Westinghouse Non-Proprietary Class 3

WCAP-16096-NP-A
Revision 0
(Previously released as CE-CES-195)

May 2003

Software Program Manual For Common Q Systems

SOFTWARE PROGRAM MANUAL

FOR

COMMON Q SYSTEMS

WCAP-16096-NP-A, Revision 0
CE-CES-195-NP-A, Revision 2

Westinghouse Non-Proprietary Class 3

Prepared by: _____ Date: 5/30/03
W. R. Odess-Gillett, Software Engineer

Reviewed by: _____ Date: 5/30/03
M. J. Stofko, Common Q Platform Lead

Approved by: _____ Date: 5-30-03
D. M. Popp, Manager, Regulatory Compliance

Issue Date: May 2003

Total Pages: 173

RECORD OF REVISIONS

REV. NO.	DATE	PAGES INVOLVED	PREPARED BY	REVIEWED BY	APPROVALS
0	May 2003	Original Issue of WCAP doc.	W. R. Odess-Gillett	M. J. Stofko	D. M. Popp

File Name: k:\icee\qp_ped\nrc_top\spm\WCAP-16096-NP-A Rev 0 or CE-CES-195-NP-A Rev 2 May03.doc

File Size: 777728

REVISION ABSTRACT

Revision 0:
This is the original issue of this document. This document was previously released as CE-CES-195. It is being prepared to create the accepted version in accordance with the USNRC Safety Evaluation dated February 24, 2003. The previously released document CE-CES-195, Revision 1 has been modified as follows:

- A document titled, "Changes to the Westinghouse Software Program Manual for Common Q Systems, CE-CES-195-P, Revision 1" was provided to the NRC via LTR-NRC-02-41, dated August 14, 2002. The changes identified in that document have been incorporated into this revision.
- Minor changes were made to clarify some of the process descriptions.
- Correction of typographical errors.
- Minor document format changes were made.
- In accordance with established NRC procedures, the accepted version of the Common Q Topical Report (WCAP-16097-P-A) includes the original Safety Evaluation, two supplemental Safety Evaluations and associated cover letters that were issued by the NRC. This Software Program Manual was treated as an appendix to the Common Q Topical Report during the NRC review and approval process.
- References to the ABB Quality Assurance Manual and procedures were replaced with the Westinghouse Quality Management System and its implementing procedures. Westinghouse letter I&CE-LIC(02)-114 dated 11/14/02 provides a mapping to the applicable implementing procedures. Changing the references to the Westinghouse Quality Management System and its implementing procedures does not in any way change the specific life cycle processes described in the SPM.

TABLE OF CONTENTS

LIST OF EXHIBITS

1. INTRODUCTION

1.1 PURPOSE

Computer software is essential to the design, analysis, operation and control of Common Qualified (Q) systems. This Software Program Manual (SPM) describes the requirements for the software design and development process and for the use of software in Common Q systems. The SPM expands the procedural requirements for computer software in Reference 1.6.4.

The Software Program Manual consists of several basic elements:

1) A Software Safety Plan, which identifies the processes that will reasonably assure that safety-critical software does not have hazards that could jeopardize the health and safety of the public.

2) A Software Quality Assurance Plan (SQAP), which describes the process and practice of developing and using software. The SQAP addresses standards, conventions, reviews, problem reporting and other software quality issues.

3) A Software Verification and Validation Plan, which describes the method of assuring correctness of the software.

4) A Software Configuration Management Plan, which describes the method of maintaining the software in an identifiable state at all times.

5) A Software Operations and Maintenance Plan, which describes software practices after delivery to a customer.

The SPM also discusses Software Management, documentation and other matters related to software design and use.

Exhibit 1-1 shows the relationship of this Software Program Manual to codes, standards and project specific documentation.

1.2 SCOPE

1.2.1 This SPM shall apply to all software and firmware, whether developed in-house, licensed or procured from a commercial vendor, obtained from another organization or otherwise acquired and used in a Common Q system for delivery to a customer.

The Common Q systems and modules are identified as belonging to one of the following classes:

- Protection (safety critical)

  Software whose function is necessary to directly perform RPS control actions, ESFAS control actions, and safe shutdown control actions.

- Important to Safety

  Software whose function is necessary to directly perform alternate protection system control actions or software that is relied on to monitor or test protection functions, or software that monitors plant critical safety functions.

- Important to Availability

  Software that is relied on to maintain operation of plant systems and equipment that are critical to maintaining an operating plant.

- General Purpose

  Software that performs some purpose other than that described in the previous classifications. This software includes tools that are used to develop software in the other classifications, but is not installed in the on-line plant system.

The SPM makes distinctions regarding the methods applied to each of the above classes. Specific parts of the software in a single system may be assigned to different classes. Each part of the software must have an assigned class.

The SPM makes distinctions regarding methods applied to each of the following categories of Common Q software:

- Original, Developed for a Common Q System

- Existing, to be Modified
- Existing, to be used as is

Every Common Q software item is assigned to one of the above categories. Software in several categories may be included in each Common Q system. For example, a typical computer system may rely on:

- An operating system from a commercial supplier that is existing, used as is.
- Some residual code to be updated from a previous project (existing, to be modified)
- New algorithms (originally developed)

This SPM applies to all software used in the development, testing or delivered Common Q systems.

1.2.2 The following software is excluded from the requirements of this SPM:

- Administrative software used for purposes such as ordering, scheduling and project management.
- Commercial applications software for use in database management systems, word processing, and commercially purchased CAD systems. Such applications are Excel, Word, and AutoCAD.

1.3 OVERVIEW

Common Q software developers shall proceed through a software development effort by following the approach described in this manual.

The software developers should first become familiar with the Software Quality Assurance Plan which is a guideline for all activities relating to Common Q software.

The Cognizant Engineering Organization is required to determine the class and category of all software to be used for the Common Q system as described in the SQAP. The Cognizant Engineer identifies the applicable standards which must be followed for those specific classes and categories of software. The software tasks and responsibilities are outlined in the SQAP based upon software classification and category.

Each quality assurance task is described in the SQAP for each software life cycle phase. The narrative description, along with the corresponding Exhibit, assist the Cognizant Engineer in making the required decisions concerning the appropriate tasks to be performed and who is responsible for performing them. In addition, the specific documents which must be produced for each software life cycle phase are discussed in the SQAP. Required documents vary for each software category.

The Software Verification and Validation Plan and the Software Configuration Management Plan describe the details of some of the activities outlined in the SQAP.

Adherence to the Software Verification and Validation Plan will ensure the verification of an accurate translation from one step in the software development process to the next step and the validation that the software product does fulfill the requirements for which the software was developed. The degree of independence required by this plan varies with the software classification. The applicability of the tasks varies with the software category.

The software configuration management plan describes the procedures necessary to maintain the Common Q software in an identifiable state at all times. These procedures do not vary with the software class or category.

Recommendations are made for a Software Operation and Maintenance Plan to be administered by others.

The recommendations for the Software Operations and Maintenance Plan describe the activities necessary to maintain the Common Q software, to remove latent errors, to

respond to new or revised requirements and to adapt the software to changes in operating environments.

The documentation section of this Software Program Manual describes the various documents which are required. The set of required documents for each software class is specified in the Software Quality Assurance Plan.

The problem reporting and corrective action section of the Software Program Manual describes procedures necessary to ensure that all software errors and failures are promptly acted upon and in a uniform manner encompassing all Common Q software. The procedures in this section tie together the requirements of the Software Verification and Validation Plan and the Software Configuration Management Plan.

1.4 GENERAL REQUIREMENTS

1.4.1 The management and control of software applies to computer software and associated documentation developed or used for Common Q applications. Software shall be developed, acquired, procured, controlled, and maintained in accordance with this Software Program Manual. The SPM meets the requirements of Reference 1.6.11 as augmented by Reference 1.6.17.

1.4.2 Software Life Cycle

This Software Program Manual is based on a Software Life Cycle model consistent with Reference 1.6.2 and 1.6.5 which includes the following phases:

- Concept
- Requirements Analysis
- Design
- Implementation or Coding
- Test
- Installation and Checkout
- Operation and Maintenance
- Retirement

1.4.3 Indoctrination and Training

The Cognizant Engineering Organization personnel involved in Common Q software design, development and use shall have documented training in this SPM. Such training records shall be prepared and maintained in accordance with the requirements of Reference 1.6.4.

1.5 ACRONYMS AND DEFINITIONS

Acronyms

ABB	Asea Brown Boveri
ANSI	American National Standards Institute
BTP	Branch Technical Position
CCC	Cross Channel Communications
CDR	Critical Design Review
CEO	Cognizant Engineering Organization
CET	Core Exit Thermocouple
CFR	Code of Federal Regulations
CM	Configuration Management
CMS	Configuration Management System
CPC	Core Protection Calculator
CPU	Central Processing Unit
DESFAS-AC	Digital Engineered Safety Features Actuation System Auxiliary Cabinet
DPPS	Digital Plant Protection System
DT	Design Team
ENM	Existing Software not to be modified
ETBM	Existing Software to be modified
ESFAS	Engineered Safety Features Actuation System
FAT	Factory Acceptance Test
IEEE	Institute of Electrical and Electronics Engineers
I/O	Input/Output
I&C	Instrumentation and Control
LAN	Local Area Network
LCL	Local Coincidence Logic
NPP	Nuclear Power Plant
NQA	Nuclear Quality Assurance
PAMS	Post Accident Monitoring System
PDR	Preliminary Design Review
RPS	Reactor Protection System
RTA	Requirements Traceability Analysis
RTM	Requirements Traceability Matrix
RVL	Reactor Vessel Level
SAT	Site Acceptance Test
SCA	Software Change Authorization
SCM	Software Configuration Management
SCMP	Software Configuration Management Plan
SCR	Software Change Request

SDD	Software Design Description
SM	Subcooled Margin
SOMP	Software Operation and Maintenance Plan
SPM	Software Program Manual
SRS	Software Requirements Specification
SQAP	Software Quality Assurance Plan
SRR	Software Requirements Review
SVVP	Software Verification and Validation Plan
SVVR	Software Verification and Validation Report
TER	Test Exception Report
USNRC	United States Nuclear Regulatory Commission
V&V	Verification and Validation
VT	Verification Team

Definitions

Cognizant Manager	Manager of the cognizant group responsible for control of a software configuration item. The Cognizant Manager may delegate the performance of necessary tasks to other persons but remains responsible for their execution.
Testing	The process of exercising or evaluating a system or system component by manual or automated means, to verify that it satisfies specified requirements or to identify differences between expected and actual results.
Safety Related	Software that is Quality Class 1 (QC-1) as defined in Reference 1.6.4. Protection class software is always safety related, and Important To Safety class software can be safety related. For all Common Q Systems, all software in the AC160 and the QNX-based Flat Panel Display Software are safety related.

Definitions for all other terms used in this document can be found in Reference 1.6.5.

1.6 REFERENCES

Unless stated otherwise, the latest revision is applicable.

1.6.1 Westinghouse Quality Management System

1.6.2 ASME NQA-1-1997, Subpart 2.7, "Quality Assurance Requirements for Nuclear Facility Applications."

1.6.3 Guidance on Evaluation and Acceptance of Commercial Grade Digital Equipment for Nuclear Safety Applications, EPRI TR-106439, October 1996

1.6.4 Nuclear Automation Edition of the Westinghouse Policy and Procedures Manual

1.6.5 IEEE Std 610.12-1990, "Standard Glossary of Software Engineering Terminology."

1.6.6 IEEE Std 830-1993, "Guide to Software Requirements Specifications."

1.6.7 IEEE Std 1016-1987, "Recommended Practice for Software Design Descriptions."

1.6.8 IEEE Std 1012-1986 (Reaff. 1992), "Standard for Software Verification and Validation Plans."

1.6.9 IEEE Std 1063-1987, "Standard for Software User Documentation."

1.6.10 IEEE Std 828-1990, "Standard for Software Configuration Management Plans."

1.6.11 IEEE Std 7-4.3.2-1993, "Application Criteria for Programmable Digital Computer Systems in Safety Systems of Nuclear Power Generating Stations."

1.6.12 IEEE Std 1008-1987, "IEEE Standard for Software Unit Testing."

1.6.13 IEEE Std 730-1989 "Standard for Software Quality Assurance Plans."

1.6.14 IEEE Std 829-1983 (Reaff. 1991), "IEEE Standard for Software Test Documentation."

1.6.15 IEEE Std 1219-1992, "IEEE Standard For Software Maintenance."

1.6.16 IEEE Std 1028-1988, "IEEE Standard For Software Reviews and Audits"

1.6.17 Reg. Guide 1.152, Rev. 1 (1996), "Criteria for Digital Computers in Safety Systems of Nuclear Power Plants."

1.6.18 Reg. Guide 1.168, Rev. 0 (Sept. 1997), "Verification, Validation, Reviews, and Audits for Digital Computer Software used in Safety Systems of Nuclear Power Plants."

1.6.19 Reg. Guide 1.169, Rev. 0 (Sept. 1997), "Configuration Management Plans for Digital Computer Software used in Safety Systems of Nuclear Power Plants."

1.6.20 Reg. Guide 1.170, Rev. 0 (Sept. 1997), "Software Test Documentation for Digital Computer Software used in Safety Systems of Nuclear Power Plants."

1.6.21 Reg. Guide 1.171, Rev. 0 (Sept. 1997), "Software Unit Testing for Digital Computer Software used in Safety Systems of Nuclear Power Plants."

1.6.22 Reg. Guide 1.172, Rev. 0 (Sept. 1997), "Software Requirements Specifications for Digital Computer Software used in Safety Systems of Nuclear Power Plants."

1.6.23 Reg. Guide 1.1.73, Rev. 0 (Sept. 1997), "Developing Software Life Cycle Processes For Digital Computer Software Used In Safety Systems Of Nuclear Power Plants"

1.6.24 IEEE Standard 1074-1995," IEEE Standard for Developing Software Life Cycle Processes"

1.6.25 IEEE/EIA Standard 12207.1, "Information technology – Software life cycle processes – Life cycle data"

1.6.26 IEEE Standard 1228-1994, "IEEE Standard For Software Safety Plans"

1.6.27 NUREG-0800, "USNRC Standard Review Plan for the Review of Safety Analysis Reports for Nuclear Power Plants", (SRP) Chapter 7, Revision 4, dated June 1997

1.6.28 NUREG/CR-6430, "Software Safety Hazard Analysis"

1.6.29 ABB CENO Proposal Development and Project Execution Process Manual

1.6.30 (Reference Deleted)

2. ORGANIZATION

Reference 1.6.1 describes the organization. Within the Cognizant Engineering Organization, software activities are organized into the following two teams:

- The design team develops the system requirements, software design, and code for the Common Q systems. The design team may develop common software that is used in systems developed by other supervisory groups.

- The V&V team performs verification and validation activities on the Common Q systems. The design team may perform the validation activities. In this case, the V&V team shall oversee the conduct of these activities by reviewing documentation and witnessing testing.

Reference 1.6.11 requires that the V&V team for a safety system be organized independently of the design team. The CEO meets this requirement by not allowing V&V team members to participate on the design team, even on a part time basis. The V&V team leader, responsible for the V&V, must not be the design team leader. Exhibit 2-1 shows the relationship between the design team (under the Project Manager) and the V&V team.

Team leaders are assigned specific responsibilities and the authority to assure the accomplishment of software management and control through written plans, procedures, standards, and instructions.

The CEO Manager is the manager of the CEO or his/her immediate reporting staff (i.e., manager/supervisor), and is ultimately responsible and accountable for:

- Plans, schedules, procedures, methods, and techniques required in the technical and administrative performance of the Common Q related software.

- Compliance with this Software Program Manual.

He may assign a Project Manager to be responsible for the above items for a specific Common Q project.

Verification of the implementation of quality assurance requirements is performed by the independent Westinghouse Quality Assurance organization (see Reference 1.6.4) according to 10CFR50, App. B requirements.

3. SOFTWARE SAFETY PLAN

3.1 INTRODUCTION

3.1.1 Purpose

The goal of this safety plan is to enable the development of safety critical software for Common Q Systems, that has reasonable assurance that any software defects do not present severe consequences to public health and safety.

The safety objective of this plan is to provide procedures and methodologies for the development, procurement, maintenance, and retirement processes of Common Q safety critical software that the CEO will follow to mitigate the potential of a software defect jeopardizing the health and safety of the public.

Any acceptable risks shall be defined in the specific Project Plan (Reference 1.6.4) for a given system implementation.

3.1.2 Scope

This plan is prepared in accordance with Reference 1.6.27, Branch Technical Position (BTP) HICB-14, "Guidance on Software Reviews for Digital Computer-Based Instrumentation and Control Systems", and Reference 1.6.26.

It applies to all Common Q safety critical software whose failure could result in severe consequences to public health and safety. For Common Q systems, safety critical software is defined as software belonging to the "Protection" class as defined in the Section 1.

3.2 DEFINITIONS, ACRONYMS, ABBREVIATIONS AND REFERENCES

Refer to Section 1.6 for a list of definitions, acronyms, abbreviations and references.

3.3 SOFTWARE SAFETY MANAGEMENT

Consistent with Reference 1.6.26 this section provides a description of the software safety organization, and the management of software safety activities and safety analysis requirements.

3.3.1 Organization And Responsibilities

Section 2 defines the organization that is responsible for design and implementation of Common Q safety-related software.

The software safety organization is composed of two parts:

1. An independent Quality Assurance department outside of the CEO generates quality assurance procedures and directives that are followed by all CEOs. They provide oversight by way of periodic audits to verify that the CEO is correctly abiding by both the procedures and directives it generates, and those generated by the CEO. The Manager of Quality Programs approves this Software Program Manual which includes the software safety plan.

2. An independent V&V team within the CEO performs the safety activities of the CEO for a given Common Q system implementation project. Refer to Section 5 for a description of the V&V team.

The V&V team leader shall have the following software safety responsibilities:

1. Ensure there is sufficient, independent, technically qualified and trained resources to implement the requirements of this software safety plan. Training includes familiarizing the V&V team members with the methods, tools and techniques described in Section 5.4.5.

2. Coordinate software safety task planning and implementation with the CEO design team per Section 5.

3. Ensure that records are kept in accordance with Section 5 and Reference 1.6.4.

4. Support the QA department on any audits within the purview of its responsibilities.

The mechanism for communicating safety concerns, raised by project staff, to software safety personnel is defined in Reference 1.6.4.

3.3.2 Resources

The Cognizant Engineering Organization management shall develop an early understanding of the resources required to develop safety-related software, so that these resources are put in place when they are required. The Cognizant Engineering Organization Project Manager and the V&V team leader shall determine the resources required to implement a Common Q system. The CEO Manager shall assign the appropriate resources to the Project Manager and V&V team leader. The following resources are considered for both the design and V&V team:

- Personnel
- Test materials and data
- Computers and other equipment
- Equipment support
- Tools
- Financial and Schedule

The Cognizant Engineering Organization Manager shall maintain an up-to-date resource plan and assures that the resources are made available when required.

A detailed schedule with linked tasks and work breakdown structure shall include software safety activities and be maintained to monitor progress throughout the project. Project schedules and resource allocations shall be evaluated.

3.3.3 Staff Qualifications And Training

The qualifications and training requirements for those personnel performing software safety functions are primarily the same as those performing the software design.

The following table identifies the team that will perform the tasks identified in Reference 1.6.26, Section 4.3.3:

Task	Assignee
Define Safety Requirements	Design Team
Design and Implement safety-critical portions of the system	Design Team
Perform Software Safety Analysis Tasks	V&V Team
Test safety-critical features	V&V Team*
Audit Software Safety Plan Implementation	V&V Team**
Perform Process Certification	V&V Team***

*The V&V team can delegate the responsibility of testing to the design team. In this case, the V&V team shall review the test plans, procedures and execution for compliance.

**The V&V team shall audit the design teams compliance to the plan. The QA department shall audit the CEO's compliance to the plan.

***The V&V team shall certify the project execution process including compliance with the software safety plan. The QA department shall certify the CEO's processes by audit.

One of the most important factors in developing reliable software is the development and use of a qualified staff. Safety-related software needs to be prepared by carefully selected personnel who can produce a reliable product. Although staff selection is one means of establishing a capable staff, training is also a key factor in establishing and maintaining a qualified staff. In assessing the training requirements, the Cognizant Engineering Organization Manager considers that:

- Training needs vary by individual;
- Training and retraining may be needed at various project phases;
- Staff qualification and training needs need to be periodically reassessed.

The V&V team shall be trained in the tools, techniques and methodologies described in Section 5.4.5.

The Cognizant Engineering Organization Manager assures that all personnel participating in the design, implementation, test and verification of safety-related software are qualified to perform their assigned tasks. Since there is currently no industry sanctioned certification program for safety-related software personnel, the Cognizant Engineering Organization Manager assesses the capabilities of candidates and selects appropriately qualified personnel based on the manager's experience.

In determining whether any candidate is qualified, the Cognizant Engineering Manager considers whether the candidate:

- Understands the system and its potentially hazardous effects, as described in Section 3.4;
- Understands the job to be performed;
- Has, or is capable of obtaining, working knowledge of system software and tools required to do the job;
- Possesses the combination of skills and knowledge to perform the job; through a proper level of formal education, supplemental training, and experience;
- Understands the related quality assurance, configuration management, and verification and validation plans;
- Is able to produce reliable software, good documentation, and can implement required quality assurance practices.

Throughout a project, requirements and tasks may change. These changes, which are sometimes subtle, may result in previously qualified personnel being unqualified for the changed work. Therefore, the Cognizant Engineering Manager shall periodically reassess the qualifications of all personnel working on safety-related software, particularly when specific changes to the project become known. The Cognizant Engineering Manager may direct additional training before the changes are effective, in order to maintain a fully qualified project team.

Personnel performing software safety reviews shall have meet the qualifications for an independent reviewer as defined in Reference 1.6.4.

3.3.4 Software Life Cycle

The software life cycle to be implemented for Common Q system development activities including V&V is defined in Section 1.4.2. Section 3.4 describes the relationship among specific software safety tasks and the activities for each phase of the software life cycle.

3.3.5 Documentation Requirements

The documentation for Common Q software shall be prepared in accordance with the requirements in Section 8, and incorporates the software safety documentation

requirements. The change and approval process for software safety-related portions of project documentation is the same as for other documentation as specified in Section 4.6.

3.3.5.1 Software Project Management

A project plan, compliant with Reference 1.6.4 shall be developed that will coordinate both the system development, software safety and quality assurance activities to ensure that all are done according to prescribed procedures and that adequate resources are allocated for their proper execution.

3.3.5.2 Software Configuration Management

Section 6 contains the requirements for software configuration management. Any deviations to these requirements shall be documented in the project specific project plan. Section 6 defines specific SCM responsibilities for a Common Q project and covers each phase of the software life cycle.

3.3.5.3 Software Quality Assurance

Section 4 is the Software Quality Assurance Plan (SQAP) that describes the requirements and methodology to be followed by the Cognizant Engineering Organization in developing, acquiring, using and maintaining safety-critical software. This SQAP is compliant to Reference 1.6.13.

3.3.5.4 Software Safety Requirements

The System Design Requirements (Section 8.2) document specifies the safety requirements to be met by the software to avoid or control system hazards.

3.3.5.5 Software Design Description

The Software Design Description (SDD, Section 8.3) includes descriptions of the software design elements that satisfy the software safety requirements.

3.3.5.6 Software Development Methodology, Standards, Practices, Metrics and Conventions

The standards, practices, conventions and metrics to be applied to the Common Q project are defined in Section 4.5.

3.3.5.7 Test Documentation

Test documentation includes test plans, test procedures and test reports. Test procedures incorporate test design and test cases.

3.3.5.7.1 Test Plans

The test plans provide a high level description of all tests that will be conducted for the Common Q project. They shall contain the requirements for all acceptance test procedures and defines each required test to be conducted. They also define the methodology for the disposition of test exceptions (errors). This document is verified against the outputs generated from the requirements phase of V&V for completeness. All prerequisites for testing shall also be identified. Section 4.3.2.2 describes the requirements for a test plan.

3.3.5.7.2 Test Procedures

The test procedures are the actual tests conducted on the Common Q safety-related software. They include test setup, precautions and limitations, and the test cases used to validate proper operation. The test procedures are verified against both the test plan and outputs generated from the requirements phase of V&V. Refer to Section 5.5.6 for a description of test procedure contents.

3.3.5.7.3 Test Reports

The test reports document the execution of the acceptance test procedures. In addition to attaching the signed and checked off test procedure, the test reports provide an overall summary of the test results and the resulting test exception reports generated during the test. The system configuration at the time of test execution is also documented in the test reports. Test Reports are prepared in accordance with Reference 1.6.14, Section 10.

3.3.5.8 Software Verification and Validation

The Software V&V documentation is described in Section 5.

3.3.5.9 Reporting Safety Verification and Validation

V&V reporting is described in Section 5.

3.3.5.10 Software User Documentation

User documentation is described in Section 8.6.

3.3.5.11 Results of Software Safety Requirements Analysis

The results of the Software Safety Requirements Analysis as described in Section 3.4.2 below shall be included in the following documents:

- Software Requirements Specifications (Section 8.2)
- Test Plan (Section 4.3.2.2)
- Test Report (3.3.5.7.3)
- Hazards Analysis (Section 3.4.1)

3.3.5.12 Results of Software Safety Design Analysis

The results of the Software Safety Design Analysis as described in Section 3.4.3 below shall be included in the following documents:

- Methods – Software Program Manual Sections 5, 6, 7 and 9
- Hazards – Software Requirements Specification and Software Design Description (Sections 8.2 and 8.3)
- Classification – Software Program Manual Exhibit 4-2
- Architecture Evaluation – V&V Report (Requirements Traceability Analysis)
- Requirements Compliance – Requirements Traceability Analysis (Section 4.3.2.2)
- Required Tests – Test Plan (Section 4.3.2.2)
- Modifications to Coding and Testing Techniques – Final V&V Report (Section 8.5)

3.3.5.13 Results of Software Safety Code Analysis

The results of the Software Safety Code Analysis as defined in Section 3.4.4 below shall be found in the V&V Report for the Implementation Phase of the software life cycle. Any changes will be documented in either V&V Discrepancy Reports or as suggestions in the V&V Report.

3.3.5.14 Results of Software Safety Test Analysis

The results of the Software Safety Test Analysis as defined in Section 3.4.5 below shall be found in the V&V Report for the Testing Phase of the software life cycle.

3.3.5.15 Results of Software Safety Change Analysis

The results of the Software Safety Change Analysis as defined in Section 3.4.6 below shall be found in the V&V Report. For each software life cycle that is revisited by the design team, the V&V team will analyze the impact on the previous life cycle phase as well as the phase it is analyzing. The results of the each phase's analysis will be found in the V&V Report for that software life cycle phase.

3.3.6 Software Safety Program Records

Records generation and maintenance procedures required for Common Q software are described throughout this Software Program Manual. Originals of issued documents for Common Q software are maintained according to Section 8.

Before the software requirements phase is completed and after the overall system design is known, an evaluation is made to determine the safety critical hazards posed by the system through its interfaces. The analysis assumes that a worst case scenario of possible errors (hardware or software) has occurred in the system. Based on this assumption, the analysis results in an identification of system malfunctions that are injurious to public health and safety.

For each hazard identified above, the analysis further determines whether a software malfunction could produce the hazardous condition. These software hazards are identified in the Software Requirements Specification. Each software producible hazard is evaluated during each phase of development of the safety critical software.

Results of V&V analyses performed on requirements, design, code, test and other technical documentation is documented in the V&V Phase Summary Reports and the Final V&V Report. Information on suspected or confirmed safety problems in the prerelease or installed system is recorded in the Final V&V Report. Results of audits performed on software safety program tasks is documented in the V&V Phase Summary Reports and in the Final V&V Report. Results of safety tests conducted on all or any part of the entire system are documented in the Test Report. A record of training provided to software safety program personnel is maintained by the CEO per Reference 1.6.4. Software safety certification is documented in the Code Certificate.

Retention of software safety program records is in accordance with Reference 1.6.4. The initiation and completion criteria for software safety program tasks for each phase in the software life cycle are defined in Section 5.

The tracking system used to ensure that hazards and their status are tracked throughout the software life cycle through retirement is the RTA and RTM as described in Section 5.

3.3.7 Software Configuration Management Activities

A key factor in developing reliable software is strict and detailed configuration management. Software configuration management activities for Common Q software are described in Section 6.

3.3.8 Software Quality Assurance Activities

Software quality assurance activities for Common Q software are described in Section 4.

3.3.9 Software Verification and Validation Activities

Software verification and validation activities for Common Q software are described in Section 5. These activities conform to the requirements in reference 1.6.8 and 1.6.11.

3.3.10 Tool Support and Approval

Section 4.9 describes the use of software tools that are used in development of Common Q systems. Tools may produce better program structure and more reliable software through the automation of repetitive or time consuming tasks. The Project Manager approves the use of any tool. This approval is based on an evaluation of the tool's readiness for use on a project involving safety-related software. This evaluation considers:

- The tool's past performance;
- The extent of tool validation already performed;
- The consistency of tool design with planned use.

The inadvertent introduction of software hazards by project tools is mitigated by the proper use of techniques for software configuration management, software quality assurance and V&V as described in this SPM.

3.3.11 Previously Developed Or Purchased Software

Section 4.1.2 describes the requirements for using existing software, including purchased software, as safety critical software.

3.3.12 Subcontract Management

Section 4.12.2 specifies the provisions for ensuring that subcontractor software meets established software safety program requirements.

3.3.13 Process Certification

The Computer Code Certificate is included in the final V&V report and is used to document that the delivered software produced for a Common Q is approved for design use (see Section 8.).

3.4 SOFTWARE SAFETY ANALYSES

3.4.1 Software Safety Analyses Preparation

It is vitally important to understand the ways that a system could potentially present hazards to public health and safety. The system design and review techniques described in this SPM are used to avoid, preclude, or mitigate the impact of potential software hazards in systems built using the Common Q platform.

A Preliminary Hazards Analysis (PHA) will identify the following:

1. Hazardous System States

   Before the software requirements phase is completed and after the overall system design is known, an evaluation is made to determine the safety hazards posed by the system through its interfaces that are injurious to public health and safety. The plant safety analysis (documented in the PSAR) defines the safety-critical hazards (accidents) posed by the plant that may be injurious to public health and safety. The Failure Modes and Effects Analysis performed for the specific Common Q System analyzes the vulnerability to single failures (hardware and/or software hazards) at the subsystem module level, including existing compensating provisions (hazard controls) within the design of each system. These two sources (PSAR and FMEA) form the design bases for software safety requirements for the Common Q Safety System.

2. Sequences of actions that can cause the system to enter a hazardous state

   For each identified hazard, the analysis determines whether a software malfunction could produce the hazardous condition, or the hazard could affect software operability. These hazards are identified in the Software Requirements Specification. Each software related hazard is evaluated during each phase of development of the protection class software. Reference 1.6.28 shall be used as a guide in performing this analysis.

3. Sequences of actions intended to return the system from a hazardous state to a nonhazardous state

   For each hazardous state, the system design must account for returning the system to a non-hazardous state. In preparing the Software Requirements Specification, the software developer considers techniques that can avoid a hazardous condition, or return the system to a nonhazardous state. The result of the requirements phase may be a set of required or forbidden design, coding or testing techniques. The

requirements phase may also identify specific tests to be performed or the implementation of certain hazard recovery techniques.

The functional requirements in the System Requirements Document (Section 8.2.1) provide the high level system design as required in Section 4.4.1 b) of Reference 1.6.26. The interfaces between the software and the rest of the system is defined in the software requirements of the System Requirements Document (Section 8.2.2).

3.4.2 Software Safety Requirements Analysis

In preparing the Software Requirements, the software developer considers techniques that can avoid a hazardous condition. The result of the requirements phase may be a set of required or forbidden design, coding or testing techniques. The requirements phase may also identify specific tests to be performed or the implementation of certain hazard recovery techniques.

Refer to Section 5.5.3 for a description of the software safety requirements analyses performed. These activities provide reasonable assurance that each system safety requirement is satisfied by the software safety requirements.

3.4.3 Software Safety Design Analysis

Refer to Section 5.5.4 for a description of the software safety design analyses performed. These activities provide reasonable assurance that each software safety requirement is satisfied by the software safety design.

3.4.4 Software Safety Code Analysis

Refer to Section 5.5.5 for a description of the software safety code analyses performed. These activities provide reasonable assurance that each software safety design element is satisfied by the software safety code.

3.4.5 Software Safety Test Analysis

Refer to Sections 5.5.5 and 5.5.6 for a description of the software safety test analyses performed for (module/unit testing and system testing respectively). These activities provide reasonable assurance that each system and software safety requirement is tested.

3.4.6 Software Safety Change Analysis

Refer to Sections 5.5.8 and 7 for a description of the software safety change analyses performed. These activities provide reasonable assurance that changes to safety critical

software do not create, impact a previously resolved, or exacerbate a currently existing hazard, and does not adversely affect any safety-critical software design elements.

3.5 POST DEVELOPMENT

In spite of the best efforts by software personnel in developing reliable safety-related software, inappropriate use or maintenance of the software may undo the software reliability by the recipient after delivery. It is important that the recipient be trained and qualified to use or maintain the software. Software personnel shall be trained in the procedures in Section 9 involving problem reporting and correction.

3.5.1 Training

For each protection class software system, a separate training program will be developed to ensure safe operation and use of the software within the overall system. The training program will include safety training for the users, operators, maintenance and management personnel, as appropriate. A training record will be on file for each training session recording the instructor, date, material covered, and personnel attending, to ensure that the appropriate training has been obtained before using the system. The V&V team will review the training documentation for traceability to safety requirements per Section 5.5.7.

3.5.2 Deployment

3.5.2.1 Installation

Installation documentation shall be developed, prior to the installation and checkout phase of the software life cycle, that will include the procedure(s) for installing the software. The V&V team shall review this documentation according to the procedure in Section 5.5.7.

3.5.2.2 Startup and Transition

Changes to installed systems may be disruptive to operations, particularly if problems occur or the resulting system operates differently. A Software Installation and Startup Procedure will be prepared addressing the following (as appropriate to the configuration of the system being installed):

- Fallback modes for the new system
- Startup of backup components and subsystems
- Startup of the new system
- Parallel operation with backups

- Parallel operation of the old system and the new system
- Subsystem vs. full system operation
- Switchover to full system operation
- Validation of results from the new system
- Cross validation of results between the old system and the new system
- Fallback in the case of failure of the new system, including fallback to an old system if one exists.

3.5.2.3 Operations Support

Documentation of the system and its software is supplied as described in Section 8. This documentation includes design documents, user manuals and instructions for maintenance expected by plant personnel.

3.5.3 Monitoring

The Software Operations and Maintenance Plan (Section 7) contains requirements for monitoring the use of delivered software and associated problem reporting.

In addition, protection class software is designed so that the integrity of the software can be verified periodically to detect unauthorized modification of code or data. Procedures necessary to perform this verification shall be documented. Methods shall be considered that provide automatic verification of the system during operation.

3.5.4 Maintenance

Software changes during all software life cycles are executed according to the Software Configuration Management Plan in Section 6 and the Operational and Maintenance Plan in Section 7.

3.5.5 Retirement And Notification

Section 6.2.2 describes the retirement of software and associated notification to current users.

3.6 PLAN APPROVAL

The Software Safety Plan is to be reviewed and approved by the Manager of the Cognizant Engineering Organization. It should be reviewed against requirements specified in Reference 1.6.26 and Reference 1.6.27.

4. SOFTWARE QUALITY ASSURANCE PLAN

4.1 PURPOSE

4.1.1 Introduction

The Software Quality Assurance Plan (SQAP) describes the requirements and methodology to be followed by the CEO in developing, acquiring, using, and maintaining software to be used for the design and operation of Common Q systems. The SQAP complies with Reference 1.6.13.

Software to be developed and used for the Common Q systems shall be placed into the following software classes (see Section 1.2.1):

- Protection (safety critical)
- Important To Safety
- Important To Availability
- General Purpose

All software modules developed or used within a system shall be documented by class and category as shown in Exhibit 4-1 for PPS/RPS, CPCS and PAMS. Common Q applications not listed in Exhibit 4-1 shall document the software classification using Exhibit 4-2 or its equivalent. This shall be performed by the Common Q design team and may be included in the software requirements documents.

4.1.2 Scope

This SQAP is required for all quality classifications defined for the Common Q system: protection, important to safety, important to availability, and general software within the scope of Westinghoouse software.

This SQAP is based on the software life cycle model which is described in Reference 1.6.2.

Within each software class described in Section 4.1.1, there are categories of software which this SQAP addresses. These categories are described as follows:

1) Original software

2) Existing software

a. To be modified

b. Not to be modified

All software modules developed or used within a system shall be documented by class as shown in Exhibit 4-1 for DPPS, CPCS and PAMS. The documentation required depends on the category of software and shall be consistent with Exhibit 4-4. Common Q applications not listed in Exhibit 4-1 shall document the software classification using Exhibit 4-2 or its equivalent. The V&V team shall review the completed Exhibit 4-2 form. This shall be performed by the Common Q design team and may be included in the software requirements documents.

Software that is initially assigned to one software class can be reassigned to any other class provided that all tasks appropriate for the new class, up to the current phase of the software life cycle, are completed and satisfactorily reviewed.

Existing software is software that has been created, but not under this SPM. To qualify for use under this SPM, the software must be evaluated by the CEO to meet the following criteria:

- Existing commercial software may be used for nuclear safety related software if it is qualified using a Commercial Grade Dedication Program (CGDP) such as the one described in Reference 1.6.3. To qualify existing commercial general purpose software, the CEO shall select applicable portions of the CGDP and qualify the software to those portions.

- Existing NPP non-commercial software, that has been actively used in a nuclear power plant, may be used for the same class of software under this SPM, provided it has been maintained under an acceptable quality plan with an active program for problem and corrective action reporting. This software shall also have adequate design documentation, user documentation and well commented source code. This software shall have been verified and validated under another program that is judged by the V&V team to be acceptable.

- Other existing non-commercial software (i.e., source code freely available (e.g., freeware)) may be used under the following conditions:

  1. The software fulfills a specific requirement identified in the SRS.

2. The code is well organized and has adequate design documentation, and source code commentary.

3. Will undergo the V&V process starting at the design phase.

For existing software that is qualified as above, design documentation and code may be used without revision to meet format or content requirements of this SPM. Modifications to this software may be made in accordance with prior documentation and code format.

Under this SQAP, a software product that is contracted for development by a subcontractor is treated as original software unless the software already exists and is in use. In this case, it is treated as existing software.

This SQAP describes the methodology by which all software and associated documentation is managed throughout the life cycle. Software elements produced in the process of quality assurance are as follows:

- Test plans, cases, procedures and reports
- Review and audit results
- Problem reports and corrective action documentation
- Software configuration management plans
- Software verification and validation plans

4.1.3 Software Development Process

The software development process for original software is shown in Exhibit 4-3. This exhibit shows the relationship between software and hardware, the process of software integration and testing, the design documentation produced, and the quality assurance documentation required throughout the software life cycle.

As shown in Exhibit 4-3, software quality is assured through the process of verification reviews, validation testing at the different stages of development, and software configuration management during all phases of software development. Software Verification and Validation (V&V) activities are governed by the Software V&V Plan described in Section 5. Required test procedures and test reports are shown in Exhibit 4-3, and are based on the level of the test and the class of the software.

4.2 REFERENCES

References are listed in section 1.6.

4.3 MANAGEMENT

The management of all software for Common Q projects spans the software life cycle defined in Reference 1.6.2 and applies to all software classes described in Section 4.1.1.

4.3.1 Organization

The implementation of an effective SQAP is the responsibility of all persons involved in the software development process. Each person responsible for the software development shall perform their work in accordance with established standards, methods, and procedures identified in this SQAP.

Software life cycle activities for this project shall be performed by the Cognizant Engineering Organization (CEO) described in section 2. The CEO is responsible for the execution of all quality assurance tasks including Verification and Validation (V&V) and software configuration management.

Management of the SQAP is overseen by the CEO Manager. Organizationally, verification of the implementation of quality assurance requirements is performed by Quality Assurance in accordance with References 1.6.1 and 1.6.4.

The CEO shall ensure that software and associated documentation has been developed in accordance with the standards specified in this SQAP. This includes ensuring that the coding standards (Section 4.5.2.1), testing standards established in the test plan and documentation standards (Section 8) have been followed.

In general, software configuration management responsibilities for the CEO span all phases of the software life cycle for

- Development of Software Configuration Management Plans
- Execution of software configuration management activities per the SCMP
- Control of software through a librarian
- Baselining and integration of new software versions

4.3.2 Tasks and Responsibilities

This section describes the specific tasks and responsibilities to be performed by the CEO. All tasks and responsibilities described in this section apply to each system assigned to the CEO. Tasks are listed in the life cycle phase for which they will be performed. Typical tasks are: software design and development, software quality assurance planning, verification reviews, audits, test planning, test execution, and test reporting. Tasks required are based on software category. Exhibit 4-4 shows the software tasks for each category in each phase.

The following are some procedural type of actions that are performed to ensure traceability throughout the development and verification stages:

1) The software design documents are dated and signed by the designer and the design team leader.

2) Each source code listing document is dated and signed by the programmer.

3) The corresponding Common Q software verification report and software test procedures documents are dated and signed by the author and the verification team group leader. If test procedures are generated by the design team and reviewed by the V&V team, the V&V phase summary report will be generated and signed by the V&V group leader to indicate the review and acceptance of these documents.

4) Each software module is verified, dated, and signed by the verifier.

4.3.2.1 Initiation Phase

Common Q system software quality assurance planning shall be performed during this phase. A Project Plan (Reference 1.6.4) shall be developed. Any deviations to or additional project specific information for the SQAP, SVVP, SCMP or SOMP shall be specified in the Project Plan. The CEO shall also develop coding standards at this time.

4.3.2.2 Software Requirements Phase

Common Q system Software Requirements Specifications are developed during this phase. Input from the system requirements specification provides the necessary system and functional requirements to develop software requirements and hardware design. The system requirements specification is used to generate equipment specifications and software documents. These system requirements are noted in Exhibit 4-4.

The CEO shall be responsible for developing, maintaining, and updating its Software Requirements Specification (SRS). A separate SRS shall be developed for each Common Q system based on system requirements, and shall provide the detail and information sufficient to design the software. The SRS shall be divided to describe software requirements for the software in each class in the system. The SRS shall be developed in accordance with Section 8.2.2 of this SPM.

The V&V team, as shown in Exhibit 5-1 shall verify each SRS. The verification review shall ensure that the system requirements are properly reflected in the SRS. Verification of SRSs shall be performed in accordance with Section 4.6.2.1.

A Common Q specific test plan shall start to be developed in accordance with Reference 1.6.14, Section 3, to identify how the test activities will be implemented. It shall include the following topics as a minimum:

- General approach including: identification of test procedures and test cases, general test methods, documentation of results, and traceability methods to the SRS and SDD.
- Requirements for testing including: test boundary conditions on inputs and unexpected input conditions.
- Test management including: personnel, resources, organization, and responsibilities.
- Procedures for qualification and control of the hardware to be used in testing.
- Qualification and use of software tools.
- Installation test requirements for existing software which is used without modification.
- Regression test requirements for existing software which is modified.

Test plan development continues into subsequent phases and is completed in the implementation phase.

4.3.2.3 Software Design Phase

The CEO shall be responsible for developing, maintaining and updating a Software Design Description (SDD) for each software module. Each SDD shall be traceable to the

requirements set forth in the SRS, and shall include enough detail to begin coding in the Implementation Phase. All SDDs shall be developed in accordance with the requirements of Section 8.3.

The V&V team as indicated in Exhibit 5-1 shall verify each SDD. The verification review shall ensure that the software requirements identified in the SRS are properly reflected in the SDD. Verification of SDDs shall be performed in accordance with Section 4.6.2.2.

Prototype software may be developed to prove a new principle or to help further define the software design during this phase. Prototype software has a different software life cycle than the other categories of software that is usually shorter in duration. Specifically, prototype quality assurance tasks shall include:

- Adherence to coding standards
- Documentation of prototype design (format at the discretion of the individual CEO group)
- Informal verification reviews within the CEO group
- Limited software configuration management

Wherever prototype software is reused and integrated into the deliverable software, it shall undergo the respective software quality measures based on its software class. This includes software quality assurance tasks described above from the integration point forward in the life cycle plus any "skipped" tasks in the life cycle for; verification reviews, audits, software configuration management activities, required documentation, and conformance to coding standards.

4.3.2.4 Software Implementation Phase

Original software development and modifications to existing software shall begin with module coding by the CEO in accordance with the appropriate coding standard(s) listed in Section 4.5.2.1.

Existing software which has been qualified as described in Section 4.1.2 may be integrated into the software system and tested during this phase.

Verification of module code listings shall be performed by the group identified in Exhibit 5-1. Details of software module code verification is described in Section 4.6.2.3.

The test plan, started during the software requirements phase, is completed in this phase.

Testing during this phase can be accomplished by several methods at the discretion of the CEO group. Some possible methods are identified below:

- One method is to hierarchically assemble the modules into units and perform a unit test, and subsequently assemble all the units into the system and perform a system test.

- Or, the test sequence can be performed in a series of expansions. This could be accomplished by continually adding successfully tested modules to the "system" and test after each addition until the complete system is assembled and tested.

Module and unit testing shall be performed in accordance with the Test Plan and Reference 1.6.12. The responsibility for testing will be assigned to the design team and V&V team as shown in Exhibit 5-1. Unit test procedures and reports are only required for software classified as protection and as important to safety. Module test procedures and reports are only required for software classified as protection.

4.3.2.5 Testing Phase

System testing shall be conducted during this phase in the development environment when all of the system components have been integrated by the CEO group per the Test Plan. The purpose of this test is to evaluate the system as a whole for its ability to meet system usage and performance requirements. Test procedures and reports shall be documented in accordance with Reference 1.6.14, Sections 6 and 10 respectively, and verified by the groups identified in Exhibit 5-1. The groups identified in Exhibit 5-1 shall conduct system tests. Inter-system testing (the process of testing system interfaces) shall be performed in accordance with the Test Plan. Also, test procedures and reports shall be developed in accordance with Reference 1.6.14, Sections 6 and 10 respectively, and consistent with the Test Plan.

The Final Software Verification and Validation Report (SVVR) for the deliverable software shall be prepared during this phase. All User Documentation shall be developed during this phase in accordance with Section 8. Also, during this phase, all user documentation shall be verified by the V&V team.

4.3.2.6 Site Installation and Checkout Phase

Site Installation and Checkout of Common Q software will be dependent on the contractual arrangements made with the customer that purchased the specific Common Q system. If Westinghouse is responsible for software installation and checkout then the CEO shall have the responsibility for the Site Installation and Checkout Phase and the associated V&V requirements.

The preparation of the site test plan will be initiated during the requirements phase to support evaluation of requirement testability on-site. Validation of the installed software shall be performed to determine that the software was installed correctly. Software installation validation applies to initial software and any subsequent revisions.

During this phase the software becomes part of the installed equipment incorporating applicable software components, hardware, and data. The process of integrating the software with applicable components in the plant consists of installing hardware, installing the software, and verifying that all components have been included.

A TER log shall be maintained during the installation and checkout phase in accordance with the SAT plan. For protection class software this log shall be verified by the V&V team after installation.

After installation, the equipment and software shall be checked out according to the test plan and Site Acceptance Test (SAT) Procedure. All test exceptions shall be documented using the TER form and entered into the TER log.

In this phase, the site portion Software Verification and Validation Report (SVVR) shall be prepared for protection class software. Details of the SVVR are described in Section 8.5.

4.3.2.7 Operations and Maintenance Phase

Activity in this phase consists of maintenance of the software to:

- remove identified latent errors
- respond to new requirements, or
- adapt the software to changes in the operating environment.

Software modifications shall be approved, documented, verified and validated, and controlled in the same manner as described previously in the Design, Implementation and Test Phases. The SVVP (Section 5), in conjunction with the SCMP (Section 6), shall also be used to assist in the management of these activities and procedures.

4.4 DOCUMENTATION

4.4.1 Purpose

The documentation required for each category of software is listed in Exhibit 4-4. Section 8 of this SPM provides guidance for the development of documents. If required, documents listed shall be made lifetime quality records in accordance with Reference 1.6.4.

4.5 STANDARDS, PRACTICES, CONVENTIONS AND METRICS

4.5.1 Purpose

The standards, practices and conventions to be applied to the Common Q systems are contained in Reference 1.6.1. Compliance with these standards shall be monitored and assured through the review and audit process described in Section 4.6.

4.5.2 Content

4.5.2.1 Coding Standards

The software development process shall provide guidance to ensure standardization, compatibility and maintainability of resulting software products. The process shall provide a coding standard for each language, database, or software tool that allows author discretion in establishment or use of convention.

This requirement applies to the following typical software products:

Assembly languages
C/C++
Display building languages

Each coding standard shall contain, but is not limited to, the following information:

1) General

   This area outlines general ideas and concepts used to guide the creation of software written under a specific language.

2) Naming conventions

   - Filename extensions as far as how they are used to organize files.
   - Information pertaining to file organization within a system.
   - Variables naming.

3) Internal documentation guidelines

   - Program identification header content, placement, type, quality, and quantity.

- Revision history recording within each source file.

4) Stylistic conventions - issues that affect readability, such as indentation and use of white space.

5) Use of specific language features

6) Software tool usage guidelines

- Information and use of automatic make facilities
- Appropriate compiler flag usage.

7) Functions

- Modularity
- Naming

4.5.2.2 Software Testing Standards

Software testing methodologies, policies and practices shall be described in the system specific Test Plan. Specific format and content for test procedures (with test cases) and test reports shall also be provided in the Test Plan and shall be consistent with Reference 1.6.14, Sections 6 and 10.

4.5.2.3 Documentation Standards

All documents developed for Common Q systems shall comply with the requirements for format and content described in Section 8.

4.5.2.4 Metrics

The following metrics should be maintained for each Common Q system:

1. The errors discovered during dry run of the FAT should be identified through the use of Test Exception Report (TER) forms (Exhibit 8-1) so that all errors discovered can be resolved prior to FAT and that the number of errors discovered can be tracked for error discovery metric reporting. The overall goal is to identify a decreasing number and severity of errors as the testing progresses from pre-FAT to FAT to SAT.

2. FAT errors shall be reported through the use of TERs and the number and severity shall be identified for error discovery metric reporting.

3. Software errors discovered after FAT and before SAT shall be tracked through the use of TERs and the number and severity shall be identified for error discovery metric reporting.

4. Software errors discovered during SAT shall be tracked through the use of TERs and the number and severity shall be identified for error discovery metric reporting.

5. Software errors discovered after SAT (after system acceptance) shall be tracked through the use of Computer Code Error Notifications (or similar process), or software problem reports received from customers and the number and severity shall be identified for error discovery metric reporting.

4.6 REVIEWS

4.6.1 Purpose

The purpose of this section is to address the review requirements throughout the software life cycle.

The software reviews required by this SQAP address software classes and categories described in Sections 4.1.1 and 4.1.2.

Reviews are technical in nature and are designed to verify the technical adequacy and completeness of the design and development of the software.

Review activities applicable for original software are:

- SRS verification,
- SDD verification,
- code verification, and
- program documentation verification including each SVVR and testing review.

The reviews are the responsibility of the individual CEO group as described in Exhibit 5-1. The methodology of performing reviews is described in the SVVP (Section 5). Reviews performed within the CEO group shall be performed by peers who have an equivalent knowledge of the topic but who are not directly involved with the application as required in Section 2.0.

Audits are designed to ensure that software documentation and processes comply with the established standards and guidelines set forth on the project.

References to the SVVP are provided in this section to address system specific areas of the review and audit process. The intent of this section is to provide guidelines to conduct reviews and audits. Procedural aspects of the review shall be contained in the SVVP.

4.6.2 Minimum Requirements

Technical reviews shall evaluate specific software elements (such as files, functions, modules, or complete systems) to ensure that the requirements are adequate, technically

feasible and complete. The technical review shall be managed by the CEO design group according to the SVVP, and shall address the following items prior to all reviews:

- Statement of objective for the technical review
- Software element(s) to be examined
- Specifications for software elements to be examined
- Plans, standards or guidelines against which the software elements are to be examined

4.6.2.1 Software Requirements Review (SRR)

The Software Requirements Review (SRR) shall examine the Software Requirements Specification (SRS) to verify that it is clear, verifiable, consistent, modifiable, traceable, and usable during the operations and maintenance phases. The SRR shall include evaluation of the software requirements against the user's software application which is described in a higher level requirements document such as a system design specification document.

Specific SRR items are described in Section 5 and shall be described in detail as necessary in the SVVP. As a minimum, these items shall include:

- Traceability and completeness of the requirements
- Adequacy of rationale for derived requirements
- Testability of functional requirements
- Adequacy and completeness of verification and acceptance requirements
- Conformance to documentation standards
- Adequacy and feasibility of performance requirements
- Adequacy and completeness of interface requirements

Responsibilities, methodologies, and reporting of results are described in Section 5 and shall be described in detail as necessary in the SVVP. Frequently encountered categories or types of errors normally found in the SRS may also be included in the SVVP in order to aid the independent reviewer.

4.6.2.2 Software Design Review

4.6.2.2.1 Preliminary Design Review

The Preliminary Design Review shall include a review of the preliminary SDD, RTM, emphasizing the following issues:

- Detailed functional interfaces with other software, system equipment, communication systems, etc.
- Software design as a whole emphasizing allocation of software components to function, functional flows, storage requirements and allocations, software operating sequences, and design of the database
- An analysis of the design for compatibility with critical system timing requirements, estimated running times and other performance issues
- Human factor requirements and the human machine interfaces for adequacy and consistency of design
- Testability of design
- Technical accuracy of all available test documentation and its compatibility with the test requirements of the SRS
- General description of the size and operating characteristics of all support software
- Description of requirements for the operation of the software
- Identification of requirements for functional simulation, environmental recording, configuration, etc.

The review shall be organized by the CEO Manager and other technical competent personnel should participate in the review with the goal of finding any errors as early as possible in the design process.

The results of the review shall be documented using a technical review form that identifies all deficiencies identified in the review and provides a plan and schedule for corrective action.

4.6.2.2.2 Critical Design Review

The Critical Design Review evaluates acceptability of the detailed design documented in the SDD, and establishes that the detailed design satisfies the requirements of the SRS. The review also verifies the design's compatibility with the other software and hardware that the product is required to interact with and assess technical, cost and schedule risks of the product design.

The Critical Design Review shall include a review of the SDD for the following items:

- The compatibility of the detailed design with the SRS
- Available data in the form of logic diagrams, algorithms storage allocation charts, and detailed design representations
- Compatibility and completeness of interface requirements
- All external and internal interfaces including interactions with the data base
- Technical accuracy of all available test documentation and its compatibility with the test requirements of the SRS
- Requirements for the support and test software and hardware to be used in the development of the product
- Final design including function flow, timing, sizing, storage requirements, memory maps, data base, other performance factors

The results of the review shall be documented using the V&V Design Phase Checklist and should describe all deficiencies identified in the review and provides a plan and schedule for corrective action. After the SDD is updated to correct any deficiencies, it shall be placed under configuration control to establish the baseline to be used for the software coding.

4.6.2.3 Code Verification

Software code shall undergo periodic peer review by means of a code inspection. Code reviews are the responsibility of an independent reviewer within the CEO, and shall be performed in accordance with the Software Coding Standards described in Section 4.5.2.1. Code reviews shall include evaluation of the source code implementation against the SDD. The review criteria are specified in EXHIBIT 5-4.

4.6.2.4 Software Verification and Validation Plan Review

The SVVP (Section 5) is reviewed for adequacy and completeness of the verification and validation methods defined in the SVVP. An independent reviewer meeting the qualifications of Reference 1.6.4 does this review as part of the review process for this SPM.

4.6.2.5 Functional Review

The Functional Review is held prior to software delivery to verify that all requirements specified in the Software Requirements Specification have been met. The CEO Manager shall perform the Functional Review. The review shall include an overview of all documentation and a review of the results of previous reviews, including Software Requirements Review, PDR, CDR, and V&V Report (for safety related software).

The results of functional reviews shall be documented.

4.6.2.6 Physical Review

The Physical Review is held to verify that the software and its documentation are internally consistent and are ready for delivery. This shall include a review of the following deliverable items:

- Deliverable software media
- V&V Report and code certificate (for safety related software)
- User/Technical Manual
- Installation instructions (if required)

The results of physical reviews shall be documented.

4.6.2.7 In-Process Audits

In-process audits of a sample of the design are held to verify consistency of the design. Software inspections may be included as part of the in-process audit activity. Nuclear Quality shall perform in-process audits for nuclear safety-related systems including "Protection" and "Important to Safety" software classes. CEO Managers shall perform the in-process audits for all other systems. The audit shall review different items depending upon the software phase the audit is held and can include a review of the following items:

- Code versus design documentation (code walkthroughs or code inspections)
- Interface specifications
- Design implementations versus functional requirements
- Functional requirements versus test description

- Test descriptions versus test procedures
- Test procedures versus test reports

The results of in-process audits shall be documented identifying all deficiencies found. The CEO Manager then must evaluate the deficiencies and prepare plans and schedules for resolving the deficiencies.

4.6.2.8 Managerial Reviews

Managerial reviews are held periodically to assess the execution of all of the actions and the items identified in this SQA guideline.

The management review shall be documented by a report summarizing the review findings, exceptions to the process stated in the SQAP and recommended changes or improvements to the SQA process. The reviews result in statement as to the adequacy of the SQA process and its execution.

4.6.2.9 Software Configuration Management Plan Review (SCMPR)

The Software Configuration Management Review is held to evaluate the adequacy and completeness of the configuration management methods defined in the SCMP (Section 6) and their implementation. The review shall be performed by the V&V team (for protection class software) or by the CEO Manager and results documented to identify all deficiencies found and plans for their resolution.

4.6.2.10 Post Mortem Review

The CEO Manager shall hold a project closeout review with the project team upon completion of the project to ensure that all project activities have been completed, all deliverables have been shipped, and that all project quality assurance activities have been fulfilled.

4.7 TEST

Required testing to be performed for all software related projects includes:

- Module level tests*
- Unit level tests (can be part of Integration and System Tests)
- Integration Tests (Pre-FAT)
- System Tests (FAT)
- Installation Tests (SAT)

*For protection class software documented module tests are required.

4.8 PROBLEM REPORTING AND CORRECTIVE ACTION

4.8.1 Purpose and Scope

The purpose of a formal procedure of software problem reporting and corrective action is to ensure that all software errors and failures are promptly acted upon and in a uniform manner encompassing all project software. This procedure ties together the requirements of the SVVP and the SCMP. V&V activities are the primary vehicle to uncover software problems, while the SCMP shall ensure that actions taken to correct problems by changing configured software are consistent and traceable.

Problem reporting and corrective action procedures shall span the entire software life cycle and all software classes identified in this SQAP. These procedures are detailed in Section 9 of this SPM.

4.9 TOOLS, TECHNIQUES AND METHODOLOGIES

Software development for Common Q projects shall use a number of techniques to help assure all software is designed, implemented, and documented in accordance with the Common Q objectives of building software which meets the requirements and which is maintainable over time in the most cost effective manner. The tools, techniques and methodologies employed in this process shall ensure that the software is verifiable from each phase of the project to the next.

- Use of structured design techniques for analyzing and developing the software design. These shall include data flow diagrams, where applicable, to represent the interactions among modular elements and the flow of data among them. Entity-relation charts may be used to represent any relational database structures.

- CEO Management sign-off and approval of all design and V&V documentation shall include one of the following:

  1. The immediate supervisor or manager of the preparer, or
  2. The CEO Project Manager

- All members of the Common Q design and V&V team shall be trained in the contents of this SPM. This training shall be documented in the individuals QA training record.

- Use of the waterfall model of software development and testing techniques to help assure that the requirements are correctly translated into design and implementation products.

- The use of commercially available automated tools for software configuration management should be employed to the maximum extent possible.

4.10 CODE CONTROL

Code Control shall be provided as part of software configuration management per Section 6. Methods and facilities used for maintenance, storage, documentation and security for controlled versions of the software during all phases of the software life cycle are also defined in Section 6.

All software items shall be controlled to maintain the items in a known and consistent state at all times. New software and modifications to existing software shall follow the configuration requirements for all life cycle phases. Existing software which is not to be modified, including tools used in the software development, test, and documentation process, shall be placed under configuration control procedures upon its introduction or use within the software system.

4.11 MEDIA CONTROL

The methods and facilities used to protect computer program physical media from unauthorized access or inadvertent damage or degradation are described herein.

4.11.1 Media Identification

Media identification is described in Section 6.3.1. Removable storage media should not be switched, renamed, or initialized without prior approval from the CEO group.

A cabinet and disk file cabinet shall be provided for storage and easy access.

4.11.2 Archival Requirements

A locked storage facility shall be used to store all project software in a location separate from the software development area. This locked storage facility shall be able to accommodate the storage of all utilized types of media.

After important Common Q software development milestones or baseline configurations are archived, a known software configuration shall be completely backed up and periodically stored in the vault.

4.12 SUPPLIER CONTROL

The purpose of this section is to describe the level of software quality assurance measures to be applied to software supplied to a Common Q system from parties other than the CEO.

4.12.1 Existing Software

This SQAP defines existing software as software which was previously developed prior to the Common Q system being developed, to satisfy a general market need and may be considered for use on a Common Q project. The software may be subsequently modified prior to delivery, or it may be used "as is."

Existing software includes commercial software which is integral to the delivered system and software which is determined to be in support of the delivered system. Examples of integral software would be:

- Operating systems
- Compilers, Linkers, Loaders
- Database software
- Communication Drivers
- Man-Machine Interface software
- Display building software

All commercial software which will be used in Common Q nuclear safety related systems must meet the requirements established in a Commercial Grade Dedication Program like the one described in Reference 1.6.3.

For existing software which is modified for a Common Q project, all software requirements specified in this SQAP for original software shall be in effect for the modifications. The minimum V&V activities applicable for modifications to existing software are: software modifications requirements verification, software modifications design verification, program modification documentation verification, and software validation. Regression testing using test cases shall be conducted to ensure that the modifications do not produce unintended adverse effects, and to ensure that the modified software still meets the original software requirements.

Existing software that is not modified shall be qualified for use according to Section 4.1.2.

Once qualified for use, the software shall fall under the Common Q SCMP (Section 6). Once installed, the software shall meet the following requirements:

- Verification and Validation during Installation and Operation per the SVVP,
- Configuration Management during Installation and Operation per the SCMP,
- Documentation including: Test Plans, Procedures, SVVR, and User Manuals,
- Problem reporting and corrective action procedures, and
- Records of delivered documents and software

4.12.2 Sub-Contracted Software/Services

Original software which is developed by a contractor and purchased shall adhere to the quality assurance requirements specified in this SQAP for original software. This applies regardless of whether the software will be subsequently modified or not.

Where available, third party contractors who supply software are required to provide feedback to the CEO of problems encountered by the supplier or other users of similar software. Where available, this feedback information shall be supplied to the CEO automatically without request. Also, the CEO group has the responsibility of informing the supplier of such software of any problems encountered by the user in the use or maintenance of the software.

Additional requirements for subcontracted software and services are as follows:

- Software and services must be procured from approved supplier per Reference 1.6.4.
- Suppliers must have written quality assurance policies which meet the principles and intent of this SQAP.
- Purchase orders shall require the Supplier to make available documents which are evidence of compliance with the principles and intent of this SQAP.

- Purchase orders shall require the Supplier to deliver adequate user documentation, test procedures and test reports.

4.13 RECORDS COLLECTION, MAINTENANCE AND RETENTION

Records collection, retention, and maintenance shall be in accordance with Reference 1.6.4.

4.14 TRAINING

All design and V&V team members involved with Common Q software shall be trained on the Software Program Manual (either by classroom training or self-study). The individual's QA training record shall be used as documentation that this training took place.

4.15 RISK MANAGEMENT

Reference 1.6.29 describes the process and requirements for risk management for project execution.

5. SOFTWARE VERIFICATION AND VALIDATION PLAN

5.1 PURPOSE

The purpose of this section is to establish requirements for the V&V process to be applied to Common Q systems. It also defines when, how and by whom specific V&V activities are to be performed including options and alternatives, as required. The section includes various V&V methodologies aimed to increase the system reliability and availability. Some of these methodologies employ systematic checks for detecting errors in the system hardware and software, during the system development and implementation process. This section explains requirements for the V&V processes starting with the system design document stage and all necessary V&V activities to verify and/or validate I&C systems. This SVVP complies with References 1.6.8. The life cycle processes are consistent with References 1.6.24 (as augmented by Reference 1.6.23) and 1.6.25.

The goals of this V&V plan, when applied to a specific project, are to:

- Improve the system reliability and availability
- Reduce system costs by exposing errors as early as possible
- Provide a systematic process of objectively evaluating the system's performance
- Demonstrate compliance with customer requirements, industry standards and licensing requirements.

5.1.1 Categorization of Software Items and Review Scope

V&V is performed on documents and materials that are produced according to the category of each software item, as described in Section 4. For example, a software design description is not required for an existing commercial off-the-shelf software package. V&V activities only include documents and materials identified in Section 4.

5.1.2 V&V Program Implementation

V&V activities are integrated into the requirements, design, implementation, test and installation phases described in Section 4. Experience has shown that the earlier a deficiency is discovered, the easier and more economical it is to resolve. The initial activity is the review of system functional requirements prior to any detailed design of hardware or software. Verification activities are performed at the end of this phase, and each subsequent phase. These activities determine that all requirements have been

properly transferred from the input products to the output products of the phase, with amplifications or modifications appropriate to the phase. Upon completion of the software implementation, validation activities are performed. These activities determine that the operation of the system is consistent with the system requirements. Thus V&V activities are integrated with project activities from the beginning to end.

Once a system design and implementation has been verified and validated, any succeeding systems manufactured of the same design are certified by standard manufacturing test procedures. Many of the tests used by manufacturing are the same or equivalent to those used in the system V&V process. The equivalent of the system validation tests are performed as a minimum on every successive system of the same design that has been previously verified and validated. This is referred to as a Factory Acceptance Test. Traceability of all tests performed on manufactured units are maintained under configuration management control. Any design changes that would impact manufactured units are reverified and maintained under configuration management control.

5.1.3 Prominence of V&V Documentation

Traceability is important, not only to document the V&V activities, but also to record appropriate actions taken to resolve discrepancies. Thus a V&V program is, by its nature, oriented heavily towards documentation and the ability to trace changes in project documents. The generic documentation plan defines the structure and format of contents of the documents which may be produced during various phases of the project. The documents contents will vary depending on the specifics of system or project; however a system to trace the documentation and deficiency resolution is required. In the early phases of the system design process the system is divided into manageable modules of software and hardware. In the later phases, these modules are integrated into a total system.

The Configuration Management Plan addresses these issues and details (1) how the documents are controlled, (2) how records of changes and distribution are maintained, and (3) status of each document is identified.

5.1.4 Overall Common Q and System Specific V&V Plans

This Common Q V&V plan details the V&V process and activities involved during the various phases, and details various tools and techniques to be used. Any deviations or additional project specifics to the SVVP, such as scheduling specific V&V tasks and resource identification, shall be defined in either a project-specific V&V plan or in the Project Plan as described in Reference 1.6.4.

5.2 REFERENCED DOCUMENTS

References are found in Section 1.6.

5.3 DEFINITIONS

Acronyms and definitions are found in Section 1.5.

5.4 VERIFICATION AND VALIDATION OVERVIEW

5.4.1 Organization

All Systems are classified as described in Section 1. V&V activities vary according to system class in the degree of independence required for the reviews. Protection class systems utilize non-complex software items and an independent reviewer as defined in Reference 1.6.4, shall perform review of any particular protection software item.

Independent reviewer means that the reviewer has not been a member of the system's requirements team or design team while the system has been designed. Such independence is important to obtaining an objective review, based solely on the documents and materials produced, and not influenced by designer intentions or assumptions. Consideration shall be given to performance of these reviews by outside agencies or consultants.

The reviewers of software in non-safety critical classes may be members of the requirements team or, in some cases, the design team. Nevertheless, the review of any particular software item shall not be performed by the individual(s) responsible for the requirements or design of the item. An independent reviewer must also be one who can perform a competent review.

Exhibit 5-1 identifies the minimum review independence required for each type of document or software item, for each class of system.

5.4.2 Master Schedule

The project plan (described in Reference 1.6.4) shall include the project V&V schedule and required milestone delivery dates. This shall be developed in coordination with the V&V team leader (for a V&V team of more than one person).

5.4.3 Resources summary

5.4.3.1 Design Team

The composition of the design team shall be established in terms of the functions that are required within the team. The following functions are fulfilled by one or more people depending on project size and complexity.

5.4.3.1.1 Lead Engineer

This is the team leader, responsible for all technical matters in the development of the system. Normally one person is designated as the lead engineer for a project. The lead

engineer shall have the responsibility for the development of the software design requirements and software design specification documents. Global decisions on the structure of the software, decomposition, and data base are made by the lead engineer. Some critical sections of the programs, both in terms of importance and complexity, may be coded by the lead engineer. The lead engineer supervises the rest of the design team in technical matters.

5.4.3.1.2 Programmer

A programmer's main responsibility is to develop the code and provide the details for the software design at the module level to meet the software design requirements. In most projects, it is anticipated that there will be more than one programmer.

5.4.3.1.3 Librarian

The maintenance of the software library is a key element in the V&V process. The librarian, in the execution of that position ensures that a project's software conforms to library standards, retains records of the software modules in use and revision levels, and assures the procedures for software changes are followed.

5.4.3.1.4 Language Expert

This team member supplies the technical information on the programming language that is used. This person is preferably one of the programmers.

5.4.3.1.5 Hardware Expert

The hardware expert's responsibility is to maintain all hardware in working order in the "as delivered" system configuration. The hardware expert should also have software experience in order to assist in writing software drivers. There could be more than one hardware expert per project.

5.4.3.1.6 Project Manager

The CEO Manager (or designated project manager) is responsible for the technical adequacy of the work performed by design and requirements teams reporting to him/her. This manager also performs all the administrative functions for the system development and V&V in the project. These functions include the development planning, budget, and scheduling, as well as the necessary administrative interfacing.

5.4.3.2 Verification Team

The composition of the verification team shall be established by the functions carried out, similar to the manner of the design team. The following functions are fulfilled by one or more people depending on project scope and complexity.

5.4.3.2.1 *Lead Verifier*

The lead verifier is the team leader responsible for all technical matters concerning the verification of the system. The lead verifier has the responsibility for the development of the verification requirements and validation test procedures documents. It is also the responsibility of the lead verifier to check the documentation compiled by the design team to the requirements specified in Section 5.

5.4.3.2.2 Verifiers

The verifiers check the portions assigned to them with the use of the project validation test procedures and requirements documents. These checks are carried out by the verifier with any of the appropriate tools and techniques discussed that are agreed upon with the lead verifier. The verifier is also the independent reviewer for the design team. As is the case with the number of programmers in a project, it is anticipated there will be more than one verifier.

5.4.4 Responsibilities

5.4.4.1 Verification Team Responsibilities

The Verification team shall evaluate the test documentation and results and perform supplemental testing as deemed necessary.

The emphasis shall be placed on assuring that the documentation detailing the system, hardware and software functional requirements and performance specifications are clear, accurate and complete.

The documentation shall be reviewed looking for omissions, inconsistencies, inaccuracies and errors of omission/irrelevant requirements. Some significant functional requirements may be identified and monitored as development progresses.

If warranted, supplemental testing shall be performed without evaluating or using designers test data base/bed, or test results.

The emphasis shall be placed on full independent analysis of the system requirements and design specifications, supplemental testing and evaluation for the systems requiring the highest reliability.

The test progress shall be monitored and the execution of tests and test results shall be evaluated. Supplemental testing shall be recommended when found necessary.

The actual assignment of team members for engineering, verification, testing, and validation is shown in Exhibit 5-1.

Requirements and the implementation of design shall be evaluated to ensure that the resulting system operation is functionally correct and meets the performance objectives.

5.4.5 Tools, Techniques and Methodologies

5.4.5.1 Automated Tools

Part of the V&V planning process includes the selection of appropriate tools for a given project.

5.4.5.2 V&V Core Activities

The following V&V core activities are applicable to every system (irrespective of the selected verification test strategy).

1) Upon completion of the V&V review of a particular software item, the reviewer will complete and sign the checklist (Exhibit 5-2 through 5-6) for the phase in which the preparation of the software item is completed. The questions in this checklist provide a basic set of considerations that the V&V reviewer shall include in the review.

2) Reviews assure clear, accurate and complete documentation detailing the system, hardware and software design requirements and design specifications.

3) System validation testing, as a minimum, will be performed on every system as part of both the development and manufacturing processes; details of test bed, validation test procedures and test results will be documented for review and audit purposes in accordance with Reference 1.6.14, Sections 6 and 10 respectively.

4) Unit and Module Testing will be performed by the design team or V&V team according to Exhibit 5-1. The test plan, test procedures and test results will be documented, as required and in accordance with Reference 1.6.14, Sections 3, 6 and 10 respectively. Note: There is not necessarily a one-to-one relationship between design units, implementation units and test units. Several design units may make up an implementation unit (e.g. a software program) and several implementation units may make up a test unit.

5) The V&V team will participate in design reviews, audits and configuration management activities.

6) Commercial off-the-shelf (COTS) software used in a nuclear safety related application must go through a Commercial Grade Dedication process as described in Reference 1.6.3. The V&V team shall review the Commercial Grade Dedication report for COTS software used in the system to determine its applicability and suitability for the system requirements.

If the COTS software to be used in the nuclear safety related system has changed since the Commercial Grade Dedication report was issued, then the V&V team must do one of the following:

1. Review the changes to COTS software and determine their impact on the system. Evaluate the reported errors for new releases and determine their impact on the application. Recommend tests to be conducted where an impact is identified.

2. Verify that the changes to the COTS software were performed in accordance with acceptable industry standards (e.g., IEEE 7-4.3.2 or IEC-880).

5.4.5.4 Requirements Traceability Analysis

Throughout the software life-cycle, a software requirements traceability analysis will be performed and a requirements traceability matrix maintained for each system.

The analysis provides a method to cross-reference each software requirement against all of the documents and other software items in which it is addressed. The purpose of this analysis is to assist the reviewer in determining the propagation of requirements through the software items.

A typical format of the Requirements Traceability Analysis Matrix lists all of the requirements down the left side, where each row is a separate requirement. All documents and other software items are listed across the top, where each column is a unique document. The items listed across the top of the matrix include revisions of every item. The inclusion of revision documents within the analysis provides a history of requirements changes throughout the project. A database format is acceptable provided its contents include the information and relationships described in this section.

Requirements entered in the analysis are organized into successive sub-requirements as described in each document.

Within each cell of the analysis, a code is entered to denote the status of the requirement within the document. The codes are as follows:

Y - Requirement is included in this document
N - Requirement is not included in this document

The V&V team shall update the traceability matrix throughout the software life cycle phases.

5.4.5.5 Data Base Testing

It is not sufficient to test only the algorithm to verify the correctness of a program. It is also necessary to establish the correctness of the data base used by that program. This potentially involves review of four different areas:

- data accuracy,
- data completeness,
- data structure, and
- data accessibility.

Data accuracy review deals with the correctness of the individual data items stored in the data base.

Data completeness ensures that all the data which needs to be present is in fact present in the data base.

Data structure review deals with the analysis of the structure of the data base. It may include the ordering of the individual data items within the data base as well as the structuring for accurate and efficient searches or access.

Data accessibility reviews determine the extent to which the data items could be modified, intentionally or unintentionally. Methods for "data hiding", that limit the ability to modify data to known software items, are preferred. These methods protect software against unintended function brought on by unexpected changes to data made by unauthorized program functions. In contrast, global data techniques that result in unrestricted access and modification are undesirable.

5.5 LIFE-CYCLE VERIFICATION AND VALIDATION

5.5.1 Management of V&V

The management of V&V spans all life-cycle phases. Software development is a cyclic and iterative process. The V&V effort shall reperform previous V&V tasks or initiate new V&V tasks to address software changes. V&V tasks are reperformed if errors are discovered in the V&V inputs or outputs.

Management of V&V includes:

1. **Software V&V Plan:** Any deviations or project specific additions to the SVVP shall be defined in the Project Plan (Reference 1.6.4). This may include resources and schedule of the specific V&V activities.

2. **Baseline Changed Assessment:** Evaluate proposed software changes for effects on previously completed V&V tasks. When changes are made, plan iteration of affected tasks which includes reperforming previous V&V tasks or initiating new V&V tasks to address the software changes.

3. **Management Review:** Conduct periodic reviews of the V&V process in the area of technical accomplishments, resource utilization, future planning and risk assessment. Support daily management of V&V phase activities. Review final and interim V&V reports. Evaluate V&V results and anomaly resolution to determine when to proceed to the next life-cycle phase and to define changes to V&V tasks to improve the process.

4. **Review Support:** Support management and technical reviews (e.g., Software Requirements Review, Preliminary Design Review, Critical Design Review, etc.). Identify key review support milestones in SVVP and schedule V&V tasks to meet milestones. Establish methods to exchange V&V data and results with design team.

The costs of V&V shall be identified during the proposal (concept) phase of the project. The resources for performing the V&V shall be identified in the Project Plan (Reference 1.6.4) that is prepared by the Project Manager during the conception phase of the software life cycle.

5.5.2 Concept (Initiation) Phase V&V

Concept phase V&V is the period prior to formal definition of the system requirements, which may include a feasibility phase.

Project specific V&V planning, including schedule and personnel requirements should be developed at this time and incorporated in the Project Plan. Any specific tools to be used must be stated in the plan.

The conceptual design is based on the customer's bid specification, Westinghouse's proposal and the contract.

V&V Inputs

1. Feasibility Study (if applicable)
2. Customer's Bid Specification
3. Westinghouse's Proposal
4. Contract
5. Governing NRC regulations

V&V Tasks

1. Review Concept documents for consistency, incompatibilities, and compliance to regulations.
2. Identify major constraints of interfacing systems
3. Identify constraints or limitations of proposed system
4. Assess allocation of functions to hardware and software items
5. Assess criticality of each software item.

V&V Outputs

Reporting of the concept review activities will be incorporated in the Requirements Phase report, including identification of deficiencies.

5.5.3 Requirements Phase V&V

The intent of verifying the system (or functional) requirements is to ascertain that the requirements are complete, correct, consistent, clear, traceable, and testable. The selection of these evaluation parameters is based on their ability to assess the quality of the documented communication with the designer and also the user of the system.

The system requirements form the basis of all the system design and verification efforts, and are used throughout the rest of the product life cycle. They serve as the basis for the verification of design specifications which, in turn, are the basis for the verification of design implementation. System Requirements are the bases against which all the validation activities are performed.

The principal purpose of a requirements document is:

1. To clearly define the objectives and needs of the system design and development process. Both the designer and the user must be able to understand and perform a meaningful assessment of the system.

2. To serve as a means against which an implementation can be validated and the intermediate steps can be verified.

The goal of verification activities during this phase is to ensure that the requirements documents do indeed serve the above purpose.

In order to satisfy the need of both the V&V and designer to understand and evaluate the system, real-time system requirements should be stated in clear, concise, and understandable terms. Extraneous issues which are not requirements should not be in the SRS or it should be explicitly stated that they are for information only.

As a common practice, complex systems are systematically decomposed into smaller subsystems and their functions are assigned to either hardware or software. In some systems, in order to present a clear picture, decomposition may include data flow, control flow, and intricate synchronization and timing aspects and implicitly specify the software and hardware architectural requirements.

V&V Inputs

1. System Requirements Specifications
2. Interface Requirements Documents
3. Existing User documentation
4. Other documented requirements, such as:
    a. Design inputs
    b. Functional diagrams, wiring diagrams, etc.
    c. Historical design, test and development records
    d. Instrument configuration documents
    e. Acceptance test documents
    f. Qualification test reports

The Interface requirements document(s) should not be generated unless it is an explicit project requirement. The interface information can be stated in the SRS . The fewer the sources of requirements the less chance of error in creating and reviewing these requirements.

V&V Tasks

The major objectives of the verification activities during this phase are to:

1. Evaluate the adequacy of the allocation of system requirements to hardware, software, and subsystems.

2. Evaluate the feasibility of accomplishing the system objectives and goals with the assigned requirements and using the allotted processor resources.

3. Ensure design requirements are complete, accurate, testable, and unambiguous as possible.

4. Perform software safety requirements analysis

    a. Identify any hazards and software safety requirements

    b. Identify any software safety design constraints and guidelines

    c. Identify any software safety test requirements and provide inputs to the test planning process

    d. Identify any required, encouraged, discouraged and forbidden design, coding and test techniques

Verifying the system architecture and decomposition is one of the V&V tasks. The interrelationship between hardware/software and subsystems are reviewed by the verification team to ensure that the overall integrated system does indeed have potential to meet the system needs and objectives. Thus in addition to system requirements, hardware and software requirements, specifications are also verified for proper segmentation with appropriate and sufficient details to ensure system design and decomposition integrity.

The following are specific V&V Tasks:

1. Prepare the Requirements Traceability Matrix (RTM) to include all functional, hardware, software, performance, interface, and user requirements.

This RTM shall be a living document to be used throughout each phase of the V&V process. It is recommended that this document be kept in a word processing or database format for ease of update, however, the dated hard copy output shall be the V&V record.

2. Perform the following software safety analyses as described in Reference 1.6.26, Annex A.1:

   a. Criticality
   b. Specification
   c. Timing and sizing
   d. Different software system (if applicable)

3. Complete the Requirements Phase V&V checklist (Exhibit 5-2) as well as the appropriate checklist(s) in Reference 1.6.4 for Safety-Related software.

4. Other V&V review areas should include:

   a. Review requirements source documents - what is the basis of the requirements?

   b. Review system requirements - is the right problem being solved, are they consistent with emergency procedures, are the right plant parameters being monitored?

   c. Perform analysis of requirements decomposition - are subsystems defined with interface requirements noted?

   d. Review test requirements - what testing is needed and how will it be judged (i.e., what are the acceptance criteria)?

   e. Review data interface requirements - are data management requirements consistent with hardware requirements?

   f. Review human factors requirements - what will be done to see that the operator accepts this system and finds it useful?

5. Review requirements with respect to the following possible errors which are typical:

   a. Inadequate or partially missing performance criteria.
   b. Inadequate or partially missing operating rules (or information).
   c. Inadequate or partially missing ambient environment information.
   d. Requirements that are incompatible with other requirements.

e. Inadequate or partially missing system mission information.
f. Ambiguous or requirements subject to misinterpretation.
g. User's needs not properly understood or reflected.
h. Requirement not traceable to user's needs.
i. Requirements which cannot be physically tested.
j. Accuracy specified does not conform to the need.
k. Data environment inadequately described.
l. Input/output data parameters units incorrect.
m. Erroneous external interface definition.
n. Initialization of the system not properly considered.
o. Vague requirements of the functions to be performed.
p. Required processing inaccurate.
q. Required processing inefficient.
r. Required processing not necessary.
s. Missing requirements on flexibility, maintainability.
t. Missing or incomplete requirements of response to abnormal data or events.
u. Inadequate or incorrect algorithm.
v. Incorrect timing/synchronization requirements.
w. Incorrect hardware interface requirements.
x. Incorrect allocation of system resources.

6. Tools used in the development process (such as computers) do not require V&V as long as the resultant code is subject to V&V. Configuration management of these tools will be under the Software Configuration management plan Section 6.0.

7. Review previously-developed or subvendor software in the following areas and produce a report stating whether this software is adequate for its intended use. V&V review criteria should be similar to that for newly-developed software or the V&V plan shall state any deviations which are acceptable.

   a. The software used and its documentation shall be maintained and controlled during development, implementation, and testing. Procedures shall state how verification of the configuration is to be accomplished to assure that the software for testing is the same as that used for the final system.

   b. The software and its use shall be described in sufficient detail for an independent verification to determine the impact of using this software. This description would include the following:

      - Adequacy of the documentation (complete, unambiguous, and consistent with the software).

      - User interface with the software.

- Use of the software in development.
- What control the software has over the final output; e.g., is the software primarily used as a documentation tool or does it influence the exact software running in the delivered system.
- A description of how the software will be changed after installation; or if a tool, will be used to make change.
- User documentation.
- Test plans and test cases used to validate the software for acceptability.

c. A method of notifying the user if errors are discovered in use of this program after installation which may affect operation.

d. A determination of what, if any, additional documentation, testing, or reviews are required to validate the use of this software in the system development.

The V&V team may obtain the documentation required from the supplier or perform a documented review of the documentation at the supplier facility to determine acceptability. The installed base of software installed and operating in similar environments and also vendor records of changes repair may be considered by the V&V team in their review.

If the V&V team review of this software finds it acceptable, the V&V team shall insure that the Certificate of Conformance to be issued when the system ships to the client, certifies that the procured software (name, manufacturer, part/model number, revision) is acceptable for use.

V&V Outputs

1. Completion of Requirements Review Checklist #1 (Exhibit 5-2).

2. Produce a report on concept and requirements review activities, including identification of deficiencies.

3. Prepare Requirements Traceability Matrix and include all requirements identified in this phase.

5.5.4 Design Phase V&V

The purpose of design specifications verification is to ascertain that the design specifications are a faithful translation of the design requirements before the design is committed for implementation.

The design specification documents define and provide the details of the system design structure, information flow, processing steps and other aspects required to be implemented, in order to satisfy the system design requirements. The intent of the design specification verification is to ensure that the design specifications are clear and understandable, accurate, correct, consistent, complete, implementable, testable, and traceable to the design requirements.

Considering the inherent iterative nature of design activities, V&V tasks are conducted on an ongoing basis. This is highly desirable especially when V&V efforts parallel design activities. Test planning and verifying the conformance of design documentation to established standards are the major objectives of preliminary V&V activities. As the design progresses, the design as documented is analyzed and critically evaluated for its potential to meet design requirements.

V&V Inputs

1. Design documentation, including (as necessary for the project scope):
   a. Hardware design specification(s)
   b. Software Design Document(s)
   c. Interface design specifications
   d. User documentation, operator interface specifications
2. Requirements documentation from the previous phase.
3. Other standards and requirements.

V&V Tasks

1. Review system design documentation to ensure the system design completely and correctly performs the functions specified in the requirements documents.
2. Review hardware design documentation to determine that the hardware design specifications are understandable, unambiguous, reasonable, implementable, accurate, complete, and are a faithful translation of the hardware design requirements into hardware design specifications.

3. Review software design documentation to ensure design requirements are adequately incorporated. The design documentation shall address all software requirements and provide a correlation of the design elements with the software requirements.

4. Perform the following software safety design analyses as described in Reference 1.6.26, Annex A.2

   a. Logic
   b. Data
   c. Interface
   d. Constraint
   e. Functional
   f. Software element

5. Complete the Design Phase Checklist # 2 (Exhibit 5-3).

6. Update the Requirements Traceability Analysis.

V&V Outputs

1. Completion of Design Review Checklist #2 (Exhibit 5-3).

2. Produce a report on the design review activity, including identification of deficiencies and possible enhancements.

3. Follow-up as changes and corrections are incorporated into the requirements.

4. Preparation of a System Test Plan in accordance with Reference 1.6.14, Section 3

5. Updated Requirements Traceability Analysis.

5.5.5 Implementation Phase V&V

The purpose of the implementation verification is to ascertain the implementation documents are clear, understandable, logically correct and a faithful translation of the design specifications. The objectives of the implementation documents are to facilitate the effective production, testing, use, transfer, conversion to a different environment, future modifications, and traceability to design specifications. In general the verification activities during this phase are oriented towards evaluating the following:

1. Does the implementation satisfy design specifications?

2. Does the implementation follow established design standards?

3. Does the implementation follow established documentation standards?

4. Does the implementation serve production, test, use, transfer and other needs that motivated its creation?

5. What is involved in testing the actual resulting product?

V&V Inputs

1. Software/Hardware design documents.
2. Source code listings and executable code.
3. Interface design documentation.
4. User documentation.
5. Other standards and procedures.
6. Software Configuration Management Procedures.
7. Module Test Reports

V&V Tasks

1. The V&V team shall review the as-built system documentation to ensure the as-built system completely and correctly implements the design specified in the system design documents.

2. Perform the software safety code analyses described in Reference 1.6.26, Annex A.3

   a. Logic
   b. Data
   c. Interface
   d. Constraint
   e. Programming style
   f. Noncritical code
   g. Timing and sizing

3. Review module test reports and unit test reports (if applicable), and verify correct execution of critical software elements.

4. Review the code and associated database(s) for complete and correct implementation of the design. Complete sections of the implementation checklist # 3 (Exhibit 5-4).

5. Review the hardware/configured software integration procedure to ensure they are complete and correct. Complete Implementation Checklist #3 (Exhibit 5-4).

6. Update the Requirements Traceability Analysis.

7. Evaluate Software Configuration Management activities and ensure the requirements of Section 6 are fulfilled.

8. Hardware implementation review is normally conducted as part of the hardware quality assurance activities defined elsewhere.

9. For protection class software, review software testing records to ensure adequate structural testing[1].

10. Begin preparing test procedures in accordance with Reference 1.6.14, Section 6. Where special skills are required to prepare test procedures which are only present in design team personnel, then procedures shall be prepared by the design team and reviewed by the V&V team as part of the Test Phase.

V&V Outputs

1. Software Module Test.

2. Completion of Implementation Phase Checklist #3 (Exhibit 5-4).

3. Produce a summary report on Implementation Review activity, including identification of deficiencies and possible enhancements.

4. Follow-up as changes and corrections are incorporated into the implementation.

5. Updated Requirements Traceability Analysis.

5.5.6 Test Phase V&V

The verification process has provided an orderly step-by-step assurance of a true translation through the requirements, design, and implementation phases, each step being assessed upon the basis of the previous step. The integrated system validation process involves determining whether the system meets its functional requirements; e.g., functional operations, system level performance, external interfaces, internal interfaces, testability, and other requirements as stated during the definition phase. System validation evaluates the system performance in an environment which is real, or as close to real as can reasonably be

[1] Structural testing is testing that validates all branches of a software module

created; therefore, the fully integrated system with the actual system hardware and software is required. In large system applications, it may be required that validation testing begin at the subsystem level. Subsystem validation is usually desirable, to ease the error/failure isolation, even if not mandated.

The validation test environment must be configured to fit the system being tested. It should be matched to the available resources as much as practical to create the real operating environment.

The system validation process includes a Software Safety Test Analysis which demonstrates that safety requirements have been correctly implemented and the software functions safely within its specified environment. In some instances, system validation activities overlap those conducted earlier during verification and/or subsystem validation. Typical validation tasks are listed below:

1. The system functional operation is validated using the "black box" method; i.e., validating the system outputs by means of actuating prescribed inputs. Validation is conducted using the limits and ranges as designated in the system functional requirements, which are included in the system design requirements. The major validation areas shall be:

   - Functional operation
   - System level performance – demonstrates software's performance within overall system
   - External and internal interfaces – demonstrating that critical computer software units execute together as specified
   - Stress testing – demonstrates that the software will not cause hazards under abnormal circumstances
   - Regression testing – demonstrates changes made to the software do not introduce conditions for new hazards

2. Failure performance testing is executed on a functional operations basis.

3. Transient tests are executed to validate system functional operations.

4. Independent supplemental validation tests and scenarios as required to validate the system design.

5. System validation procedures are updated if required.

6. Final developer's documentation, to be:

   - Complete,
   - Accurate/compatible with delivered system, and
   - Compliant with standards.

7. Validation test results are evaluated to be:

   - Complete/consistent with procedures,
   - Traceable to functional requirements, and
   - Document results in Test Results Manual.

V&V Inputs

1. Source code and listings
2. Executable code
3. Applicable library routines
4. User documentation
5. Code analysis tools
6. Hardware environment as close to the installation configuration as possible

V&V Tasks

1. Verify program integration with the deliverable hardware per the Test Phase Checklists #4 (Exhibit 5-5) to insure that all aspects have been considered.

2. System validation test procedures shall be prepared in accordance with 1.6.14, Section 6, based upon the requirements of the design and shall include test cases encompassing the range of usage intended for the system. Test procedure(s) shall be either prepared by the design team and reviewed by the V&V team, or prepared by the V&V team and independently reviewed by the design team. The tests shall specify the following, as applicable:

   a. Identification of the test cases.

   b. Description of the test cases.

   c. Relationship of the test cases with the requirements, both functional and safety, and testing of all applicable program logic.

   d. Expected results of the test cases with acceptance criteria.

e. Special requirements or conditions for the test, such as hardware configuration, monitoring hardware or software, sequencing of tests, etc.

f. The simulation of the inputs shall be documented, including any special hardware or software required for these simulations.

g. Procedures to report errors found during testing, and acceptable means of retesting these errors after error correction has been performed. These procedures and error correction shall be independently verified in accordance with this V&V plan.

h. If the validation test procedure is prepared by the design group, the V&V team shall review to address the following:

- Is the test procedure description complete?
- Are the test problem definitions adequate and complete?
- Is each testable requirement adequately covered?
- Is the plan for evaluating and reporting test results adequate?

3. Perform validation testing in accordance with approved test procedures.

4. The system validation test(s) shall be documented in a report. The report can consist of a completed copy of the test procedure with all blank information completed. The V&V team shall review the report to insure that it includes the following, as applicable:

a. Computer software/PROM version tested.

b. Configuration of all hardware used (model number/serial number).

c. Test equipment used and calibration data, if applicable.

d. Date of test and personnel performing the test.

e. Test problems.

f. Results and acceptability.

g. Action taken in connection with any deviations noted. Errors and their correction shall be documented and V&V'd in parallel with change control procedures found in Section 6.

The system validation test report(s) shall also be reviewed by the V&V team to ensure that the following is addressed:

a. Do the test results comply with the format specified in the test procedure?

b. Do the test results provide an accurate statement of the testing performed?

c. Are the test results acceptable and auditable by persons not involved with the test?

Documentation of these reviews shall consist of completing the Checklist 4 (Exhibit 5-5).

5. Follow-up on changes and corrections made in the system in accordance with change control procedures in Section 6.

V&V Outputs

1. Test procedures.

2. Test Report and evaluation for acceptability.

3. Test Phase checklist #4 (Exhibit 5-5).

4. Produce a summary report on test phase V&V activity results, including identification of deficiencies and possible enhancements.

5. Code certificates certifying that the software is acceptable for use.

5.5.7 Installation and Checkout Phase V&V

The system installation package shall be reviewed to ensure that all elements necessary to install and operate the system have been correctly and completely specified.

V&V Inputs

1. Installation procedures, system generation procedures, etc.
2. User documentation

V&V Tasks

1. Review installation procedures and user manuals to ensure they are complete and correct.

2. Review training materials for the following:

   a. Safety training for the users, operators, maintenance and management personnel

   b. System startup training

   c. Safety training requirements are met

3. Prepare and issue the V&V report which provides:

   a. A listing of all V&V documentation produced. This documentation shall include records of the following reviews as a minimum: Hardware design requirements review; Software design requirements review; Audit results of previously-developed software; Configuration implementation review; Hardware/configured software integration review (if separate from validation testing); Test procedure/test report review; and Installation/checkout review. All reviews shall be conducted in a similar manner and have the following format (as a minimum):

      - Review summary
      - Recommendations (including any requirements for further reviews)
      - Detailed review comments and resultant actions

   b. A listing of deficiencies detected with corrective action taken.

   c. An evaluation of the system based upon the V&V.

   d. Comments and recommendations to aid in future system upgrades and development.

4. Complete the Installation and Checkout Phase V&V Checklist #5 (Exhibit 5-6).

V&V Outputs

1. Final V&V report with summary review of the system's acceptability.

5.5.8 Operation and Maintenance Phase V&V

Situations may arise after installation of a V&V'd computer system which may require the performance of additional V&V activities:

- Modifications are made in the hardware which may cause the software to be changed.
- Modifications are made to the program for enhancements.
- Errors may be discovered which require software modifications.

The V&V activities required for program modifications are identical to those previously discussed for new program development. However, if the program modification is such that it does not affect some phase of the V&V (for example, a code error might not affect the system requirements or design documentation), these areas of V&V may be omitted with a statement to this fact in the V&V report.

5.6 SOFTWARE VERIFICATION AND VALIDATION REPORTING

V&V reporting shall occur throughout the entire software life cycle and include the following (which have been identified in the software life cycle activities).

5.6.1 Required Reports

1. V&V phase summary reports: These reports are issued after each life cycle phase of the V&V task to summarize the V&V review. Phase summary reports may be consolidated into a single report if desired. These reports shall contain the following:

   a. Description of V&V tasks performed.
   b. Summary of task results.
   c. Summary of discrepancies and their resolution.
   d. Assessment of software quality.
   e. Recommendations

2. Discrepancy reports: These reports shall document each discrepancy found during the V&V reviews and include:

   a. Title, number, and revision of document reviewed.
   b. Section/Page reference location.
   c. V&V comment.
   d. Resolution with design team.

3. Final V&V Report: This report shall be issued at the end of the V&V task to summarize and document the V&V activities performed throughout all life cycle phases. The report shall include:

   a. Summary of life cycle V&V tasks.
   b. Summary of task results.
   c. Summary of discrepancies found and resolutions.
   d. Assessment of overall software and system quality.
   e. Recommendations for enhancements.
   f. Code certificate.

5.6.2 Optional Reports

Other reports may be produced as required to document special hardware testing activities, human factors reviews, etc. The format of these reports shall include purpose, approach, and summary of results as a minimum.

5.7 VERIFICATION AND VALIDATION ADMINISTRATIVE PROCEDURES

5.7.1 Anomaly Reporting And Resolution

Any discrepancies detected during any phase of the V&V process should be immediately brought to the attention of the design team and the Project Manager of the development. Resolution shall be made in writing by the design team. The V&V team must document the resolution in the V&V phase summary reports as well as the final V&V report.

5.7.2 Task Iteration Policy

If the V&V task must be reperformed, for whatever reason, the task must be identified in the reports produced identifying the rationale and the results of the V&V task. This information should be documented in a Revision Abstract for revised V&V reports.

5.7.3 Deviation Policy

If any deviation from the reviewed and approved V&V task plan is required, the change must be identified, rationale for the change provided, and a determination of effect on software quality provided. Any deviation must be approved by the V&V team leader and management.

5.7.4 Control Procedures

Procedures in this Software Program Manual (and those generated for specific Common Q subsystems as directed by this manual) for V&V and software development provide the controls for the activities associated with these efforts.

5.7.5 Standards, Practices, And Conventions

Specific standards, practices, and conventions for the V&V effort which differ from those stated in this procedure and its references shall be specifically stated in the project specific Project plan.

5.8 V&V TEST DOCUMENTATION REQUIREMENTS

The purpose of this section is to define the purpose, format and content of required test documentation. The test documentation as a whole shall fulfill the requirements of Reference 1.6.20.

5.8.1 Test Plan

The test plan documents the scope, approach, resources, and schedule for the testing activities of the project. It identifies the test items, the requirements to be tested, the testing tasks, and the required resources to perform these tasks.

5.8.2 Test Procedure

The test procedure contains the following components:

- Test-Design Specification
- Test-Case Specification
- Test-Procedure Specification

5.8.2.1 Test-Design Specification

This portion of the test procedure specifies the details of the test approach for a software requirement or combination of requirements, and identifies the associated tests.

5.8.2.2 Test-Case Specification

This portion of the test procedure specifies the inputs, predicted results and a set of conditions for executing the test case.

5.8.2.3 Test-Procedure Specification

This portion of the test procedure specifies a sequence of actions for the execution of a test.

5.8.3 Test Report

The test report summarizes the testing activities and documents the results. It also contains an evaluation of the corresponding test items. Typically the test procedure document containing the hand-written entries by the tester becomes a part of the document.

The test report also contains the test exception report log and copies of the test exception reports. Together these identify the status of outstanding test exceptions reported during testing.

5.9 SOFTWARE INTEGRITY LEVEL SCHEME

The software integrity level refers to the software classification described in Section 1 of this Software Program Manual. The Project Plan shall describe the agreed upon software classifications established for the system. The mapping of the software classifications in this manual to those of the IEEE Std 1012-1998 are as follows:

SPM Classification	*IEEE Std 1012-1998*
Protection	High
Important To Safety	Major
Important To Availability	Moderate
General Purpose	Low

6. SOFTWARE CONFIGURATION MANAGEMENT PLAN

6.1 INTRODUCTION

6.1.1 Purpose

Software Configuration Management (SCM) is the process for identifying software configuration items, controlling the implementation and changes to software, recording and reporting the status of changes, and verifying the completeness and correctness of the released software. SCM is intended to be utilized throughout the entire software life cycle, including requirements phase, design phase, implementation phase, test phase, installation and checkout phase, operation and maintenance phase, and retirement phase.

The intent of this document is to provide additional guidance and recommendations on employing SCM for Common Q software systems, and to adhere to industry guidelines on SCM defined in the Reference documents. These guidelines meet the intent of U.S. NRC Regulatory Guide 1.169, Configuration Management Plans for Digital Computer Software Used in Safety Systems of Nuclear Plants, September, 1997, for configuration management plans. This SCM Plan conforms with the requirements of IEEE Std 828-1990.

This document will also provide recommendations on the level of SCM required for various types of software development projects. When it is necessary for an individual software development effort to differ from these guidelines or add additional requirements, the project plan (Reference 1.6.4) should incorporate these changes or a separate configuration management plan may be developed.

The goals of software configuration management are to:

1. Record and document work in progress on each software item to permit understanding of current project status.

2. Identify all software code and data associated with a system including revision level, completion status, test status and history.

3. Maintain the association among software documents, code, and data.

4. Identify sets of software items that compose the system (baseline), test status and history, and readiness for release.

5. Maintain the status of released software, users of this software, and associated problem reports.

6. Maintain an association between software problem reports, change reports, and affected documentation, lines of code, and data items.

7. Implement appropriate controls and approvals for changes to the software configuration.

8. Identify the organization responsible for a software item and its associated problem reports and changes.

9. Document criteria for generation of software to release for use.

10. Ensure that existing and prior revisions of software can be reconstituted in the future.

11. Backup the software (in progress or completed) to protect against disaster.

12. Plan for controlling access to software and protecting against software viruses.

6.1.2 Scope

SCM shall be applied to all Common Q software and software tools used in the development of Common Q software. Software intended for limited use, such as in a single design analysis, may be used without employing SCM provided that the results as well as method and/or formulas are documented in the design analysis in sufficient detail to allow independent verification. An example of this is the use of Microsoft Excel to develop a design calculation.

All software items and associated documentation shall be controlled in such a manner as to maintain the items in a known and consistent state at all times. New software and modifications to existing software shall follow the configuration requirements for all life cycle phases. Existing software which is not to be modified, including tools used in the software development, test, and documentation process, shall fall under these configuration control procedures upon modification.

SCM shall be applied to software in any form, including (but not limited to):

1. magnetic tapes

2. magnetic disks

3. magnetic diskettes

4. optical disks and diskettes

5. non alterable devices such as Read Only Memories (ROMs) alterable devices such as Programmable Read Only Memories (PROMs), Electrically Alterable Read Only Memories (EAROMs), Electrically Programmable Read Only Memories (EPROMs), etc.

Documentation of the software, such as listings, drawings, specifications, etc., shall also be subject to configuration management in accordance with procedures for document and drawings control defined in Reference 1.6.4.

6.1.3 Definitions

Definitions for terms used in this guideline can be found in References 1.6.4 and ANSI/IEEE Standard 729-1983, IEEE Standard Glossary of Software Engineering Terminology. Frequently used acronyms are:

I/O	Input/Output
CEO	Cognizant Engineering Organization
SCA	Software Change Authorization
SCM	Software Configuration Management
SCR	Software Change Request
SDD	Software Design Description
SRS	Software Requirements Specification
SPR	Software Problem Report
TER	Test Exception Report
V&V	Verification and Validation

A *software item* is defined as the collection of source code modules, object code modules, database modules, etc. which comprise the software running in one identifiable computer. Since a system may have multiple processors performing different functions, a system may have multiple software items.

6.1.4 References

References are listed in Section 1.6 of this Software Program Manual.

6.2 MANAGEMENT

6.2.1 Organization

All software configuration management functions for a system are performed by the CEO responsible for a system. V&V activities related to configuration management are performed by member(s) of the V&V team. The Program Manager as described in Reference 1.6.4 shall be the CEO Manager.

6.2.2 SCM Responsibilities

The CEO responsible for a software item is responsible for implementation of adequate measures to manage and control the software configuration up to delivery and of the software, consistent with the guidelines defined herein. This responsibility for software within the operations and maintenance phase rests with the designated CEO.

Specific SCM responsibilities are defined in the following table in accordance with the software life cycle phases.

Software Life Cycle Phase	Responsibility
Requirement Phase	1. Define software items which are to be controlled via these SCM guidelines. 2. Establish requirements for the software. This shall be in the Software Requirements Specification (SRS). SRS requirements may be found in this Software Program Manual. 3. Establish V&V requirements. V&V requirements may be found in section 5 of this Software Program Manual. V&V activities for Safety-Related software is defined Reference 1.6.4. 4. Establish organizational responsibility for SCM activities. For large projects, a software librarian and/or system administrator may be named to perform the following activities: a. Maintain controlled software. b. Maintain records. c. Maintain backup copies of the deliverable software in a separate building for security and hazards prevention. d. Maintain backup copies of software tools used in

Software Life Cycle Phase	Responsibility
Requirement Phase (Cont'd.)	development, integration, and testing. 5. Establish necessary interfaces between software and hardware development personnel, external groups, subvendors, and the end user.
Design Phase	1. Prepare a Software Design Description (SDD) for the software item, consistent with the requirements in this Software Program Manual. 2. Prepare a draft high level test plan in accordance with Reference 1.6.14, Section3 and review to verify a "testable" design has been produced.
Implementation Phase	1. Code the software item in accordance with the requirements in this Software Program Manual. 2. Software shall be entered into a controlled access account once module testing is completed and the programmer is satisfied with the quality of the software. System testing is conducted from this controlled access account. The CEO manager shall designate who shall control the test system hardware/software configuration. 3. Prepare Factory Acceptance Test (FAT) procedure(s). 4. Begin preparation of user documentation and/or technical manual.
Test Phase	1. Freeze software/hardware configuration and document this configuration in the test procedure(s). This configuration then becomes the baseline. 2. Test system in accordance with written FAT procedure(s) and prepare Test Exception Reports (TERs) when anomalies occur. 3. Initiate Software Change Request (SCR) to document software changes or required enhancements. A SCR may be used to close several TERs. 4. Review test results and prepare a test report documenting the testing. 5. Document final software configuration in the test report and (if required) the V&V report.

Software Life Cycle Phase	Responsibility
Installation and Checkout Phase	1. Complete user/technical manual and installation documentation (if required). 2. Issue Computer Code Certificate and V&V report. Include Software Problem Report form (Exhibit 6-3) for user to document problems. 3. The CEO manager shall identify the person responsible for SCM activities for the Operations and Maintenance Phase and the Retirement Phase. 4. Update the Controlled Software Distribution List (Exhibit 6-2) as necessary. 5. Revise documentation, if required, to reflect the As-Built condition of the system.
Operations and Maintenance Phase	1. Use SCM procedures to document errors found by design engineering and by the user (Software Problem Report form (Exhibit 6-3)). 2. Control software changes made by design engineering using SCM procedures and issue revisions to the software and documentation. 3. Use Controlled Software Distribution List to identify recipients of any Computer Code Error Notification forms. 4. CEO manager provides updates to the Quality Records Controller on the software items for which they are responsible. 5. The CEO reviews subvendor software problem reports for subvendor software used in the delivered system to determine if any are applicable. If applicable, the problem should be identified, identified to users of the software (Computer Code Error Notification), and control software changes required to correct this error using the SCM procedures.
Retirement Phase	1. Software which is retired (superseded by a later revision, etc.) should be identified as such on the Controlled Software Distribution List and users should be notified in writing by the CEO that the software should not be used.

1. Configuration Identification Management – CEO responsible for the software item(s) is responsible for identification of all separately identifiable modules comprising the software item(s) in any form along with any required documentation.

2. Configuration Control Management - The CEO manager is responsible for management of SCM activities.

3. Configuration Status Accounting Management – CEO manager is responsible for collecting data and reporting of SCM activities within the design team, to external groups, and to the end user.

4. Management Reviews and Audits - The CEO manager is responsible to coordinate technical reviews within and external to the project team. Audits by Quality Assurance are coordinated through the Cognizant Engineering Organization Manager. External technical audits/reviews are coordinated through the CEO. External quality audits are coordinated through Quality Assurance department in conjunction with the CEO manager.

6.2.3 Applicable Policies, Directives and Procedures

The requirements of Reference 1.6.4 apply and take precedence to these procedures for all Safety-Related software.

Requirements for documentation and drawings control are found in Reference 1.6.4.

6.3 SOFTWARE CONFIGURATION MANAGEMENT ACTIVITIES

6.3.1 Configuration Identification

All software and documentation shall be uniquely identified. The identification structure shall also have the ability to track errors, resolution of errors, and software items which comprise a system or subsystem.

1. Documentation shall be identified and controlled in accordance with References 1.6.4.

2. Drawings shall be identified and controlled in accordance with References 1.6.4.

3. Software shall be identified in accordance with the following requirements, which depend on the format of the software.

   Source and object files for software items must be identified by a unique name (which is also defined in the SDD) a unique number, and a revision. For example, object files may be identified by a date time stamp. The CEO manager shall have the responsibility for defining the name/numbering system for a project. If the project specific SCM plan does not define software identification requirements, the following shall be utilized:

   Source File - The source file should contain a program header block which includes the following information:

   Project ####### Module NNNNNNN-VV-RR

   where:

#######	is the System number
NNNNNNN	is the Module name (up to 64 alphanumeric characters beginning with a letter, length depending upon the requirements of the file management system utilized)
VV	is the Version number (2 digit sequential number beginning with 00) for successive versions which implement revised software requirements.
RR	is the Revision number (2 digit sequential number beginning with 00 for successive versions which correct errors in the code and require no changes to the software requirements)

The header block should contain a complete revision history of the software item, including comments on each version and revision. In addition, the header block should contain the following information:

- Programmer
- Brief description of the program
- Date
- Other information as necessary in a comment field

For example, a typical header block in a source file might contain:

Project 2000000 Module CALCBLOC-00-01
Control Algorithm Calculation Subroutine
Copyright notice

Description: This program calculates control algorithm setpoint offset values from entered user input of setpoints.

Revision History:

Author:	Date:	Comments:
H. Kim	07-Jan-94	This revision implements SCR number SCR-2000000-018 to correct roundoff errors. It also corrects internal naming conventions and adds additional comment fields.
H. Kim	14-Dec-93	Baseline Version

4. Media, (The physical item containing software items) shall be labeled according to the following conventions:

Name
#######-VV-RR ZZZZ XXX of YYY

where:

Name	A four to 64 alphanumeric name of the software configuration item
#######	is the 7 digit project number
VV	is the two digit successive version number of the Product containing the software
RR	is the Revision number (2 digit sequential number beginning with 00 for successive versions which correct errors in the code and require no changes to the software requirements)

ZZZZ is the media type (DISK, TAPE, PROM, etc.) which may be more than four characters but not greater than eight characters

XXX Successive Media item number, beginning with 1

YYY Total media items of one media type

An example is:

2000000-00-01 DISK 1 of 2

5. Software System the collection of modules (object files, data files, etc.) representing the entire software for a product which may contain more than one computer is identified at the time of project baseline and updated for all changes to the software contained within. This shall be in the form of a list which is identified in the Factory Acceptance Test Report and is sent to the end user (with the Code Certificate and V&V Report) upon delivery of the product. This list shall contain the media the software is contained on and an overall product version number. Media identification shall also be provided. The following list is an example:

Common Q HJTC

End User: Utility
Product Version/Revision: 01/05

Media Identification	Software Item Identification
3.5" diskette labeled: 2000000-00-01 DISK 1 of 2	CONTROL1.exe / 01-05 CONTROL1.dat / 00-03 CONTROL1.src / 01-05
3.5" diskette labeled: 2000000-00-01 DISK 2 of 2	CONTROL2.exe / 01-05 CONTROL2.dat / 00-03 CONTROL2.src / 01-05

This list should also be on the Computer Code Certificate or may be attached to it (with indication that it is a multipage Code Certificate).

6.3.2 Configuration Change Control

All software and media related to a project are uniquely identified by a number which is guaranteed to be unique.

Software configuration controls are put in place by the CEO as soon as software development is initiated on a project. Configuration controls include:

1. Limiting access to master copies of media or documentation.

2. Placing duplicate (backup) copies of media in physically different location to protect against hazards such as fire. Creating regular backups of work in process to minimize hazard loss or loss due to hardware failures.

3. Using software tools to detect and eliminate software viruses.

4. Maintaining a master list of software placed under configuration control for any given project, which is updated until the product is shipped (and a Computer Code Certificate and V&V Report are issued).

5. Controlling the configuration of any support software or software tools used in the development, integration, testing, and documentation of the software system.

6. Control of previously developed software, purchased software, and NRC approved software is described in Reference 1.6.4.

Changes to a software item are controlled through the use of a System Change Request (SCR) as follows:

SYSTEM CHANGE REQUEST PROCEDURE

All changes to software performed after system baseline has been established (prior to beginning FAT) will be performed in accordance with the following steps:

Step 1: System Change Request Initiation

The requester of a change must complete a Software Change Request (SCR) as shown in Exhibit 6-1, completing the following:

1. Name of person requesting the change
2. Date
3. Software system affected
4. Modules affected
5. Documents affected
6. Reason for the Change
7. Description of the change

SCRs may be initiated by a Test Exception Report, Software Problem Report (SPR), or by request for enhancement.

Step 2: System Change Request Approval/Rejection

The SCR is routed to the following individuals for approval/rejection.

1. CEO Manager
2. Lead software engineer
3. Others as deemed appropriate by the PM/Program Manager

Each individual determines the feasibility and appropriateness of the change and signs the form for approval/rejection. Rejections must include an explanation for the rejection. Customer/User requests for changes must be approved by the PM or Program Manager.

Step 3: System Change Implementation

After approval of the SCR, the CEO will schedule the change and the personnel responsible for implementing the change. After implementation, the changed software and accompanying documentation will be submitted for inclusion in controlled system files and documentation.

Step 4: Revised System Baseline

The SCR forms will be used as the basis to track all system changes and to verify changes have been properly implemented and that documentation has been updated.

6.3.3 Configuration Status Accounting

Information on the status of documentation and software configuration items is to be maintained by the CEO Manager or an assigned project team member. This may be accomplished for simple projects by maintaining lists using commonly available word processing or spreadsheet programs or by Computer Aided Software Engineering (CASE) tools available on the development platforms. For larger projects, database programs may be utilized to simplify the maintenance process. In all cases, information on the status of documentation, software items, Test Exception Reports, Controlled Software Distribution, Error Notification, and Software Problem Reports shall be made available for use in reports. These reports when produced shall document the system status at any given time and be maintained by the CEO Manager for inspection by the customer/user and any auditors.

6.3.4 Configuration Audits And Reviews

1) V&V reviews are to be performed in accordance with this Software Program Manual V&V procedures or a project specific V&V plan.

2) Project and technical reviews are to be managed by the CEO manager in accordance with the project plan (Reference 1.6.4) and this Software Program Manual.

3) External audits are to be supported through the CEO manager who will schedule personnel to be available.

4) In-process audits shall be performed by the V&V team to verify the consistency of the design and for proper implementation of the software QA process. These shall be documented in the V&V report.

5) A functional audit shall be performed by the V&V team prior to shipment to verify the software configuration items actual functionality and performance is consistent with the Software Requirements Specification.

6) A physical audit shall be performed by the V&V team to verify that the as-built software and its documentation are complete, meet all project technical requirements, and that the software change control process was adequately followed.

7) Quality audits may be held at any time by Westinghouse Quality Assurance to ensure the software development guidelines, including configuration control, Verification and Validation, and Software Quality Assurance are being adequately executed.

All audits and reviews shall be documented by meeting minutes or formal report which will be tracked by the CEO Manager for resolution of outstanding issues.

6.3.5 Interface Control

The CEO Manager is responsible for coordination of communications and information transfer between the following entities to ensure that all interfaces external to the Common Q System is effectively controlled:

1. The project team and the customer.
2. The project team and subvendors/subcontractors.
3. Hardware, software, and functional engineering design personnel within the project team.

Interface communications external to the design team shall be documented with numbered and dated correspondence. Correspondence logs are controlled via reference 1.6.4. Interface between the design team and the independent V&V team shall also use written correspondence.

The hardware configuration which supports the documented software configuration for a deliverable computer system must be controlled using drawing control procedures identified in Reference 1.6.4. The hardware configuration supporting software tools shall be documented in the user manual.

Interface communications external to the CEO shall be documented. Interface between the CEO and the independent V&V team shall also use written correspondence.

The software requirements and design documents shall define the following for each external interface of the Common Q System:

a. Interface design
b. The organizations involved

The CEO is responsible for configuration control of CIs for the Common Q System side of the interface. All documentation on the interface, that was generated external to the CEO, shall be placed in configuration control.

6.3.6 Subcontractor/Vendor Control

6.3.6.1 Subcontractor Software

New Safety-Related software to be developed by a subcontractor shall meet the requirements of Reference 1.6.4 and shall be maintained by the subcontractor prior to shipment to Westinghouse using a SCM plan judged by the V&V team to be in accordance with the requirements of this SCM guideline. It is the responsibility of the CEO Manager to ensure SCM requirements are fulfilled by the subcontractor and of the V&V team to verify that the SCM guidelines are adequately employed by the subcontractor.

6.3.6.2 Vendor Software

Existing vendor software previously developed may be used "as-is" or modified prior to incorporation within the software system. This may include software which is in support of the delivered system or may be integral to the delivered system, such as operating systems, compilers, database software, etc.

Existing software which is not modified prior to delivery shall be evaluated to determine the adequacy of this software. The level of evaluation is determined by the following classifications:

- Development Tools (compiler, linker, loader, etc.) shall not require extensive V&V or testing to quality their use, since the end product is extensively tested and the tool is not used in on-line operation of the system.

- Software to be incorporated into the delivered product "as-is" or with modifications by design group is to be evaluated to determine the adequacy of this software for the intended application. This evaluation is to performed in accordance with the V&V guidelines documented in section 5.0.

6.4 SCM SCHEDULES

The project schedule shall include all SCM activities. The SCM activities in the schedule shall indicate their dependencies on other activities in the project. SCM milestones that shall be indicated on the project schedule include:

- Establishment of a configuration baseline,
- Implementation of change control procedures, and
- Configuration audit.

6.5 SCM RESOURCES

The CEO Manager shall identify in the appropriate tools, techniques, and methodologies which may assist in SCM activities. These may include commercially available products for code control, version identification, and media backup/control. If project specific tools, techniques and methodologies are not identified, the following are to be used (minimum requirements):

1. At project baseline, a list of software shall be maintained by the CEO Manager in the design file to include module name, version and revision, and executable file identification. In addition, a list of software tools (compilers, linkers, loaders, etc.) and their version/revision shall be maintained by the CEO Manager and kept in the design file. These lists may be maintained by commercially available word processing, spreadsheet, or database programs.

2. Software backups of all program files, including tools, shall be started upon system baseline and shall be updated on a regular basis, with changed files backed up on a weekly basis as a minimum. Backup methodology (saving all files or those which have changed in the last "x" days) shall be established by the CEO Manager. Backup files shall be kept in a separate building from the development location. Backups may be kept as read-only files on a computer network as long as the file locations are physically separate from the software development location.

Documentation is to be maintained physically and electronically in accordance with Reference 1.6.4.

6.6 SCM PLAN MAINTENANCE

The Quality Assurance department is responsible for monitoring that Common Q software design groups are adhering to this plan. This plan shall be updated when nuclear and industry standards for software configuration management have been changed. The CEO Manager shall evaluate the new standards and determine if this plan requires revision. If a revision is required then this plan shall be revised and approved by both the CEO Manager and the Quality Assurance department. The revised plan shall be distributed to all Common Q CEOs doing software design work.

7. SOFTWARE OPERATION AND MAINTENANCE PLAN

7.1 INTRODUCTION

To specify the requirements for the maintenance and use of computer software utilized for nuclear safety related functions. This plan uses Reference 1.6.15 as a guide.

7.2 PROBLEM/MODIFICATION IDENTIFICATION, CLASSIFICATION AND PRIORITIZATION

A four-level priority scale shall be used in the classification of software problems (refer to Exhibit 6-1). Metrics and measures for this phase are specified in Section 4.5.2.4.

7.2.1 Input

Input for the problem/modification and classification phase shall be a Software Change Request (SCR) (Exhibit 6-1). A description of the SCR process is found in Section 6.3.2.

7.2.2 Process

The SCR shall specify:

1) An identification (SCR) number

2) A classification number identifying the maintenance type and prioritization

3) A description of the software modification that describes the magnitude of the change.

The SCR is submitted to the CEO lead engineer for technical analysis and the CEO Manager for approval. They can accept/reject the SCR or request further clarification. If the SCR is approved, then the modification is scheduled by the CEO manager.

7.2.3 Control

An SCR log shall be maintained for the specific Common Q system implementation. The CEO Manager shall ensure that the approved SCR is entered into this log.

7.2.4 Output

The approved SCR is the output to this process. The original problem report shall be attached to the SCR if applicable. The CEO manager should be provided an estimate for the modification as input into the next phase.

7.3 ANALYSIS

This phase of Software Operation And Maintenance involves a feasibility and detailed analysis of the modification. If the modification is a correction to an error and the requirements remain the same, this phase of software maintenance may not be applicable.

7.3.1 Analysis Input

Input to the analysis phase of the maintenance process shall include:

1) Validated SCR
2) Entry of the SCR into the SCR log
3) Any relevant project or system documentation

7.3.2 Analysis Process

This section specifies the process requirements for analyzing the modification.

7.3.2.1 Feasibility Analysis

If the scope of the modification requires it as defined by the CEO project management manual, a project plan (Reference 1.6.4) shall be developed. In addition to the required information, the project plan should address the following if applicable:

1) Impact of the modification
2) Alternate solutions
3) Analysis of conversion requirements
4) Safety and security implications
5) Human factors
6) Costs
7) Value of the benefit of making the modification
8) How the design, implementation, testing and delivery of the modification is to be accomplished with minimal impact to current users.

7.3.2.2 Detailed Analysis

If the modification is a change to existing requirements, then firm requirements for the modification are defined in revised System and/or Software Requirements Specifications. The SRS shall identify the software elements that require modification. Any safety and security issues shall be identified in these documents.

During this phase a test plan shall be developed in accordance with Reference 1.6.14, Section 3 that specifies the test strategy for the modification including any regression testing requirements. For protection class software, the test plan shall address any requirements for module testing.

If necessary the project plan shall be updated to reflect any changes to the planned implementation (design, implementation, testing and delivery) of the modification such that current users are minimally impacted (see Section 7.3.2.1).

7.3.3 Analysis Control

At this phase of the analysis, the V&V team shall review any changes to the requirements specifications and review the test plan(s) as defined in Sections 5.5.4 and 5.5.8.

The relevant version of project and system documentation from the appropriate configuration control organization (CEO or customer) shall be retrieved (refer to Section 6 for Software Configuration Management). The CEO shall review the proposed changes and newly revised requirements specifications. The CEO shall then consider the integration of the proposed change within the existing software.

The project plan shall be reviewed for any changes to the risk analysis after the CEO reviews the proposed changes and revised requirements.

7.3.4 Analysis Output

The output of the analysis phase of software maintenance includes:

1) Project Plan
2) Revised System and/or Software Requirements Specifications
3) Test Plan
4) V&V Requirements Phase Report including RTM

7.4 DESIGN

This section defines the design requirements for software maintenance. Any metrics for this phase are defined in Section 4.5.2.4. If the modification does not effect the design of the software, then this phase of software maintenance may not be applicable.

7.4.1 Design Input

All outputs from the identification and analysis phases are used as inputs into this phase of software maintenance.

7.4.2 Design Process

At this phase the affected software modules are identified and the SDD is revised to incorporate the modification into the design.

For protection class software, module test procedures are created/modified in accordance with the test plan and Reference 1.6.21. Unit and integration test procedures (with test cases) are developed in accordance with Reference 1.6.14, Section 6 to test the modification in accordance with the test plan.

At this phase, the CEO shall identify any installation or user documentation that must be revised to incorporate the modification.

7.4.3 Design Control

The V&V team shall review the revised SDD as defined in Sections 5.5.4 and 5.5.8 and the test procedures for the modification as defined in Sections 5.5.6 and 5.5.8.

7.4.4 Design Output

The output of the design phase of software maintenance shall include:

1) Revised SDD
2) Test Procedures
3) Design Phase V&V Report including Requirements Traceability Matrix

7.5 IMPLEMENTATION

This section defines the requirements for the implementation phase of software maintenance. Any metrics for this phase are defined in Section 4.5.2.4.

7.5.1 Implementation Input

The inputs to the implementation phase shall include all outputs from the identification, analysis and design phases.

7.5.2 Implementation Process

The implementation phase shall include the following subprocesses.

7.5.2.1 Coding and Module Testing

At this phase the source code is modified and compiled, and new executables generated. For protection class software, module test procedures are run and results documented. For other software classes, informal module testing is conducted. The V&V activities related to module testing for protection class software is performed in accordance with Sections 5.5.6 and 5.5.8.

7.5.2.2 Integration

Integration is the process of running the revised software in an integrated system environment. It includes informal integration and regression testing to ensure the system as a whole is fully operational prior to system testing. Any anomalies shall be documented using the TER form and changes shall conform to the software configuration management plan in Section 6.

7.5.2.3 Documentation

Any user, training or installation documentation that is impacted by the modification shall be revised at this time. It shall be submitted to the V&V team for review per Section 5.5.5.

7.5.2.4 Risk Analysis And Test-Readiness Review

The CEO Manager shall review the status of the integration and determine when the software is ready for official system testing. In addition, the project plan shall be updated if the risk assessment has changed.

7.5.3 Implementation Control

The V&V activities associated with the implementation phase of the software life cycle as defined in Sections 5.5.5 and 5.5.8 shall be performed to ensure implementation control. The CEO Manager shall ensure that all software is under software configuration management control in accordance with Section 6.

7.5.4 Implementation Output

The outputs of the implementation phase of software maintenance shall include:

1) Updated software
2) Updated module test procedures
3) Updated user, training, and installation documentation
4) Implementation Phase V&V report

7.6 TEST

At this phase, formal testing is performed on the new software system.

7.6.1 Test Input

All outputs from the previous phases are used as inputs into this phase of software maintenance.

7.6.2 Test Process

During this phase the V&V team revises or develops new validation test procedures with test cases to test the modification in accordance with Reference 1.6.14, Section 6. If the design team developed the test procedures, then the V&V team reviews the test procedures per Section 5.5.6.

After the test procedures have been V&V'd, the validation tests are performed on the new software system according to the test plan. Any test exceptions shall be documented using the TER form and changes shall conform to the software configuration management plan in Section 6.

After the completion of the validation test, a test report shall be issued and reviewed in accordance with Section 5.5.6.

7.6.3 Test Control

Validation tests shall be conducted by the V&V team or by individuals not associated with the design of the system. Any test exceptions shall be documented using the TER form (Exhibit 9-1) and changes shall conform to the software configuration management plan in Section 6. The test report shall be issued and reviewed in accordance with Section 5.5.6.

7.6.4 Test Output

The outputs for the validation test phase of software maintenance are the same as the test phase V&V outputs specified in Section 5.5.6.

7.7 DELIVERY

This phase of software maintenance is the final acceptance of the modification prior to shipment to the customer. All metrics have been collected in accordance with Section 4.5.2.4.

7.7.1 Input

The inputs to this phase of software maintenance include the outputs from all previous phases.

7.7.2 Process

A physical configuration audit on the new software system shall be performed according to Section 4.6.2.6. The users of the software shall be notified in accordance with Section 9. An archival version of the software shall be performed accordance with Section 6.

7.7.3 Control

In addition to the physical audit, the V&V team shall perform the activities associated with the Installation and Checkout Phase, Section 5.5.7.

7.7.4 Output

In addition to the modified software, the outputs for the delivery phase of software maintenance include a final V&V report and Code Certificate.

8. DOCUMENTATION

8.1 GENERAL REQUIREMENTS

Software documentation shall be provided for all computer software to be used or delivered for Common Q systems. All documentation shall comply with Reference 1.6.4.

8.2 SYSTEM REQUIREMENTS DOCUMENT

For a Common Q system the System Requirements are composed of: Functional Requirements and Software Requirements. The Software Requirements section of the System Requirements Document shall conform to the requirements in Reference 1.6.6.

The Functional Requirements Specification and the Software Requirements Specification may be developed as separate documents to fulfill the requirements of the System Requirements Document.

Each requirement in the System Requirements Document shall be defined such that its achievement is capable of being verified by the SVVP.

8.2.1 System Requirements

The System Requirements include:

- System Operational Requirements
- System Performance Requirements
- System Safety Requirements
- System Design Basis
- System Design Constraints

The System Requirements define high level system requirements identifying those functions that will be performed by software and specifying the software safety-related actions that are required to prevent the system from entering a hazardous state, or move the system from a hazardous state to a nonhazardous state, or to mitigate the consequences of an accident.

8.2.2 Software Requirements

The Software Requirements are developed according to Reference 1.6.6 as augmented by Reference 1.6.22 and is used as the source document for design of the software, including:

1) Description of major software components which reflect the software requirements

2) Technical description of the software (i.e. control flow, data flow, control logic, data structures)

3) Description of all interfaces and allowable ranges of inputs and outputs

4) Any other design items which must be translated into code

5) A description of the intended platform and programming language(s) expected to be utilized

6) Data necessary for final implementation such as setpoints

7) Abnormal conditions to be accommodated by the software shall be described, including resulting functional operations.

8) Plant input signal transient conditions to be accommodated by this software shall be described.

9) Identify software safety requirements that, as a minimum, address System Safety Requirements.

8.3 SOFTWARE DESIGN DOCUMENT (SDD)

The software design document is prepared according to Reference 1.6.7.

The purpose of the SDD is to depict how the software will be structured to satisfy the requirements of the SRS, including software safety requirements. The design shall be described such that it can be translated into software code.

The SDD is a detailed description of the software to be coded. It describes decomposition of the software into entities. Each entity is described by its type, purpose or function, subordinate entities, dependencies, interfaces, resources, processing and data.

Each design feature shall be described and defined, and each software safety design element identified that satisfy the software safety requirements, such that its achievement is capable of being verified and validated per the SVVP. The adequacy of the SDD shall be verified against how the requirements of the software (documented in the SRS) are to be implemented in code, and how the design is traceable to the requirements in the SRS.

8.4 SOURCE CODE DOCUMENTATION

Source code documentation shall include source code listings which reflect the translated design representation into a programming language. Also, any associated documentation generated during the coding, module testing, and unit testing process such as code reviews, test procedures, test cases or test results should be referenced in the source code documentation.

Source code shall be traceable to the software design documented in the SDD and the requirements in the SRS. It shall include sufficient comments to provide the user of the source code with an understanding of the functioning and programming of each module. All source code, whether developed or modified from existing software, shall be documented in accordance with the coding standards listed in Section 4.5.2.1.

8.5 SOFTWARE VERIFICATION AND VALIDATION DOCUMENTATION

Software V&V documentation shall include Software V&V Reports (SVVR), prepared according to Reference 1.6.8 as augmented by Reference 1.6.18.

8.5.1 Software Verification and Validation Plan

The Project Plan shall identify the software items to be evaluated. The SVVP, Section 5, describes the V&V evaluation and reporting activities. Verification review requirements and guidelines are described in Section 4.6 and Section 5. Validation tests to be performed shall be described in a separate Test Plan which is subordinate to the SVVP, and is included as part of the software V&V documentation.

For custom software to be developed, the project specifics for V&V shall be documented in the Project Plan.

For existing software to be modified, the Project Plan includes methods for verifying and validating modifications to this existing software.

The Project Plan shall provide adequate planning for the following, referencing Section 5 as appropriate:

- Software V&V process for the various software categories described in Section 4.1.1
- Software V&V process for existing software to be modified and to be used "as-is".
- Software V&V process for prototype software

The Project Plan shall also define the tracking and recording process for the hardware configuration pertinent to the software verification and validation process during all phases of the software life cycle.

8.5.2 Software Verification and Validation Report

A V&V file shall be maintained by the CEO throughout the software life cycle to document all V&V activities. It shall be an ongoing compilation of all validation test results, problem reports and corrective actions (Section 9.0), verification review results and phase reports (Section 4.6), and the results of all NQA audits (Section 4.6.2.7). This file shall form the basis for the development of a final SVVR upon installation and checkout life cycle phase.

The final SVVR shall be developed by the CEO in accordance with Section 5.5.7.

8.6 USER DOCUMENTATION

User documentation is prepared according to Reference 1.6.9. The purpose of User Documentation is to provide sufficient information about the software to permit users to employ the code as it was intended. It shall be written by the design team. User documentation will be developed to the extent practical during the Test Phase and delivered to the user during the Installation and Checkout Phase.

User documentation shall consist of vendor documents and documents prepared as part of the project. Project prepared user documents shall be as follows:

- User's Manual
- Installation and Operations Manual
- Maintenance Manual

User Documentation shall include all error messages and identify the necessary corrective-action procedures. Also, it shall provide the means for the user to report problems to the CEO group or software maintenance group.

8.7 SOFTWARE CONFIGURATION MANAGEMENT DOCUMENTATION

Project-specific SCMP details, such as the identification of specific SCM tools, shall be defined in the Project Plan.

8.8 TEST DOCUMENTATION

This section describes the requirements for test plans and test procedures.

8.8.1 Test Plans

The requirements for test plans can be found in Section 4.3.2.2.

8.8.2 Test Procedures

The requirements for Common Q module, unit, integration and system test procedures can be found in Section 5.5.6, page 88.

8.9 COMPUTER CODE CERTIFICATE

The completion of the Implementation and Checkout Phase Software Verification and Validation report is the basis for the issuance of a Computer Code Certificate, Exhibit 8-1.

Computer Code Certificates are issued for safety-related software only. It shall identify the software classification of each software component listed on the certificate.

The issuance of a Computer Code Certificate allows the release of a configuration item for use in a safety-related application.

Software intended for limited use, such as in a single design analysis, may be used provided that the results as well as methods and/or formulas are documented in the design analysis in sufficient detail to allow independent verification. A Computer Code Certificate shall not be issued for such software on this basis alone.

9. PROBLEM REPORTING AND CORRECTIVE ACTION

9.1 INTRODUCTION

There are two modes of problem reporting. The first is during the software development phase when validation testing is being performed and test exceptions are found. Section 9.2 describes the reporting process for these errors.

The second mode of error reporting occurs when a user or CEO discovers an error after a software release. Section 9.3 describes this reporting process.

9.2 ERROR REPORTING BEFORE SOFTWARE RELEASE

Discrepancies, deficiencies, or comments identified as a result of testing, review, or other means shall be documented in a formal manner. This includes any general discrepancies found outside of the normal V&V test process. The following table illustrates the type of report required by each method:

METHOD	REPORT
Verification Reviews	V&V Reports
Validation Tests	Test Exception Report (TER) (Exhibit 9-1)
General Findings	Test Exception Report (TER) (Exhibit 9-1)

The appropriate configuration identification data (see Section 6.3.1) for each deficient software item or document shall be included on the appropriate form (or report). The form (or report) shall also include a description of the observed deficiency, the name of the individual reporting the deficiency, and the date of the report finding.

In the case of a TER, each form shall include space for a description of the resolution and any retest or review required after the resolution. If retest is performed, a copy of the test procedure or test case used shall be attached or referenced in the completed TER. Copies of all completed TER forms shall be distributed by the independent reviewer to CEO management and placed under configuration management per the system specific SCMP. All TER forms shall also be placed in a lifetime records per reference 1.6.4. The steps taken to cause the discrepancy to occur should also be included on the TER form in order to reproduce the problem. These steps should be noted as best as possible if the problem is not repeatable.

The extent of the retest shall be determined by the CEO group based on the relative impact of the software change on the overall system operation. For Protection and Important to Safety software, all changes require complete system or function retest, unless otherwise justified in writing including steps to ensure that new errors were not introduced.

9.3 ERROR REPORTING AFTER SOFTWARE RELEASE

Software errors found after the software Code Certificate has been issued may be reported using the Software Problem Report (SPR) form described in the SCMP, or the CEO Manager may receive error notification in other forms from the customer. These errors are evaluated, documented and reported to other users and the software developers by the CEO Manager using the Computer Program Error Notification form (Exhibit 7-1). For safety-related software, error notification procedures defined in Reference 1.6.4 shall be followed. The Computer Program Error Notification form is used to report the error to all users, who then return the Computer Program Error Notification Response Receipt (Exhibit 7-2). When the error impacts safety related software or safety related designs using the software, then the user is responsible for documenting appropriate action as necessary, including 10CFR21 considerations.

9.4 CORRECTIVE ACTION

The CEO shall establish as a clear objective the goal of resolving all validation test problems (via TERs), verification review comments, and reported errors (via SPRs) expeditiously to minimize the potential for unidentified effects during later life cycle phases.

The corrective action procedures used shall be based on the level of problem reported.

In addition, the CEO shall adhere to the following corrective action methodology that:

- Problems are identified, evaluated, documented and, if required, corrected by the appropriate reporting mechanism (Section 9.1 and 9.2).
- Corrections or changes shall be controlled in accordance with the SCMP (Section 6.3.2).
- Preventive actions and corrective actions are documented on the appropriate form and distributed to the cognizant CEO group.

Corrective actions shall be documented on TERs and V&V responses by the cognizant engineer and shall be completed by the due date specified on the form. If a resolution is not received by the specified due date, the problem shall be escalated to the next level of CEO management. This escalation process is continued through each higher level of CEO management until a response is received. Once the independent reviewer is satisfied with the corrective action taken, the report form shall be signed off and entered into the report form central log.

EXHIBIT 1-1 SOFTWARE PROGRAM DOCUMENTATION HIERARCHY

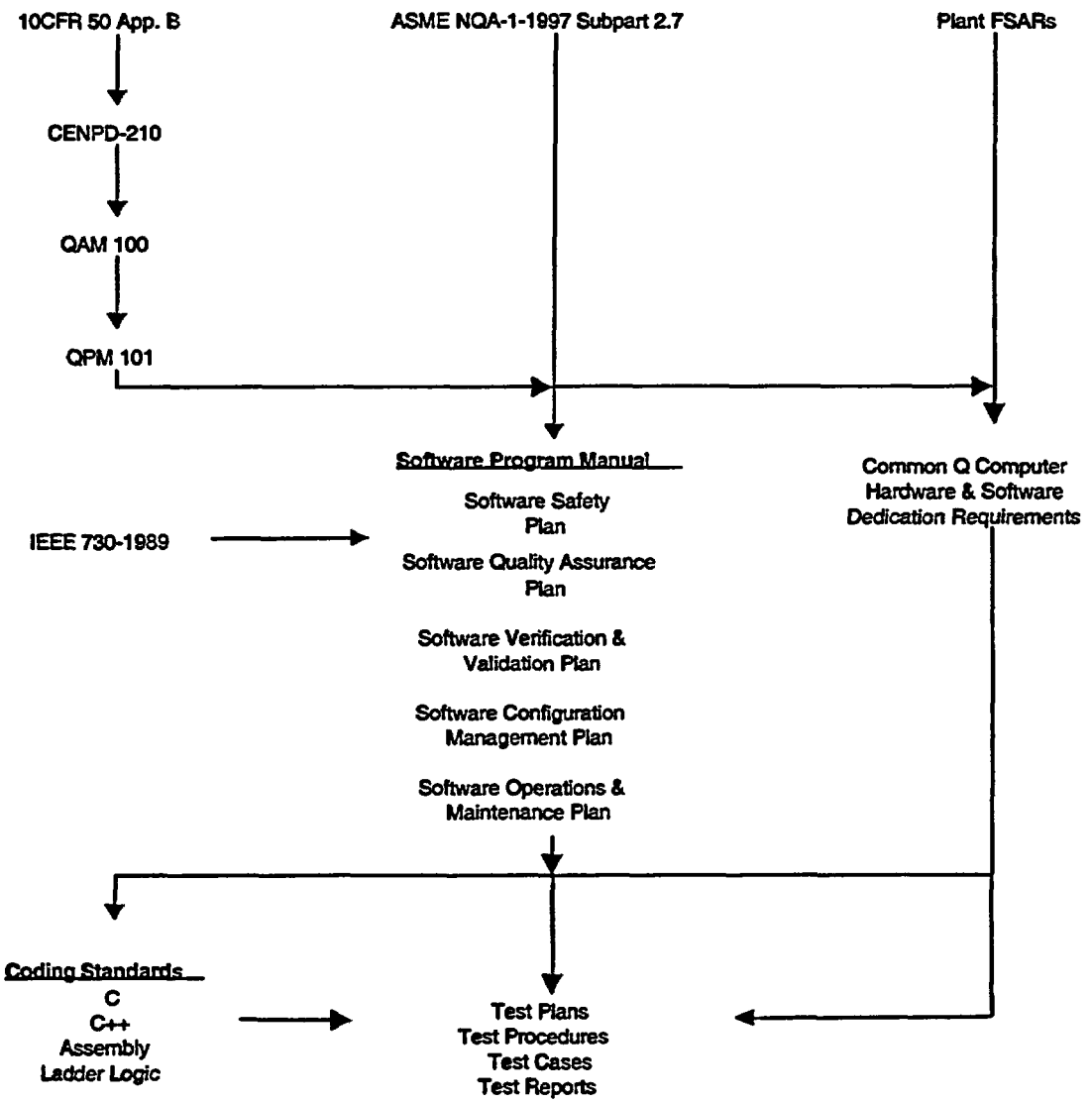

EXHIBIT 2-1 DESIGN/V&V TEAM ORGANIZATION

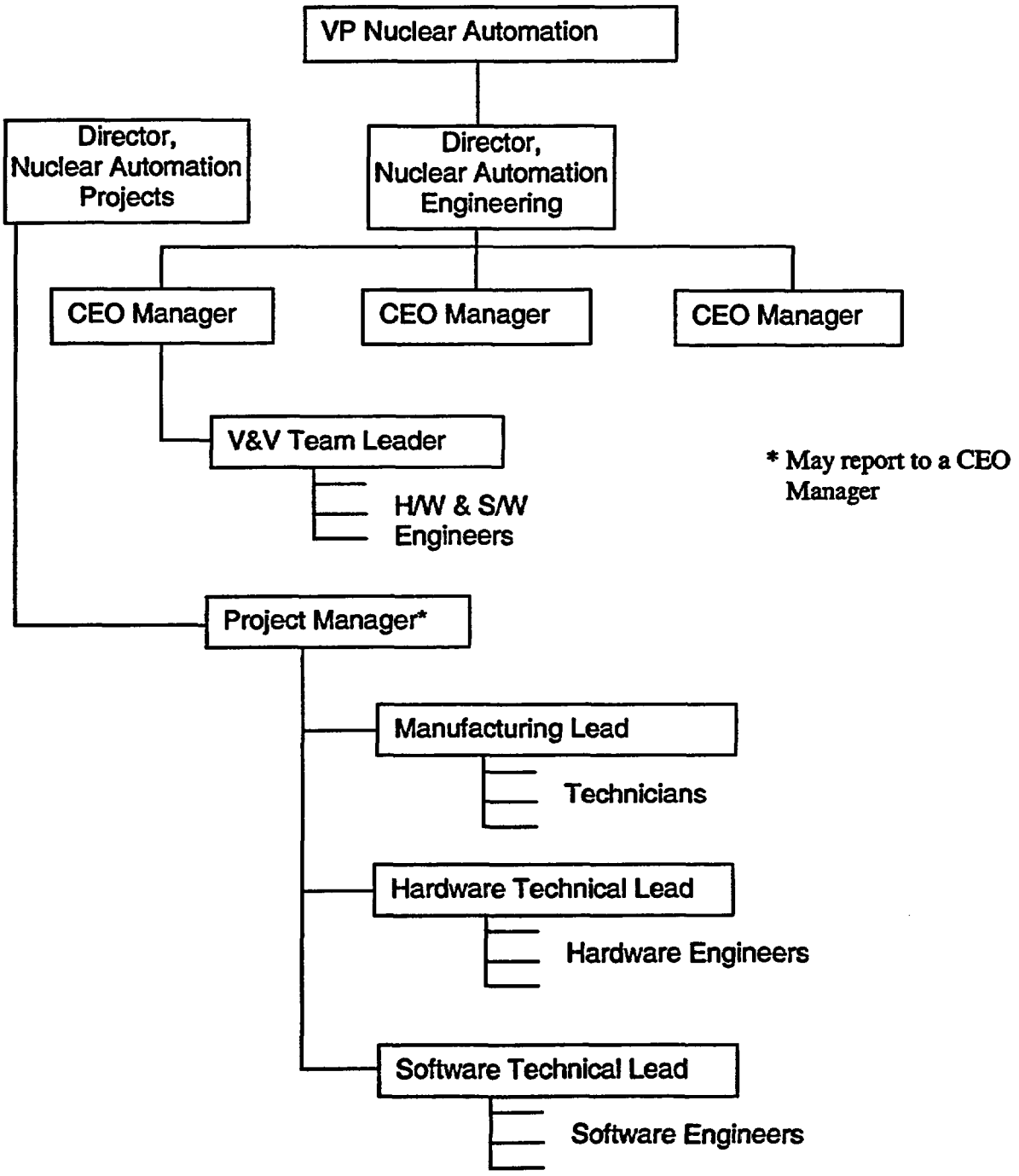

EXHIBIT 4-1 ASSIGNMENT OF COMMON Q SOFTWARE TO CLASSES

SYSTEM	SUB-SYSTEM SCOPE	CLASS
Plant/Reactor Protection System (PPS/RPS)	Safety Critical Kernel (LCL, Bistable, CCC)	Protection
	Maintenance and Test Panel (MTP)	Important To Safety
	Operator's Module	Important To Safety
	Interface and Test Processor (ITP)	Important To Safety
	Inter-System Communication Software	Important To Safety
	All Other Software	General Purpose
	Development Tools	General Purpose
Engineered Safety Features Actuation System Auxiliary Cabinet	Safety Critical Kernel (Pump and Valve)	Protection
	MTP	Important To Safety
	CIP Software	Important To Safety
	Inter-System Communication Software	Important To Safety
	All Other Software	General Purpose
	Development Tools	General Purpose

SYSTEM	SUB-SYSTEM SCOPE	CLASS
Core Protection Calculator	Safety Critical Kernel (FLOW, UPDATE, POWER, STATIC)	Protection
	CEAC Software	Protection
	MTP	Important To Safety
	Operators Module	Important To Safety
	Inter-System Communication Software	Important To Safety
	CEAPDS	Important to Availability
	All Other Software	General Purpose
	Development Tools	General Purpose
PAMS	Kernel Software (CET, SM, RVL monitoring)	Important To Safety
	Flat Panel Display System	Important To Safety
	Inter-System Communication Software	Important To Safety
	All Other Software	General Purpose
	Development Tools	General Purpose

EXHIBIT 4-2
Software Class and Category Worksheet

System:
Module Class (circle):
Protection Important to Safety General Purpose
Module Scope Description:_____

_____

_____

_____

Software Item	Check Applicable Box		
	Original	Existing, to be Modified	Existing, not to be Modified

Prepared By: _____ Date: _____

Reviewed By
V&V Team: _____ Date: _____

Approved By
Team Leader: _____ Date: _____

EXHIBIT 4-3
TYPICAL COMMON Q SOFTWARE DEVELOPMENT PROCESS

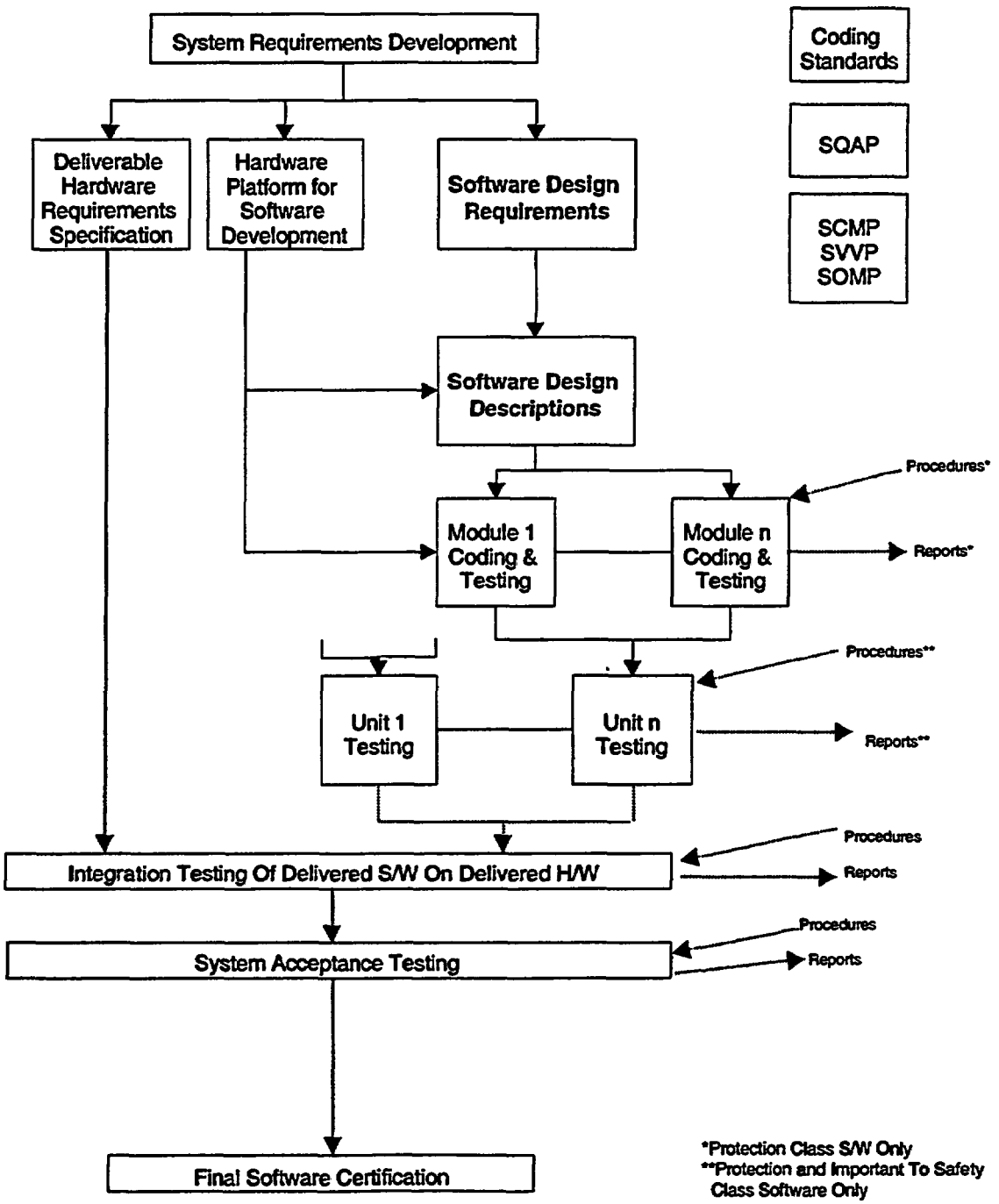

EXHIBIT 4-4: TASKS REQUIRED FOR SOFTWARE CATEGORIES

TASK	ORIGINAL SOFTWARE	ETBM SOFTWARE	ENM SOFTWARE
SQA PLANNING PHASE			
SOFTWARE QUALITY ASSURANCE PLAN	X	X	X
CODING STANDARDS	X	X	
SOFTWARE VERIFICATION AND VALIDATION PLAN	X	X	X
SOFTWARE CONFIGURATION MANAGEMENT PLAN	X	X	X
SOFTWARE REQUIREMENTS PHASE			
SYSTEM REQUIREMENTS	X	X	X
PROTOTYPE CODING	As Required		
SOFTWARE REQUIREMENTS	X	X	X
SOFTWARE DESIGN PHASE			
SOFTWARE DESIGN DESCRIPTION	X	X	
REQUIREMENTS TRACEABILITY ANALYSIS	X	X	X
SOFTWARE IMPLEMENTATION PHASE			
MODULE CODING	X	X	
TEST PLAN	X	X	X
MODULE TEST PROCEDURE (Protection)	X	X	
MODULE TEST EXECUTION	X	X	
MODULE TEST EXECUTION REPORT (Protection)	X	X	
UNIT TEST PROCEDURE (Protection and Important to Safety)	X	X	X
UNIT TEST EXECUTION	X	X	X
UNIT TEST REPORT (Protection and Important to Safety)	X	X	X
REQUIREMENTS TRACEABILITY ANALYSIS	X	X	

EXHIBIT 4-4: TASKS REQUIRED FOR SOFTWARE CATEGORIES (Continued)

TASK	ORIGINAL SOFTWARE	ETBM SOFTWARE	ENM SOFTWARE
TESTING PHASE			
SYSTEM TEST PROCEDURE	X	X	X
SYSTEM TEST EXECUTION	X	X	X
SYSTEM TEST REPORT	X	X	X
INTER-SYSTEM TEST PROCEDURE	X	X	X
INTER-SYSTEM TEST EXECUTION	X	X	X
INTER-SYSTEM TEST REPORT	X	X	X
USER DOCUMENTATION	X	X	X
SITE ACCEPTANCE TEST PROCEDURE	X	X	X
SITE INSTALLATION AND CHECKOUT			
SITE ACCEPTANCE TEST EXECUTION	X	X	X
SITE ACCEPTANCE TEST REPORT	X	X	X
FINAL V&V REPORT	X	X	X
OPERATION AND MAINTENANCE			
MAINTAIN SOFTWARE	X	X	X
RETIREMENT	X	X	X

ETBM - Existing Software To Be Modified
ENM - Existing Software Not To Be Modified

EXHIBIT 4-5
COMMON Q COMMENT RECORD

DOCUMENT NUMBER:		
ITEM NO.	COMMENT	COMMENT RESOLUTION
Fill in on last page only:	COMMENT ORIGINATOR: _____ _____ Signature Date	RESOLVED BY: _____ _____ Signature Date

EXHIBIT 5-1: SOFTWARE TASKS AND RESPONSIBILITIES

TASK	PROTECTION	IMPORTANT TO SAFETY	IMPORTANT TO AVAILABILITY	GENERAL
SOFTWARE REQUIREMENTS PHASE				
SYSTEM AND SOFTWARE REQUIREMENTS	DT/VT	DT/VT	DT	DT
REQUIREMENTS VERIFICATION	VT	VT	DT	DT
SOFTWARE DESIGN PHASE				
SDD	DT/VT	DT/VT	DT	DT
PROTOTYPE CODING	DT	DT	DT	DT
DESIGN VERIFICATION	VT	VT	N/A	N/A
SOFTWARE IMPLEMENTATION PHASE				
TEST PLAN (MAY BE PART OF SVVP)	VT/DT	VT/DT	DT	DT
MODULE CODING	DT	DT	DT	DT
MODULE TEST PROCEDURE	DT/VT	N/A	N/A	N/A
MODULE TEST EXECUTION	DT/VT	N/A	N/A	N/A
UNIT TEST PROCEDURE	VT/DT*	VT/DT*	N/A	N/A
UNIT TEST EXECUTION	DT/ VT	DT/VT	N/A	N/A
IMPLEMENTATION VERIFICATION	VT	VT	DT	DT
TESTING/INSTALLATION & CHECKOUT PHASES				
SYSTEM TEST PROCEDURE	VT/DT*	VT/DT*	DT	DT
SYSTEM TEST EXECUTION	VT*	VT*	DT	DT
SYSTEM TEST REPORT	VT/DT*	VT/DT*	DT	DT
INTEGRATION TEST PROCEDURE	VT/DT*	VT/DT*	DT	DT

TASK	PROTECTION	IMPORTANT TO SAFETY	IMPORTANT TO AVAILABILITY	GENERAL
INTEGRATION TEST EXECUTION	VT*	VT*	DT	DT
INTEGRATION TEST REPORT	VT/DT*	VT/DT*	DT	DT
USER DOCUMENTATION	DT/VT	DT/VT	DT	DT
SVVR	VT	VT	N/A	N/A
SAT PROCEDURE	DT/VT	DT/VT	DT	DT
SAT REPORT	DT/VT	DT/VT	DT	DT
KEY: ORIGINATOR/REVIEWER (E.G. DT/VT)	DT = DESIGN TEAM			VT = V&V TEAM

* The CEO may choose to have the DT write and execute the procedures and the VT review them and the test reports. See pages 17, 22, 41, 86, and 89.

Any single software item may be handled, at the discretion of the CEO, according to a class that is more important to safety. For example, a particular important to safety software item may be module tested by the V&V team without redesignating the software item as protection class.

This page intentionally left blank.

EXHIBIT 5-2

CHECKLIST NO. 1

SOFTWARE VERIFICATION AND VALIDATION REQUIREMENTS PHASE CHECKLIST

Notes on the use of this Checklist: Separate copies of this checklist may be completed by different individuals at different times for the various life cycle phases. Completed sections must be filled in their entirety. Unused sections may be left blank. Items identified with an asterisk (*) require that the reviewer attach his notes or describe on the reverse side of the checklist the method used or activities performed by the reviewer to verify the checklist item.

Software Item Name: _____ Software Item ID:

		YES
1.	Does the review of available documentation identify:	
a.	Completeness and correctness in specifying the performance requirements and operational capabilities and concepts of the system. Determine if the right problem is being solved and if the requirements are consistent with emergency procedures, etc.	____
b.	Completeness and correctness in system definition and interfaces with other equipment.	____
c.	Unambiguous, correct, and consistent description of the interfaces and performance characteristics of each major function.	____
d.	Establishment of a reasonable and achievable set of test requirements. These requirements should be related to performance goals and also define acceptability criteria.	____
e.	Definition of physical characteristics, reliability and maintainability objectives, operating environment, transportability constraints, and design standards.	____
f.	Definition of the necessary training requirements and considerations.	____
g.	Treatment of man/machine interface requirements.	____
h.	Definition of integration requirements.	____
i.	Definition of installation, operation, and maintenance requirements.	____

j. Are the bases for the requirements identified?

k. Review requirements with respect to the following possible errors:

1. Inadequate or partially missing performance criteria.
2. Inadequate or partially missing operating rules (or information).
3. Inadequate or partially missing ambient environment information.
4. Requirements that are incompatible with other requirements.
5. Inadequate or partially missing system mission information.
6. Ambiguous or requirements subject to misinterpretation.
7. User's needs not properly understood or reflected.
8. Requirement not traceable to user's needs.
9. Requirements which cannot be physically tested.
10. Accuracy specified does not conform to the need.
11. Data environment inadequately described.
12. Input/output data parameters units incorrect.
13. Erroneous external interface definition.
14. Initialization of the system not properly considered.
15. Vague requirements of the functions to be performed.
16. Required processing inaccurate.
17. Required processing inefficient.
18. Required processing not necessary.
19. Missing requirements on flexibility, maintainability.
20. Missing or incomplete requirements of response to abnormal data or events.
21. Inadequate or incorrect algorithm.
22. Incorrect timing/synchronization requirements.
23. Incorrect hardware interface requirements.
24. Incorrect allocation of system resources.

2. Do the hardware requirements identify: YES

a. All input/output and requirements, including range, accuracies and data rates. ____

b. Design features (e.g., keylocks) which provide administrative control of all devices capable of changing the content of the stored programs or data. ____

c. Initialization requirements, such as power-up and power-down. ____

d. Design features for the detection of system failures (e.g., on-line self-tests). ____

e. Manually-initiated in-service test or diagnostic capabilities. ____

f. Human factors engineering design features which ease the interaction with the system for operation, maintenance, and testing. ____

g. Margins for timing, memory/buffer size, etc., including minimum margins for design. ____

h. Interrupt features. ____

3. Do the software requirements identify: YES

a. Process inputs including voltage and sampling frequency. ____

b. System software, utility routines and other auxiliary programs required for operation ____

c. Algorithms to be programmed with consideration to handling of abnormal events ____

d. Data files and data required for the algorithms, including symbolic names and requirements for flexibility. ____

e. Process outputs, including ranges, accuracies, update interval, and human factors considerations of the operator interface. ____

f. Initialization requirements, such as initial values and start-up sequence. ____

g. Parameters to configure system program logic for response to detected failures. ____

h. Operator interface requirements (switches, readouts). ____

i. In-service test or diagnostic capabilities. ____

j. Timing requirements for all time-dependent events, including overall system requirements. ____

k. Limitations on processor time and memory capabilities. ____

l. Security requirements (e.g., passwords). ____

m. After *critical analysis* all software requirements that have safety implications have been identified. Each requirement in the SRS has been evaluated against the various system hazard analyses to assess its potential for unacceptable risk. Refer to IEEE Std. 1228-1994 Annex for guidance. ____

Reviewer's comments(Optional): _____

_____

Reviewed by: Name _____ Signature _____ Date:_____

EXHIBIT 5-3
CHECKLIST NO. 2

SOFTWARE VERIFICATION AND VALIDATION
DESIGN PHASE CHECKLIST

Notes on the use of this Checklist: Separate copies of this checklist may be completed by different individuals at different times for the various life cycle phases. Completed sections must be filled in their entirety. Unused sections may be left blank. Items identified with an asterisk (*) require that the reviewer attach his notes or describe on the reverse side of the checklist the method used or activities performed by the reviewer to verify the checklist item.

Software Item Name: _____ Software Item ID: _____

1. Does the available documentation adequately address:	YES
a. Architecture, for both hardware and software.	____
b. Input/output interface.	____
c. System and Executive Control.	____
d. Operating Sequences - initialization, start-up, error detection, restart, etc.	____
e. Testability - use of test equipment, such as data tapes, simulations, etc.	____
f. Timing analysis - sampling rates, response time, etc.	____
g. Availability - what does analysis and data indicate?	____
h. Algorithm design and data verification.	____
i. Information flow - communication between subsystems, data management and signal conversion to engineering units.	____
j. Human factors engineering.	____
k. Is the design correct, complete, and traceable to requirements?	____
l. Is the design internally consistent?	____

m Is the design feasible? ____

n. Is the design clear and unambiguous? ____

o. Is the design testable? ____

p. A Software Safety Design Analysis was performed verifying that the safety-critical portion of the software design correctly implements the safety-critical requirements (after *Critical Analysis* in requirements phase) and introduces no new hazards. Refer to IEEE Std. 1228-1994 Annex for guidance. ____

Reviewer's comments(Optional): _____

_____

Reviewed by: Name _____ Signature _____ Date_____

EXHIBIT 5-4
CHECKLIST NO. 3

SOFTWARE VERIFICATION AND VALIDATION
IMPLEMENTATION PHASE CHECKLIST

Notes on the use of this Checklist: Separate copies of this checklist may be completed by different individuals at different times for the various life cycle phases. Completed sections must be filled in their entirety. Unused sections may be left blank. Items identified with an asterisk (*) require that the reviewer attach his notes or describe on the reverse side of the checklist the method used or activities performed by the reviewer to verify the checklist item.

Software Item Name: _____ Software Item ID: _____

		YES
1.	Review the source code with respect to the following:	
	a. Does the source code conform to specified standards and procedures?	____
	b. Are the comment statements sufficient to give an adequate description of each routine?	____
	c. Is the source code clearly understandable?	____
	d. Is the source code logically consistent with design specs?	____
	e. Are all variables properly specified and used?	____
	f. Is there satisfactory error checking?	____
	g. Do all subroutine calls transfer variables correctly?	____
	h. Is the data read in each file consistent with the data written to it?	____
	i. A Software Safety Code Analysis was performed that analyzes the safety-critical portions of the design for correct implementation including: Logic, Data, Interface, Constraint, Programming Style, Non-critical code, Timing/Sizing analyses. Refer to IEEE Std. 1228-1994 Annex for guidance.	____

2.	Do the database modules adequately and correctly reflect:	YES
	a. Program and general content.	___
	b. File organization, layout, and residence.	___
	c. File accessing methods.	___
	d. File size.	___
	e. Data record description(s) – record layout, field allocations, field names, detailed description of field contents.	___
	f. Initialization requirements.	___
	g. Maintenance.	___
3.	Review Module Test Reports	YES
	a. Module test reports (and unit test reports if applicable) indicate correct execution of critical software elements.	___

4. Do procedures exist (as necessary) to: YES

   a. Generate all object code required for system generation and produce the corresponding software listings. ____

   b. Generate a customized database and system parameter file according to plant-specific requirements and produce the corresponding listings. ____

   c. Configure the operating system according to the plant-specific hardware configuration. ____

   d. Generate the system from the above results. ____

   e. Initialize and boot the system after system generation. ____

   f. Modify, enhance, and maintain the system including the usage of diagnostic and debugging utilities. ____

   g. Generate and update displays. ____

Reviewer's comments(Optional): _____

_____

Reviewed by: Name _____ Signature _____ Date_____

EXHIBIT 5-5

CHECKLIST NO. 4

SOFTWARE VERIFICATION AND VALIDATION TEST PHASE CHECKLIST

Notes on the use of this Checklist: Separate copies of this checklist may be completed by different individuals at different times for the various life cycle phases. Completed sections must be filled in their entirety. Unused sections may be left blank. Items identified with an asterisk (*) require that the reviewer attach his notes or describe on the reverse side of the checklist the method used or activities performed by the reviewer to verify the checklist item.

Software Item Name: _____ Software Item ID: _____

1.	Verify program integration with hardware in accordance with the following:	YES or NO
	a. Does the integrated program conform with the maximum resource requirements for memory size and program execution time?	____
	b. Does the integrated program interface properly with external files?	____
	c. Have all of the elements of the integrated program been identified in the module list?	____
	d. Does the code compile and link without errors?	____
	e. Are interfaces between programs, data files, and libraries correctly programmed?	____
2.	Verify program validation in accordance with the following:	YES or NO
	a. Has each section of the test procedure been completed accurately?	____
	b. Have all tests passed and have all requirements of testing been fulfilled?	____
3.	Verify test results and report in accordance with the following:	YES or NO
	a. Does the Test Report comply with the format specified in the Test Plan?	____
	- Does it provide complete identification of the program tested?	____
	- Does it specify the scope of the Test Report?	____

- Does it reference the Test Plan and any other relevant documents? ____

- Does it include a complete and accurate description of the test environment: ____

  Hardware configuration? ____

  Support software used? ____

- Does it describe and justify each deviation from the Test Plan? ____

- Does it provide a summary of test results? ____

- Does it include an evaluation of the program performance with respect to requirements? ____

- Does it provide recommendations for retesting, or program acceptance, or both? ____

- Does it provide a detailed description of the results of each test case? ____

- Does it include a copy of the test case log? ____

- Does it include all discrepancy reports prepared during the testing? ____

b. Is the information in the Test Report an accurate statement of the testing performed? ____

- Is output summary of test results accurately reflect the test output produced? ____

- Is the evaluation of the program a realistic and accurate reflection of the test results? ____

- Are the recommendations regarding retesting and acceptance sound and based on the test results? ____

- Do the descriptions of the test case results accurately reflect actual test outputs? ____

- Is the test case log complete and consistent with actual test output? ____

- Are the discrepancy reports complete and consistent with actual test output? ____

c. Have all test cases been executed correctly? ____

- Does the test case log indicate performance of each test case in the specified test environment using specified test procedures? ____
- Is there an explanation for any deviation from the specified test environment or procedures? ____
- Is there a test exception report for each deviation from expected results? ____
- Were correct input data used for each test case? ____
- Is the output produced by each test case accurately reported? ____

Reviewer's comments(Optional): _____

_____

Reviewed by: Name _____ Signature _____ Date_____

EXHIBIT 5-6

CHECKLIST NO. 5

SOFTWARE VERIFICATION AND VALIDATION INSTALLATION AND CHECKOUT PHASE CHECKLIST

Notes on the use of this Checklist: Separate copies of this checklist may be completed by different individuals at different times for the various life cycle phases. Completed sections must be filled in their entirety. Unused sections may be left blank. Items identified with an asterisk (*) require that the reviewer attach his notes or describe on the reverse side of the checklist the method used or activities performed by the reviewer to verify the checklist item.

Software Item Name: _____ Software Item ID:_____

		YES
1.	Is the user documentation installation package sufficient to install the software on the delivered hardware?	____
2.	Is the user documentation clear, unambiguous, and consistent with system requirements?	____
3.	Does the V&V report have positive findings?	____
4.	Have all discrepancies and V&V findings been resolved to the satisfaction of the V&V team?	____
5.	Are SCM controls in place for the user to report errors?	____
6.	A Software Safety Change Analysis was performed for all changes made to the software. Refer to IEEE Std 1228-1994 Annex for guidance.	____
7.	Training documentation meet Safety Training Requirements	____

Reviewer's comments (Optional): _____

_____

Reviewed by: Name _____ Signature _____ Date:_____

EXHIBIT 6-1
SOFTWARE CHANGE REQUEST

SOFTWARE CHANGE REQUEST FORM SCR # ____

Date: _____ CUSTOMER : _____ Page 1 of ___

Subject: _____ Software Affected: _____ Originator: _____ Version: _____ Revision: _____ Classification[ ]: 1-Emergency 2-Corrective 3-Adaptive 4-Perfective
Summary of Requested Change:
Reason for Change:
Documents Affected (Document No./Revision):
Design Approval/Date: _____ _____ Cognizant Engineering Organization Manager/Date _____ Other Approval/Date

Implementation Completed: (Including Documentation) _____ Implementation Engineer/Date	Testing Completed: TER #: _____ Documentation: _____ Review/Date: _____

Exhibit 6-2

CONTROLLED SOFTWARE DISTRIBUTION LIST

Code Name: _____
Version/Revision Number: _____
Comments:

_____

DISTRIBUTION

User Name	Location	Date

Comments:

Cognizant Engineering Organization Manager: _____
Date: _____

Exhibit 6-3

SOFTWARE PROBLEM REPORT

Code Name: _____

Version /Revision Number: _____

(Print or type)
User Name: _____

User Company/Organization: _____

Telephone Number: _____

Description of Problem (Attach additional pages as necessary):

Comments:

User Signature: _____ Date _____

Please Return to: _____
Cognizant Engineering Organization Manager

Westinghouse Electric Company LLC
2000 Day Hill Rd.
Windsor, CT 06095

EXHIBIT 7-1

COMPUTER CODE ERROR NOTIFICATION

Code Name: _____

Version/Revision Number: _____ **Page 1 of 2**

Associated Documentation Name:	Documentation Number:

Error Description:

Possible Impact of Error:

Recommendation to User (Error Avoidance and/or Remedial Action):

Prepared by: _____ Date: _____

Cognizant Engineering Organization Manager: _____ Date: _____

EXHIBIT 7-2

COMPUTER CODE ERROR NOTIFICATION RESPONSE RECEIPT

Page 2 of 2

Code Name: _____

Version/Revision Number: _____

I acknowledge receipt of the attached error notice, and have assessed the impact as indicated. When the error impact is yes appropriate quality actions as necessary, including 10CFR21 considerations will be taken and documented.

User Name (Print or type)	Signature	Date	Error Impact No	Error Impact Yes

Users are to return this signed form within 60 days to:
Cognizant Engineering Organization Manager: _____

Westinghouse Electric Company LLC
2000 Day Hill Rd.
Windsor, CT 06095

EXHIBIT 8-1
COMPUTER CODE CERTIFICATE

Page 1 of ____

The following computer code, as noted by its name, version number and executable file identification, is approved for design use.

Code Name:_____

Version/Revision Number:_____

Executable File Identification:_____

Computer(s):_____

Restrictions (List any limitations on use, special hardware considerations, etc.):

_____

_____

_____

Listed are the software modules and their current revision (use additional pages as necessary):

Module Name/Classification	Version/Revision

Verification and Validation Report Number:_____

Cognizant Engineering Organization Manager:_____Date:_____

EXHIBIT 9-1

Test Exception Report

TER Number _____

System Name:	**Plant:**
Procedure Name:	**Procedure Number:**
Tester Name:	**Rev.:**
Summary of Exception:	Date:
Class:	Step:
Resolution: Responsibility:	
Implementation: ☐ Procedure Correction ☐ Software Change	
Implemented By:	Date:
Retested By:	Date:
Reviewed By:	Date:

EXHIBIT 9-2
CORRECTIVE ACTION PROCESS
Issue Report

Step 1: Identification of Issue - (required information)

Step 1: Identification of Issue	
Created by:	Alan W Hill
Created on:	08/14/2002
Phone:	8-286-8592
Originator's:	Organization
Originator's Business Unit:	NABU
Organizational Level 2:	Projects
Organizational Level 3:	Project Controls and Contract Administration
Originator's Group:	Projects - Controls & Contract Admin
Originator's Location:	Monroeville PA
Issue Title:	Example Title
Issue Description:	Exmaple desciption of the issue is entered here.
QA/QC Controlled?	○ Yes ◉ No By choosing yes, this issue will become a CAR
Issue raised by:	◉ Employee ○ Supplier ○ Customer ○ Regulatory Agency

Step 2: Identification of Issue - (optional information)

Step 2: Identification of Issue - (optional information)	
Suggested Significance Level:	○ High ○ Suggestion for Improvement ◉ Medium ○ Best Practice ○ Watch/Trend
Recommended Resolution:	
Background:	
Date Issue First Occured:	
Affected Customer(s):	Utility X
Affected Unit(s):	Units A & B
Affected Supplier(s):	
Immediate Actions Taken:	
Impacted Documents, Systems, Components, etc:	
Submitted on behalf of:	
Is this issue potentially reportable?	○ Yes ○ No ○ I don't know
Improvement Plan #:	
Audit #:	
Disposition of Material:	○ Accept as is ○ Repair ○ Rework ○ Scrap
Have non-conforming material or calculations been marked or "Hold Tagged"?	○ Yes ○ No ○ N/A

Step 3: Comments

Step 4: Prepare for Issue Review Meeting

Step 5: Issue Review Meeting Complete

Step 6: Event Codes

Step 7: Causal Analysis

Step 8: Root Cause Coding

Step 9: Summary Table of Commitments

Step 10: Effectiveness Evaluation

Step 11: WorkFlow Information

CAM Administration Section