or Workshops		
Training Offered on Web or Through Video Conferencing		
Other Training Modes		
Training Not Provided		

Guidance for Software User Documentation:

Requirement 9 – SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 203
IEEE Standard 1063, *IEEE Standard for Software User Documentation*

5. Software Verification &Validation Documentation (Includes Test Reports)

Verification and Validation (*V&V*) documentation should confirm that a software V&V process has been defined, that V&V has been performed, and that related documentation is maintained to ensure that:

(a) The software adequately and correctly performs all intended functions, and
(b) The software does not perform any unintended function.

The software V&V documentation, either as a standalone document or embedded in other documents and should describe:

- The tasks and criteria for verifying the software in each development phase and validating it at completion,
- Specification of the hardware and software configurations pertaining to the software V&V
- Traceability to both software requirements and design
- Results of the V&V activities, including test plans, test results, and reviews (also see 5.b below)
- A summary of the status of the software's completeness
- Assurance that changes to software are subjected to appropriate V&V,
- V&V is complete, and all unintended conditions are dispositioned before software is approved for use, and
- V&V performed by individuals or organizations that are sufficiently independent.

5.a For the subject software, identify the V&V Documentation that has been prepared. [If available, please submit a PDF of the Verification and Validation Documentation, or include a hard copy with transmittal of SQAP]

5.b If the intent of the V&V Documentation information is satisfied in one or more other documents, provide the appropriate references (document number, section, and page number). For example, a "Test Plan and Results" report, containing a plan for software testing, the test results, and associated reviews may be published separately.

5.c Testing of software: What has been used to test the subject software?

☐ Experimental data or observations
☐ Standalone calculations
☐ Another validated software
☐ Software is based on previously accepted solution technique

Provide any reports or written documentation substantiating the responses above.

Guidance for Software Verification & Validation, and Testing Documentation:

Requirement 6 – *Design Phase* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
Requirement 8 – *Testing Phase* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
Requirement 10 – *Acceptance Test* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))

ASME NQA-1 2000 Section 402 (Note: Some aspects of verification may be handled as part of the Design Phase).
ASME NQA-1 2000 Section 404 (Note: Aspects of validation may be handled as part of the Testing Phase).
IEEE Standard 1012, *IEEE Standard for Software Verification and Validation;*
IEEE Standard 1012a, *IEEE Standard for Software Verification and Validation – Supplement to 1012*
IEEE Standard 829, *IEEE Standard for Software Test Documentation.*
IEEE Standard 1008, *Software Unit Testing*

6. Software Configuration Management (SCM)

A process and related documentation for SCM should be defined, maintained, and controlled.

The appropriate documents, such as project procedures related to software change controls, should verify that a software configuration management process exists and is effective.

The following points should be covered in SCM document(s):

- A Software Configuration Management Plan, either in standalone form or embedded in another document,
- Configuration management data such as software source code components, calculational spreadsheets, operational data, run-time libraries, and operating systems,
- A configuration baseline with configuration items that have been placed under configuration control,
- Procedures governing change controls,
- Software change packages and work packages to demonstrate that (1) possible impacts of software modifications are evaluated before changes are made, (2) various software system products are examined for consistency after changes are made, and (3) software is tested according to established standards after changes have been made.

6.a **For the subject software, has a Software Configuration Management Plan been prepared, or are its constituent parts covered elsewhere?** [If available, please submit a PDF of the Software Configuration Management Plan and related procedures, or include hard copies with transmittal of SQAP].

6.b **Identify the process and procedures governing control and distribution of the subject software with users.**

6.c **Do you currently interact with a software distribution organization such as the Radiation Safety Information Computational Center (RSICC)?**

6.d **A Central Registry organization, under the management and coordination of the Department of Energy's Office of Environment, Safety and Health (EH), will be responsible**

for the long-term maintenance and control of the safety analysis toolbox codes for DOE safety analysis applications. Indicate any questions, comments, or concerns on the Central Registry's role and the maintenance of the subject software.

Guidance for Software Configuration Management Plan Documentation:

Requirement 12 – *Configuration Control* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 203
IEEE Standard 828, *IEEE Standard for Software Configuration Management Plans.*

7. Software Problem Reporting and Corrective Action

Software problem reporting and corrective action documentation help ensure that a formal procedure for problem reporting and corrective action development for software errors and failures is established, maintained, and controlled.

A Software Error Notification and Corrective Action Report, procedure, or similar documentation, should be implemented to report, track, and resolve problems or issues identified in both software items, and in software development and maintenance processes. Documentation should note specific organizational responsibilities for implementation. Software problems should be promptly reported to affected organizations, along with corrective actions. Corrective actions taken ensure that:

- Problems are identified, evaluated, documented, and, if required, corrected,
- Problems are assessed for impact on past and present applications of the software by the responsible organization,
- Corrections and changes are executed according to established change control procedures, and
- Preventive actions and corrective actions results are provided to affected organizations.

Identify documentation specific to the subject software that controls the error notification and corrective actions. [If available, please submit a PDF of the Error Notification and Corrective Action Report documentation for the subject software (or related procedures). If this is not available, include hard copies with transmittal of SQAP].

7.aProvide examples of problem/error notification to users and the process followed to address the deficiency. Attach files as necessary.

7.bProvide an assessment of known errors or defects in the subject software and the planned action and time frame for correction.

Category of Error or Defect	Corrective Action	Planned schedule for correc
Major		

Minor		

7.cIdentify the process and procedures governing communication of errors/defects related to the subject software with users.

Guidance for Error/Defect Reporting and Corrective Action Documentation:

Requirement 13 *Error Impact* - SQA Procedures/Plans (Table 3-2 of SQA Plan/Criteria (DOE, 2003a))
ASME NQA-1 2000 Section 204
IEEE Standard 1063, *IEEE Standard for Software User Documentation*

8. Resource Estimates

If one or more plans, documents, or sets of procedures identified in parts one (1) through seven (7) do not exist, please provide estimates of the resources (full-time equivalent (40-hour) weeks, FTE-weeks) and the duration (months) needed to meet the specific SQA requirement.

Enter estimate in Table 4 only if specific document has not been prepared, or requires revision.

Table 4. Resource and Schedule for SQA Documentation

Plan/Document/Procedure	Resource Estimate (FTE-weeks)	Duration of Activity (months)
1. Software Quality Assurance Plan		
2. Software Requirements Document		
3. Software Design Document		
4. Test Case Description and Report		
5. Software Configuration and Control		
6. Error Notification and Corrective Action Report		
7. User's Instructions (User's Manual)		
8. Other SQA Documentation		

Comments or Questions:

9. **Software Upgrades**

Describe modifications planned for the subject software.

Technical Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

User Interface Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Software Engineering Improvements

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Other Planned Modifications

Priority	Description of Change	Resource Estimate (FTE-weeks)
1.		
2.		
3.		
4.		
5.		

Thank you for your input to the SQA upgrade process. Your experience and insights are critical towards successfully resolving the issues identified in DNFSB Recommendation 2002-1.

REFERENCES

CFR Code of Federal Regulations (CFR). 10 CFR 830, Nuclear Safety Management Rule.

DNFSB Defense Nuclear Facilities Safety Board (2000). *Quality Assurance for Safety-Related Software at Department of Energy Defense Nuclear Facilities*, Technical Report DNFSB/TECH-25, (January 2000).

DNFSB Defense Nuclear Facilities Safety Board (2002). *Recommendation 2002-1, Quality Assurance for Safety-Related Software* , (September 2002).

DOE, U.S. Department of Energy (2000a). *Appendix A, Evaluation Guideline,* DOE-STD-3009-94, *Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Safety Reports* (January 2000).

DOE, U.S. Department of Energy (2002). *Selection of Computer Codes for DOE Safety Analysis Applications* (August 2002).

DOE, U.S. Department of Energy (2003). *Implementation Plan for Defense Nuclear Facilities Safety Board Recommendation 2002-1: Quality Assurance for Safety Software at Department of Energy Nuclear Facilities*, Letter (March 13, 2003); Report (February 28, 2003).

DOE, U.S. Department of Energy (2003a). *Software Quality Assurance Plan and Criteria for the Safety Analysis Toolbox Codes,* Interim Report, (September 2003).

DOE/EH, U.S. Department of Energy Office of Environment, Safety and Health (2003), *Designation of Initial Safety Analysis Toolbox Codes*, Letter, (March 28, 2003).