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

TABLES

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 2003e).

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	Compliant	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.