GE Energy
Nuclear

NEDO-33226
Class I
eDRF# 0000-0051-8268
March 2006

LICENSING TOPICAL REPORT

ESBWR I&C
SOFTWARE MANAGEMENT PLAN

INFORMATION NOTICE

This document NEDO-33226, Rev. 0 contains no proprietary information.

IMPORTANT NOTICE REGARDING CONTENTS OF THIS REPORT PLEASE READ CAREFULLY

The information contained in this document is furnished as reference to the NRC Staff for the purpose of obtaining NRC approval of the ESBWR Certification and implementation. The only undertakings of General Electric Company with respect to information in this document are contained in contracts between General Electric Company and participating utilities, and nothing contained in this document shall be construed as changing those contracts. The use of this information by anyone other than that for which it is intended is not authorized; and with respect to any unauthorized use, General Electric Company makes no representation or warranty, and assumes no liability as to the completeness, accuracy, or usefulness of the information contained in this document.

Table of Contents

Appendices

List of Tables

List of Figures

1 Introduction

This Software Management Plan (SMP) provides the technical and administrative direction necessary to implement the design activities as outlined in the supporting documentation listed in Section 4.1.1 of this plan. This document also establishes the design and quality standards for the software-based products produced for the ESBWR Instrumentation and Control Systems (I&C[1]).

This plan includes software-based products, all safety and nonsafety-related instrumentation and control systems, which perform the monitoring, control, and protection functions associated with all modes of plant normal operation as well as off-normal, emergency, and accident conditions. Normal conditions include startup, shutdown, standby, power operation, and refueling. The Overall Requirements section (Section 3) of the MMIS and HFE Design Implementation Plan [4.1.1(2)][2] defines the off-normal, emergency, and accident conditions. The software-based products addressed by the plan are divided into two discrete groups:

1. Traditional Computer-based Systems, (e.g., a core monitoring computer system): These systems use data provided by other systems to perform computations and usually do not operate in real time. This plan will refer to such systems as *software-based systems*.

2. Firmware-based Systems that contain microprocessor(s) to control the internal functions (such as microprocessor-based instrumentation): These systems are designed to acquire data from plant processes, control or monitor the processes, and operate in real time. This plan will refer to such systems as *microprocessor-based systems*.

This plan does not provide requirements or guidance for development or use of any Engineering and Analytical software.

1.1 Purpose and Scope

This plan (SMP) outlines the approach to be followed during the software engineering process (planning, requirements, design, implementation, integration, validation, installation, and operations & maintenance) for Quality Class Q software-based products. This plan or a commercially accepted software engineering process shall be used for Quality Class N software-based products.

---

[1] The functions are the same as for MMIS as defined in the Overall Requirements section (Section 3) of the MMIS and HFE Design Implementation Plan.

[2] Section numbers referenced in this manner refer to the codes and standards documents listed in the Applicable Documents section (Section 4) of this SMP.

The purpose of this plan is to:

1. Establish the standards, methodologies, tools, and quality assurance procedures to be used during the software design and development process,

2. Define the design activities performed at each level of the software development process and how those activities are verified,

3. Define how verified requirements are passed on to lower levels of the engineering process, and

4. Specify how the safety requirements are to be documented during the design process to minimize unknown, unreliable, and abnormal conditions.

This plan is designed to remain in effect throughout the entire life cycle of the I&C software-based product. Exceptions from this plan may be accepted if the product quality level is maintained. Any deviation from this plan must be justified and approved by the Project Management Team (PMT). The justification shall be documented in the software project Design Record File (DRF) {see GEEN EOP 42-10.00 [4.1.2(3c)]}. The Software Configuration Management Plan [4.1.2(1)] is in force during all phases of the software life cycle, and includes control of software-based product documentation, source, object and executable codes, program data, development tools, environment (hardware and software), and test cases.

1.2 Software Developed by Vendors

All software developed by the vendors for use in an I&C software-based product shall conform to the

1. Requirements outlined in this plan, or an approved equivalent, and

2. Technical and quality assurance requirements as:

   a. Defined in the procurement package produced in accordance with the Project Procurement Manual {PPM [4.1.1(5)]} for vendors external to GEEN, or

   b. Outlined in the GEEN EOP 45-2.00, Procurement of Engineering Services [4.1.2(3f)] for vendors internal to GEEN (i.e. GEEN organizational units outside the project, such as GEEN Nuclear Fuel (GNF)).

The Responsible Technical Project Engineer (RTPE) (or a designated appointee) shall be responsible for the review and approval of all proposed alternate plans and documentation. The qualification of the responsible reviewer or verifier is defined in the SVVP [4.1.2(2)]. A summary of the review and a justification for the plan substitution shall be included in the software project DRF.

2 Organization and Responsibilities

2.1 I&C Organizational Units

The I&C organization is part of the Engineering organization which includes other engineering disciplines and project configuration management, and other support units. The Engineering group interfaces with GEEN's Quality organization. The I&C organization also interfaces with the related "Project Quality Unit(s)" under the GEEN Quality organization. The following units form the engineering organization:

- The I&C/Electrical Systems,
- Software Project Engineering (SPE), and
- Configuration Management.

Specific responsibilities of the key personnel are described in Section 2.3, Organization Responsibilities. Interfaces with subcontractors and suppliers are described in the specific responsibilities of Section 2.3.

2.1.1 I&C/Electrical Systems Organization

The I&C/Electrical Systems Organization typically includes the I&C System Engineering unit and I&C Application unit, as well as several Electrical Units, which are outside the scope of this document. The task lead of each unit reports to the manager of the I&C Electrical Systems Organization.

I&C Systems Engineering Unit: responsible for I&C system design and requirements;

I&C Application Unit: responsible for detailed software and hardware design and implementation, and communication interface design.

2.1.2 Software Project Engineering Organization

The SPE is responsible for development of software life cycle process plans and software QA program. It has interfaces with I&C Application and I&C Systems Engineering Units. Software Quality personnel do not report to the task lead of the I&C Application Unit. The SPE is also responsible for software safety, and the supervision of the implementation of software development according to software design process plans. It also has an interface with the Project Quality Unit under the GEEN Quality Organization and is responsible for providing and managing software design verification and validation personnel that are needed to be independent from the software designers of the I&C Application Unit personnel. Finally, the SPE is responsible for HFE/HMI & Data Communications and the simulator activities. The following units form the SPE organization:

Software Quality Assurance Unit: responsible for developing the audit plans as part of GEEN QA organization but physically from within Engineering.

Baseline Review Team: responsible for process oversight and confirming configurable items are placed under proper control.

Verification & Validation Team: responsible for the independent verification and validation of Quality Class Q software as specified by the SVVP.

Software Safety Team: responsible for the independent performance of Quality Class Q safety analysis.

HFE/HMI & Data Communications Unit: responsible for the HFE design, plant data communication system design, and specific control software design.

Simulation Platform and SAET Unit: responsible for the nonsafety-related requirement specification and design and implementation of simulator software activities.

2.1.3 Configuration Management Organization

Configuration Management Unit: responsible for project level configuration management procedures.

2.2 I&C Organizational Resources

2.2.1 Budget

The financial resources necessary to carry out the work in the SMP is proprietary and maintained by senior manager(s) responsible for the personnel assigned to their area of responsibility.

2.2.2 Personnel

The SMP [this plan] summarizes the organizational responsibilities for each unit and key members of the I&C organization.

The design team shall be composed of experienced individuals whose collective expertise covers a broad range of disciplines relevant to the design and implementation process. These disciplines shall include technical project management, systems engineering, nuclear engineering, electrical engineering, and instrumentation and controls engineering.

Staffing plans shall be established for each work package according to EOP 25-5.00 [4.1.2(3g)]. Due to the very sensitive nature of the information, these plans will be retained as GE Company Private information in Project Files for the duration of the project.

The project-training matrix defines the training for project personnel and the GE Learning Center documents and identifies the methods for obtaining, training, and retraining staff.

2.2.3 Methods/Tools

The generic methods, tools, standards, and procedures to be used for each engineering process step are defined in these software plans, as well as in the GEEN procedures and processes referenced by these plans. However, GEEN staff will evaluate the plans, procedures, and processes used by vendors outside GEEN against this GEEN guidance, to determine if the GEEN staff can accept the vendor process as is, or if enhancing the vendor's process is required. The

GEEN staff will document and maintain these evaluations in a vendor-specific DRF. The GEEN staff will evaluate and document any system-specific processes, methods, or resources that differ from these software plans in a DRF.

The Software Conventions and Guidelines [4.1.2(7)] document will contain the required conventions and guidelines for software programming. Individual systems or groups of systems may require language-specific Software Conventions and Guidelines, based on the software programming languages used and the structure and syntax unique to individual languages. Software based products, development methodology(s), and programming language(s) are defined separately for each system, group of systems, or system vendor, as appropriate. Optimal choice of such techniques is made after definition of the hardware and software content of each system is specified, which allows advances in technology to be incorporated.

2.2.4 Standards

The primary international, national, industry, and company standards and guidelines in the SMP Supporting and Supplemental Documents section are the basis for this plan.

2.3 Organizational Responsibilities

The primary duties of the organization covered by the SMP and of individuals within the organization are defined below. The Manager/Technical Task Leads for the Control/Electrical Systems and Configuration Management are defined in the SDP [4.1.2(6)] under Project Responsibilities. Figure 4 maps the organization positions to the function positions and duties described in this section.

Baseline Review Team	-	The BRT is responsible for judging adherence to the process for the work products being baselined. The members of this team are appointed by the BRT Task Lead. For safety-related software, the team members must not work for the same manager as the person who generated the work to be reviewed. {see SCMP [4.1.2(1)]}.
Baseline Review Team Task Lead	-	The person responsible for organizing the baseline review process. This person is appointed by the SPE Task Lead {see SCMP [4.1.2(1)]}.

Engineering Manager	-	The person who is responsible for directing the engineering activities of the ESBWR Project and is the final authority for software based product release. This person is also responsible for communication with the customer throughout the software engineering process on all matters requiring customer review and/or approval.
Responsible Configuration Control Engineer	-	The person assigned responsibility for the configuration management of the I&C software-based products {see SCMP [4.1.2(1)]}.
Responsible Engineer	-	The person responsible for a given technical item (e.g., the design and development of the documentation).
Responsible Engineering Technical Lead	-	A person with the overall responsibility for a set of I&C software-based products. Each I&C software-based product is assigned a RETL, including those products or systems developed by vendors. The RETL interfaces and supervises vendor's product.
Responsible Software Safety Engineer	-	The person with responsibility for ensuring the safety qualities of all the software being developed for I&C, through the integration of the software with the final hardware platform {see SSP [4.1.2(4)]}.
Responsible Technical Project Engineer	-	The person with overall technical responsibility for ensuring that the hardware and software design of a software-based product meets the specified requirements.
Responsible Verifier(s)	-	The Responsible Verifier(s) is an individual who meets the independence as described in GEEN EOP 42-6.00 [4.1.2(3b)] for verifications or EOP 42-6.10 for deferred verifications of design process and the accompanying documentation.
Task Lead	-	A person with the overall responsibility for the processes and procedures of a given I&C unit or group of units.

3 Definitions, Acronyms and Abbreviations

Acronyms and Abbreviations:

BRT	Baseline Review Team
CI	Configuration Item
DCIS	Distributed Control & Information Systems
DRF	Design Record File
ECN	Engineering Change Notice
EMC	Electromagnetic Compatibility
EOP	Engineering Operating Procedure
FX	Logic Function (See Logic Diagram Symbol Index)
GE	General Electric Company
GEEN	GE Energy Nuclear (Previously GENE)
HFE	Human Factors Engineering
HSI	Human System Interface
HSS	Hardware/Software Specification
I/A	Intelligent Automation
I/O or IO	Input/Output
IP	Installation Plan
IPS	Instrument Performance Specification
LD	Logic Diagram
MCR	Main Control Room
MMIS	Man-Machine Interface System
N/A	Not Applicable
O&MP	Operation & Maintenance Plan (Change Control)
P&ID	Piping and Instrumentation Diagram

PDM	Project Design Manual
PDS	Previously Developed Software
PMM	Project Management Manual
PMT	Project Management Team
PP	Project Plan
PPM	Project Procurement Manual
RCCE	Responsible Configuration Control Engineer
RE	Responsible Engineer
Reg Guide	Regulatory Guide
RETL	Responsible Engineering Technical Lead
RSSE	Responsible Software Safety Engineer
RTPE	Responsible Technical Project Engineer
SBD	System Block Diagram
SCMP	Software Configuration Management Plan
SDD	System Design Description
SDS	Software Design Specification
SIntP	Software Integration Plan
SIP	Software Installation Plan
SMP	Software Management Plan
SOMP	Software Operation and Maintenance Plan
SQAP	Software Quality Assurance Plan
SRP	Standard Review Plan
SRS	Software Requirements Specification

SSA	Software Safety Analysis
SSP	Software Safety Plan
SVVP	Software Verification and Validation Plan
US NRC	United States Nuclear Regulatory Commission
V&V	Verification and Validation

The following definitions apply throughout this document:

Algorithm	- A finite set of well-defined rules for the solution of a problem in a finite number of steps.
Baseline	- Items that have been formally reviewed and agreed upon, that thereafter serve as the basis for further development, and that can be changed only through formal change control procedures.
Baseline Review and Oversight	- A formal review, conducted at the end of each software life cycle phase, and requested by the Responsible Technical Project Engineer (RTPE). The baseline review process is under the control of the Baseline Review Team. The Baseline Review Team (appointed by the Baseline Review Team Task Lead) performs the review. These reviews confirm adherence to the SMP (this plan), SCMP [4.1.2(1)] and SVVP [4.1.2(2)]. All Baseline Reviews are performed and documented in accordance with the Software Configuration Management Plan (SCMP). Oversight including performance indicators is also performed during each software life cycle phase. The RTPE is informed shortly after findings and observations are confirmed. The BRT Task Lead issues an Oversight Report to the RTPE and all direct and indirect project management having a need to know shortly after the end of each software life cycle phase.

Black-box Testing	-	A system/software test methodology that is derived from external specifications and requirements of the system. Methods for black box testing include random testing and testing at boundary values. Black box testing does not evaluate operation inside the module. Instead, it verifies the inputs, outputs, and interfaces to the software being tested, but does not check the implementation techniques, nor does it assume that all statements in the program are executed.
Code Review (Code Analysis)	-	Software source code presented to project personnel for comment or approval.
Configuration Item	-	An aggregation of hardware, software, or both, that is designated for configuration management and treated as a single entity in the configuration management process.
Design Record File	-	A formal controlled information record under the GEEN procedures for in-progress and completed engineering work which is retained and from which work can be retrieved.
Design Reviews	-	Formal, design adequacy evaluations which are performed by knowledgeable persons other than those directly responsible and accountable for the design in accordance with GEEN Engineering Operating Procedure (EOP) 40-7.00 [4.1.2(3a)]. Design reviews verify that product designs meet functional, contractual, safety, regulatory, industry codes and standards, and company requirements.
Embedded System	-	A specialized computer system that is part of a larger system or machine. Typically, an embedded system is housed on a single microprocessor board with the programs stored in ROM (Firmware).
Firmware	-	Software contained in a flash or other programmable memory, designed as permanent storage, not easily upgraded. The combination of a hardware device and computer instructions and data that reside as read-only software on that device. See Software.
Firmware-based System	-	See microprocessor-based systems.

Microprocessor-based System	-	These contain microprocessor(s) or microcontroller(s) to control the internal functions (such as microprocessor-based instrumentation). These systems are design to acquire data from plant processes, control or monitor the processes, and operate in real time.
Independence	-	For most safety-related work, 10 CFR 50 Appendix B defines Independence as being someone who did not perform, or direct the performance or methods, used to implement the artifact being reviewed. For software, the NRC views this as insufficient, and requires the reviewer to work for a manager (defined as someone having fiduciary duties, and inviolate budget, schedule, and resources for the review activities) different than the manager of the person implementing the artifact to be reviewed.
Instrument	-	A hardware device used for analytical or control functions and usually containing an embedded microprocessor(s).
Quality Class N	-	Project nonsafety-related classification {see Project Quality Assurance Plan [4.1.1(4)]}.
Quality Class Q	-	Project safety-related classification {see Project Quality Assurance Plan [4.1.1(4)]}.
Software Life Cycle	-	The period of time that begins when a software product is conceived and ends when the software is no longer available for use (see Section 5). This plan does not include the Acquisition or Retirement planning phases.
Software	-	Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.
Software Module	-	The smallest segment of code (also called routine, procedure, function or subprogram).
Software Package	-	A collection of software modules (e.g., subroutines, main control tasks) brought together to form a single software product.

Software Source Code	-	Computer instructions and data definitions expressed in a form suitable for input to an assembler, compiler, or other translator. Source code includes graphical representations, such as function blocks, as well as the more traditional procedural languages, such as "C."
Software Unit	-	See Software Module.
Software-based System	-	A computer system composed of specified programs that typically run on commercially available hardware using commercially available operating systems. These systems do not directly control or interface with plant systems, and do not run in real time.
Traceability Matrix	-	This matrix or matrices trace requirements and their implementation between two or more phases of the software process. The matrices document implementation of requirements from design documentation into detailed design, code, and test documents.
Validation	-	The testing process that ensures the software-based product meets its intended use and is compliant with system functional, performance and interface requirements.
Verification	-	See Verification and Validation
Verification and Validation	-	The V&V activities performed in accordance with GEEN EOPs 40-7.00 (Design Reviews) [4.1.2(3a)] or 42-6.00 (Independent Design Verification) [4.1.2(3b)] or equivalent to ensure the quality of the design process and the associated documents produced.
White-box Testing	-	This is a software test methodology at the software module level. Test cases exercise the internal structure of the program. These tests insure that all the statements and branches are exercised to check the system implementation. Some methods of white-box testing are statement coverage, branch coverage and path coverage.

4 Applicable Documents

4.1 Supporting and Supplemental Documents

4.1.1 Supporting Documents

The following supporting documents were used as the controlling documents in the production of this plan. These documents form the design basis traceability for the requirements outlined in this plan.

Document Title	Document Number
1. Project Design Manual (PDM)	
2. Man-Machine Interface System and HFE Design Implementation Plan (MMIS/HFE IP)	NEDO-33217
3. {Left blank}	
4. NP 2010 COL Demonstration Project Quality Assurance Program	NEDO-33181
5. Project Procurement Manual	
6. Project Management Manual	NEDO-33216
7. Composite Specification (26A6007)	A11-5299

4.1.2 Supplemental Documents

The following supplemental documents are used in conjunction with this document.

Document Title	Document Number
1. Software Configuration Management Plan (SCMP)	NEDO-33227
2. Software Verification and Validation Plan (SVVP)	NEDO-33228
3. GEEN Engineering Operation Procedures	NEDE-21109
a. 40-7.00 Design Review	
b. 42-6.00 Independent Design Verification	
c. 42-10.00 Design Record File	

Document Title	Document Number
d. 55-2.00 Engineering Change Control	
e. 55-3.00 Field Deviation Disposition Request	
f. 45-2.00 Procurement of Engineering Services	
g. 25-5.00 Work Planning and Scheduling	
4. Software Safety Plan (SSP)	NEDO-33230
5. Software Quality Assurance Plan (SQAP)	NEDO-33245
6. Software Development Plan (SDP)	NEDO-33229
7. Software Conventions and Guidelines	
8. Software Installation Plan (SIP)	NEDO-33247
9. Software Integration Plan (SIntP)	NEDO-33246
10. Software Operation & Maintenance Plan (SOMP)	
11. Software Training Plan (STrngP)	

4.2 Codes and Standards

The following codes and standards are applicable to the ESBWR I&C Software Management Plan to the extent specified herein. The applicable date/revision of the code or standard is specified in the Composite Specification [4.1.1(8)].

4.2.1 American Society of Mechanical Engineers (ASME) Codes

1. ASME NQA-1

4.2.2 Institute of Electrical and Electronic Engineers (IEEE) Standards

1. IEEE 1058.1, Software Project Management Plans
2. IEEE 1228, Software Safety Plans
3. IEEE 829, Standard for Software Test Documentation
4. IEEE 7-4.3.2, Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations.

4.2.3 International Standards

1. ISO-9000-3, Quality management and quality assurance standards - Guidelines for the application of ISO 9001 to the development, supply and maintenance of software

4.2.4 U.S. Nuclear Regulatory Commission (NRC) Regulatory Guides (Reg Guide)

1. Reg. Guide 1.152, Criteria for Digital Computers in Safety Systems of Nuclear Power Plants

2. Reg. Guide 1.172, Software Requirements Specifications for Digital Computer Software Used in Safety Systems of Nuclear Power Plants

3. Reg. Guide 1.170, Software Test Documentation for Digital Computer Software used in Safety Systems of Nuclear Power Plants

4. Reg. Guide 1.171, Software Unit Testing for Digital Computer Software used in Safety Systems of Nuclear Power Plants

5. Reg. Guide 1.168, Verification, Validation, Reviews, and Audits for Digital Computer Software Used in Safety Systems of Nuclear Power Plants

4.2.5 NUREG

1. NUREG/CR-6101-1993, Software Reliability and Safety in Nuclear Reactor Protection Systems

2. NUREG-0800, Chapter 7, Branch Technical Position HICB-14, for Software Safety Plans

5 Software Engineering Process

Software engineering is a set of formal elements (methods, tools, documents, practices, standards and procedures) applied during each phase of the software life cycle. The software life cycle phases defined in this plan are the same as those defined in the Standard Review Plan [4.2.5(2)]. Figure 1 provides a comparison of the two software life cycles. A well-defined software engineering process, implemented in a traceable, planned, and orderly manner is essential for the development and maintenance of high quality software.

Figure 1 shows the software life cycle phases. These phases are defined below:

1. Planning - the definition of project scope, methodologies, and resources needed to develop and maintain the deliverable software. The evolution of the planning activities includes refinement of methodologies, reevaluation of resources, and

updating schedule estimations and risk based on knowledge gained through the evolution of the remaining lifecycle.

2. Requirements - the definition of detailed functional and performance requirements, design constraints, and validation criteria.

3. Design - the methodical process that transforms requirements into:

   a. An architectural representation of software, and

   b. A detailed representation of software.

4. Implementation - transforming the human-readable design representation into machine intelligible form in a programming language, and testing the machine intelligible form for correct operation using white-box testing.

5. Integration – black box testing that:

   a. Tests for potential defects (errors) in the software module interfaces, and

   b. Evaluates the performance of the software when installed in the hardware prototype.

6. Validation - testing that:

   a. Ensures that the software-based products are operational and conform to all functional and performance requirements (both software and hardware as would be required by a typical FAT).

   b. ensures that equipment designed and implemented at various vendors integrates with other vendors' systems.

7. Installation – definition of processes and procedures for the physical installation of the software based system and the preparation for operation and maintenance of software packages, which includes the preparation of ;

   a. Installation Procedures,

   b. Installation Tables,

   c. Installation Test Procedures and Test Cases,

   d. Installation Reports,

   e. Operations Manuals,

f. Maintenance Manuals, and

g. Training Manuals.

8. Operation & Maintenance – the functional and operational life of the software package. The use, operation, maintenance, calibration, surveillance, and other processes associated with use of the system, based on data provided in the operation manual, documentation, and procedures provided with each system. Maintenance of the software includes procedures to maintain and fix any operational anomalies, but typically does not include adding new functions or features.

The first five discrete phases are mapped to an evolutionary software life-cycle model, shown in Figure 4. The evolutionary model uses the same phases as a waterfall model and all functional activities remain the same, however, it more accurately represents how the work is typically done.

Unlike the waterfall model, an evolutionary model expects and plans that these basic phases will be repeated, each time evolving the same set of output documents, code, and other work products. The outputs evolve throughout the process, and are formally verified and issued in the final evolution.

The first evolution milestone is the completion of the prototyping and concept validation. In this phase, risks are reduced by activities that prove the concepts workable. The preliminary software architecture is designed in this phase and refined as necessary in subsequent evolutions. All documents and outputs are generated for the Planning through the Integration phase. No validation testing is done during prototyping and concept validation. There are no baseline reviews in this evolution.

The second and subsequent evolutions build on the prototyping and previous evolutions. This allows for the evolution of the documentation to the detailed level, preparing for the final evolution. Documents are developed and enhanced as part of this process. Again, as before, draft documents are informal and not verified. Final validation testing is not completed, as it is a separate formal process. However, informal validation testing is encouraged. No baseline reviews are done.

Design reviews should be done at the completion of each evolution. These design reviews are done on informal and unverified documents. These reviewed documents and the design review comments and resolutions are configuration items to be placed as archived, design documents in a DRF. At least one such design review shall occur after the first or second evolution.

The final iteration is the formal evolution, verification, and issue of each output. When all items for a particular phase are complete, baseline reviews are completed for each phase.

The conclusion of the final integration software life cycle phase will be defined by the production of a set of verified documents.

The final phases (Validation, Installation, and Operations & Maintenance), while not depicted on Figure 4, are done in sequence, in the same manner described in this SMP.

Issues or errors uncovered in the final phases will be resolved by re-entry into the evolutionary process, at a point appropriate for resolution of the issue or error.

A list of the specific documents produced in each life cycle phase is provided in Tables 1 through 8. These documents, with exception of the documents, which are applicable to all software life cycle phases (e.g. SMP or SQAP), shall serve as the input documents for the next software life cycle phase. A summary of the output documents produced at each phase is depicted graphically in Figure 2.

The document retention methods for all output documents defined in the Software Engineering Process are defined in SCMP {see SCMP [4.1.2(1)]}.

5.1 Planning Phase

The purpose of the Planning Phase is to:

- Define the scope of the software-based product, and identify and evaluate the overall objectives, design requirements and required functionality,
- Identify the quality assurance classification of the software-based product based on the safety significance of the function, defined by work performed outside the software development process,
- Develop a strategy to identify and evaluate the essential resources (standards, methodology(s), documents, and tools) and project risks to accomplish the software-based product objectives,
- Generate and evaluate the management plans (i.e., SMP, PP, SCMP, SSP, SVVP, etc.), and
- Define the integrated project schedule (identifying the tasks to be performed for all software systems, on an individual system basis) and establish short, timely milestones for each system.

Input Documents:

- MMIS and HFE Design Implementation Plan [4.1.1(2)],

- System Design Descriptions,
- Logic Diagrams (including I/O, FX, and Setpoint data),
- Piping & Instruments Diagrams, and
- Human Factors Engineering (HFE) Analyses Reports.

Output documents, verification documents and verification methods:

Table 1 Planning Phase Output Documents

Output Documents[3]	Verification[4]	Method
1. SMP	MMIS and HFE Design Implementation Plan	Design Review
2. SDP	MMIS and HFE Design Implementation Plan	Design Verification
3. SSP	MMIS and HFE Design Implementation Plan, Standard Review Plan	Design Verification
4. SVVP	MMIS and HFE Design Implementation Plan	Design Review
5. SCMP	MMIS and HFE Design Implementation Plan	Design Review
6. SIntP	SMP, MMIS and HFE Design Implementation Plan	Design Review
7. SQAP	MMIS and HFE Design Implementation Plan	Design Review
8. SIP	MMIS and HFE Design Implementation Plan	Design Review
9. SOMP	MMIS and HFE Design Implementation Plan	Design Review
10. STmgP	MMIS and HFE Design Implementation Plan	Design Review
11. Planning Baseline Review Record	MMIS and HFE Design Implementation Plan	None

Quality goals:

- Identify all software-based products requirements, specifications and resources, and

[3] All output documents are abbreviated as defined by the acronyms in section 3.

[4] MMIS = Man Machine Interface System and HFE = Human Factors Engineering.

- Standardize procedures, conventions, and methodology(s).

5.1.1 Equipment Design Requirements - SDDs or System Design Specifications, LDs (Including IO/FX/Setpoint data) and P&IDs

System Design Descriptions (SDDs), Hardware/Software Specifications (HSS), Logic Diagrams (LDs), IO/FX/Setpoint data, HFE evaluations and analyses, and Piping and Instrumentation Diagrams (P&IDs) translate nuclear safety, operational, technical, and contractual requirements into the design, configuration, interface characteristics, and operational descriptions (functionality and performance) of plant systems. These design documents serve as the design input documents for those systems comprised of, or containing software-based products. The RPTE shall be responsible for ensuring that these design documents have been verified in accordance with the methods defined in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and that all contractual agreements and technical requirements have been incorporated into these input documents. Upon the completion of the evolutionary phase, the applicable documents and their revision number shall be documented in the Planning Baseline Review Record and forwarded as the input documents for the next software life cycle phase.

5.1.2 Management Plans

5.1.2.1 Software Management Plan

The Software Management Plan (SMP, this plan) is the controlling document for managing the development of all I&C products containing software. The SMP defines the technical development processes necessary to satisfy the requirements and specifications for products containing software, in accordance with the MMIS and HFE Design Implementation Plan. For all I&C products containing software, the SMP establishes standards, conventions, and design processes to be followed during software life cycle phases. A Software Development Plan (see Subsection 5.1.2.2) shall also be developed to define the managerial processes, deliverables, and production schedule for each I&C product containing software. The SMP is a configuration item (CI), and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)] and SCMP [4.1.2(1)] respectively.

5.1.2.2 Software Development Plan

The Software Development Plan (SDP) defines the managerial processes necessary to accomplish the design and development of the I&C software-based products. The SDP shall be developed as a supplemental document to the SMP. The SDP requires a separate work package to be prepared to support development for each system or set of systems. Work packages are used as a tool to aid in the overall project management activities (i.e., establish the project deliverables, schedules, constraints, milestones, resources and risk management procedures) and as such may be updated throughout the course of the project. Work packages shall be developed in accordance with GEEN EOP 25-5.00 [4.1.2(3g)]. This information is to be configured and maintained by the Responsible Technical Project Engineer in accordance with standard GEEN project management practices. The SDP is a configurable item, and is subject to verification and

configuration management as specified in the SVVP [4.1.2(2)] and SCMP [4.1.2(1)] respectively.

5.1.2.3 Software Configuration Management Plan

The Software Configuration Management Plan (SCMP) [4.1.2(1)] is developed as a supplemental document to the SMP, as specified in the MMIS and HFE Design Implementation Plan. The SCMP documents the methods to be used for:

1. identifying I&C configuration items,
2. controlling and implementing changes, and
3. recording and reporting change implementation status.

The individual(s) responsible for the software configuration management process shall be identified. The SCMP is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)] and SCMP [4.1.2(1)] respectively.

5.1.2.4 Software Verification and Validation Plan

The Software Verification and Validation Plan (SVVP) [4.1.2(2)] is developed as a supplemental document to the SMP, as specified in the MMIS and HFE Design Implementation Plan [4.1.1(2)]. The SVVP:

1. describes the software V&V organization,
2. describes the V&V methods and procedures, qualification requirements, and responsibilities of individuals performing V&V,
3. defines the V&V management, activities, and tasks at each software life cycle phase, and
4. establishes the criteria for beginning and ending each V&V activity.

The activities defined in the SVVP help assure the software quality of the software-based products. The SVVP is a configurable item, and is subject to verification and configuration management as specified in the SVVP (itself) [4.1.2(2)] and SCMP [4.1.2(1)] respectively.

5.1.2.5 Software Safety Plan

For the safety-related software (Quality Class Q), the Software Safety Plan addresses the potential software risks. It exhibits the following characteristics listed below:

1. Specifies the purpose and scope of the software safety activities.
2. Defines the responsibilities of the person or people performing the software safety tasks.

3. Defines the method and authority by which software may be rejected, including previously developed software (see Section 5.2.11), support software and tools (see Section 5.2.9), and third party software (see Section 5.2.10) if the software cannot be shown to be adequately safe.

4. Defines the resources required for verification of software safety analysis tasks, including qualification and training requirements.

5. Describes the management of software safety activities, including how the safety activities are integrated and coordinated between the development team and other organizations (i.e., Quality Assurance, Configuration Control Management, vendors).

6. Describes the safety analyses and methods to be performed and documented in each of the principal design documents such as the HSS, IPS, SDS, source code, and test procedures (see Sections 5.2 through 5.6).

7. Describes the software safety documentation requirements, including configuration management of the software safety documents.

8. Defines any safety-related tests that are not included in the Software Verification and Validation Plan [4.1.2(2)].

IEEE 1228, Software Safety Plans [4.2.2(2)], presents an acceptable format for the Software Safety Plan [4.1.2(4)]. The SSP when combined with the other GEEN nuclear safety plans, programs, procedures, and processes, provides compliance with IEEE 1228. The Software Safety Plan is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)] and SCMP [4.1.2(1)] respectively. The Software Safety Plan basis is defined in NUREG-0800, Chapter 7, Branch Technical Position HICB-14, and provides compliance with IEEE Standard 1228.

5.1.2.6 Software Integration Plan

A Software Integration Plan (SIntP) shall be developed in accordance to the MMIS and HFE Design Implementation Plan [4.1.1(2)], Reg. Guide 1.170 [4.2.4(3)], and Reg. Guide 1.171 [4.2.4(4)] to establish the procedures and guidelines for the intended testing activities (i.e., module, integration, and validation testing). The plan shall identify:

1. the test items, approach, and the features to be tested,

2. the test methods and requirements for each testing activities,

3. the testing tasks to be performed

4. test environment,

5. risks requiring contingency planning, and

6. the requirements needed to develop the system build description.

The Validation Test Procedures and Test Cases Specification (see Section 5.3.3) shall be prepared and verified prior to starting the validation testing, which can be started in any phase beginning with the Requirements Phase but shall be completed prior to staring the Validation Phase. The RE shall decide in which phase to baseline the test procedure, but the test procedure shall be baselined prior to starting the Validation Phase and as such this procedure will list it as an input to that phase. IEEE 829, Standard for Software Test Documentation [4.2.2(3)], presents the acceptable format for the Software Integration Plan. The Software Integration Plan shall be reviewed and updated when necessary. The Software Integration Plan is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)] and SCMP [4.1.2(1)] respectively.

5.1.2.7 Software Quality Assurance Plan

A Software Quality Assurance Plan (SQAP) shall be developed as a supplement to the SMP, to provide methods by which the quality of software activities shall be gauged. The SQAP shall:

1. define the SQAP activities,

2. provide the technical and administrative direction necessary to implement the defined SQAP activities,

3. outline the organization (herein referred to as the SPE),

4. define the minimum SPE organization member qualifications,

5. define the SPE organization member responsibilities,

6. provide methods by which each activity shall be executed,

7. define the measurements and metrics by which to gauge procedural and process adherence, and

8. define the measurements and metrics by which to gauge the quality and safety goals of the software product.

5.1.2.8 Software Installation Plan

The Software Installation Plan (SIP) shall be developed to establish procedures and guidelines necessary to install software on software based products. Embedded firmware is installed on the instruments in accordance with standard GEEN procedures and as such, physical installation activities for embedded software need not be addressed in the plan, but references to the procedures will be incorporated in the installation plan. Embedded firmware, however, is not exempt from certain Installation Phase activities, which include the generation of training,

operation, and maintenance manuals. The software installation plan shall define the Installation Phase activities, procedures, and methods and shall provide the requirements and guidelines necessary to prepare, execute, and document software Installation Phase activities which includes:

1. Software Installation Procedures
2. Software Installation Reporting
3. Installation Configuration Tables
4. Operations Manuals
5. Maintenance Manuals

5.1.2.9 Software Operation and Maintenance Plan

During the Installation Phase, an operations manual shall be developed for each system or logical group of systems, as specified by the installation plan. A Software Operation and Maintenance Plan (SOMP) shall be developed to be used in conjunction with the Software Installation Plan (SIP) to define any additional requirements and methods to use while developing the operations manual. The Operation manual or manuals for each system, or logical group of systems, shall include:

1. a general description of the operation of the software,
2. a general description of the functions that the software is to perform,
3. a general discussion of the means of carrying out those functions
4. a description of the organizational structure necessary to control the software operation,
5. specification of operator interface stations and actions required to support operation,
6. a description of the responsibilities and authority of the operators,
7. a description of the security requirements for operating the software system,
8. identify the controls needed over operation activities to prevent unauthorized changes to hardware, software, and system parameters,
9. the monitoring activities needed to detect penetration or attempted penetration of the system,
10. contingency plans needed to ensure appropriate response to penetration,
11. a set of indicators used to determine the success or failure of the operating

procedures,

12. a method for measuring, recording, analyzing, and reporting error rates found during operation activities,

13. a description of the procedures necessary to start, operate and stop the software system,

14. a description of procedures for executing the software in all operating modes,

15. procedures for ensuring that the software state is consistent with the plant operating mode at all times,

16. a description of backup procedures for data and code, and the intervals at which backup should occur, and

17. a list of error messages, giving a description of the error indication, the probable interpretation of the error indication, and steps to be taken to resolve the situation.

A software operation & maintenance plan (SOMP) shall be developed to define maintenance procedures and activities to modify and maintain software once the software is placed in service. For each system or logical group of systems, procedures should be developed in accordance with the SMP and provided to the customer. The plan should require the individual system level procedures to identify the controls needed over maintenance activities and maintenance and test equipment to prevent unauthorized changes to in-service hardware, software, and system parameters. At a minimum, the potential for introducing unauthorized changes during repair, testing, and calibration should be addressed. Maintenance could be limited to the process of modifying a software design output to repair nonconforming items or to implement pre-planned actions necessary to maintain performance. Modifications to improve performance or other attributes, or to adapt the design outputs to a modified environment, should be considered design changes and processed accordingly. This plan requires development of the individual system level procedures to provide the backup and restore procedures for each system or logical group of systems.

5.1.2.10 Software Training Plan

During the Installation Phase, a training manual shall be developed for each system or logical group of systems, as specified by the installation plan. A Software Training Plan (STrngP) shall be developed to be used in conjunction with the software installation plan (SIP) to define any additional requirements and methods to use while developing the training manual. The training plan should define:

1. The organization supporting the software-based product training effort,
2. Organizational interfaces and responsibilities,

The training plan shall require that training manuals;

1. Define the personnel required to perform the training,
2. Define the qualification and responsibilities of individual carrying out each training module of the software based product(s),
3. Provide the overall training objectives,
4. Describing the training needs of appropriate plant staff, including operators and I&C engineers and technicians.
5. Provides the methods, techniques, tools and facility required to accomplish the training function.
6. Define testing or assessment criteria to demonstrates the student's knowledge as it relates to the objectives.

5.1.3 Planning Baseline Review Record

The Baseline Review Team (BRT) shall perform a Planning Baseline Review on the documents output from the Planning Phase. This review is conducted to ensure that:

1. the scope for the software-based product has been adequately defined,
2. the design documents have been verified, and all nuclear safety, technical, and contractual technical agreements have been incorporated and are ready to serve as the input documents for the next software life cycle phase, and
3. the resources and standards to be used on the software-based product have been identified and properly documented.

The Planning Baseline Review also serves to review and confirm the completion status formally for all output documentation from that development phase. The Baseline Review Team Task Lead and supporting personnel shall prepare the Planning Baseline Review Record {see SCMP [4.1.2(1)]} to provide a record of the review results of the items that comprise the Planning Baseline Review, and, if necessary, document all non-conformances identified during the baseline review process. If changes are made to the documents after the review has been completed, the changes shall be controlled through the formal change control procedure {see SCMP [4.1.2(1)]} and baseline reviews performed on the changes to those documents. The Planning Baseline Review Record shall be filed in the software project DRF {see SCMP [4.1.2(1)]}.

5.2 Requirements Phase

The purpose of the Requirements Phase is to:

- identify, evaluate and analyze the software and software interface requirements (i.e., user and hardware), and if applicable, show how the constraints and limitations of hardware impact the software,
- identify the requirements for support software and tools (see Section 5.2.9),

- evaluate the resources required, such as, the third party software (see Section 5.2.10) and previously developed software (see Section 5.2.11),
- identify software design requirements, such as:
  - incorporation of defense in depth techniques,
  - assignment of limit checks, and/or default values to prevent software instability,
  - performance of range checks on data entries,
  - identification of assumptions (hardware and software) critical to safety,
  - incorporation of applicable design techniques, such as:
    - graceful degradation,
    - determinism,
    - modularity,
    - readability, and
    - appropriate commenting.
  - software portability and compatibility requirements
  - security

Input Documents:

- SMP,
- SCMP,
- SVVP,
- SDP
- SSP,
- SQAP,
- applicable HSS, SDDs, LDs, (Including IO/FX/Setpoint Data), and P&IDs. Note that most setpoint values are not firm until finalized in later design phases, or even tuned during startup,

- applicable HFE documents.

Output documents, verification documents, and verification methods:

Table 2 Requirements Phase Output Documents

Output Documents	Verification	Method
1. HSS	SDD, LDs, (including IO/FX/Setpoint Data), if applicable P&IDs	Design Verification
2. SBD	SDD and LDs, if applicable P&IDs	Design Verification
3. SRS	HSS, if applicable, LDs and P&IDs	Design Verification and Demonstrate Traceability
4. IPS	HSS, SBD	Design Verification and Demonstrate Traceability
5. Sub-system Schematic	HSS, if applicable, LDs P&IDs, and IPS	Design Verification
6. External Data Communication Protocol Specification	HSS or SRS	Design Verification
7. User's Manuals	HSS, if applicable, SRS, SBD and IPS	Design Verification
8. If applicable, Support Software/Tool and its documentation package	Performed according to the instructions in SMP, SVVP, SCMP and SSP	Design Verification
9. If applicable, Third Party Software and its documentation package	Performed according to the instructions in SMP, SVVP, SCMP and SSP	Design Verification
10. If applicable, Previously Developed Software Evaluation Report	Performed according to the instructions in SMP, SVVP, SCMP and SSP	Design Verification
11. If applicable, supplemental documentation for Previously Developed Software	Performed according to the instructions in SMP, SVVP, SCMP and SSP	Design Verification and Demonstrate Traceability
12. Requirements Baseline Review Record		None

An SRS is used for software-based systems, while an IPS is used for a microprocessor-based system. Both an IPS & SRS are not required for a single system.

Quality goals:

- All design requirements for Quality Class Q software shall be verifiable and traceable to the input documents.

- All verifications are performed in accordance with the SVVP
- Software safety analysis is performed on all outputs and verified in accordance with the SSP.

5.2.1 Hardware/Software Specification

The system Hardware/Software Specification (HSS) supplements the SDD and, if applicable, the LDs (including I/O FX Setpoint Data) and the P&IDs, by documenting the high level, system specific requirements of a software-based product. The HSS describes the algorithms and logic that are too complex to be delineated in the Logic Diagram and P&ID. The HSS, at a minimum, shall include the following requirements:

1. System architecture definition, which may include, for example:
   a. system-level boundaries and interfaces,
   b. system communications requirements to external systems,
   c. major system components (sub-systems or instruments) to the extent applicable,
   d. system configuration,
   e. requirements for system security,
   f. any restrictions required by software or system safety evaluations,
   g. any safety to non –safety-related boundaries and isolation requirements, and
   h. quality assurance classification.
2. System functional and performance requirements, which may include, for example:
   a. system algorithms, calculations, and logic,
   b. response time requirements,
   c. I/O capabilities,
   d. User interface requirements,
   e. modes of operation,
   f. calibration and testability requirements,

g. local HSI and MCR dedicated hard-switch requirements,

h. self-test and/or on-line test requirements,

i. initialization, inputs, processing, outputs, and communications, and

j. filtering, accuracy, and drift requirements.

3. Hardware and software design constraints.

4. Verification and Validation (quality assurance) requirements.

5. Descriptions of major sub-systems or instruments and allocation of subsystem(s) or instrument functions (as required). The HSS shall specifically identify all instrument functions having safety-related application.

6. Additional hardware requirements, which may include, for example:

   a. power distribution (i.e., voltages, current loads, fusing),

   b. cabling requirements, and

   c. qualification requirements (e.g., EMC, environmental, seismic, radiation, temperature, humidity).

Additional definitions may be added depending on the complexity of the system. Appendix A.1 presents one acceptable format for the HSS. The HSS is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], SSP [4.1.2(4)] and SCMP [4.1.2(1)] respectively.

The HSS document and its revision number shall be documented in the Requirements Baseline Review Record and forwarded as an input document for subsequent design activity.

5.2.2 Software Requirements Specification (SRS)

An SRS shall be generated for software-based systems identified in the HSS to further define the functional and performance requirements, specifically those assigned to software. The SRS shall include the following aspects of the software that require additional detailed information not found in the HSS:

1. Information flow and interfaces

   a. hardware,

   b. software,

   c. human system interface (HSI), and

d. data communication.

2. Functional requirements

   a. inputs or attributes,

   b. processing or methods, and

   c. outputs.

3. Design requirements and constraints

   a. data handling and storage requirements,

   b. timing requirements,

   c. security requirements, which shall include methods used to protect the system's configuration or results from inadvertent or malicious alternation,

   d. self test and/on-line test requirements, including surveillance and calibration,

   e. error handling requirements, and

   f. hardware platform constraints (e.g., numerical precision, processing power).

4. Software testing criteria and quality assurance requirements, at least the following shall be included:

   a. acceptance and performance criteria

   b. portability and compatibility requirements,

   c. the requirement that a minimum number of programming languages, compilers, and support packages be used in the implementation of the software, and

   d. the requirement that the use of assembly language programming be minimized to the extent practicable, and preferably eliminated.

Each requirement shall be defined sufficiently to enable software design engineers to:

- design the software, and
- develop a Software Test Procedures and Validation Test Procedures (see Section 5.3.3) to verify the requirements.

Appendix A.2 presents one acceptable format for the SRS. The SRS is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively. An SRS is not normally used in the design process for *microprocessor-based systems*. For these systems, a System Block Diagram (SBD, see Section 5.2.3) shall be generated and issued into the GEEN document control system {see SCMP [4.1.2(1)]}. To document the software specific requirements of microprocessor-based systems, the Instrument Performance Specification (IPS) (see Section 5.2.4) shall contain a special software requirement section.

5.2.3 System Block Diagram

The System Block Diagram (SBD) is generated for hardware-based designs that contain microprocessor(s) to control the internal functions (such as microprocessor-based instrumentation). A SBD can be generated for software-based systems, if the SBD provides information useful for documenting or understanding the design. The SBD provides generic details about the system-level operation and inter-connection of the individual instruments that comprise the essential controls equipment. For very simple or self-contained systems where all information is described on the sub-system schematic, an SBD may not be necessary.

A SBD shall be generated to define the system architecture and high-level system partitioning for hardware-based systems. The SBD, at a minimum, shall include the following information:

1. Identification of each instrument used in the system
2. Physical interconnections between each instrument (including definitions of each type of signal, signal media, and/or type of protocol in case of a communication link)
3. Physical separation information
4. Power distribution information
5. Divisional separation information:
    a. safety-related separations
    b. nonsafety-related separations
    c. associated safety separation
6. Power arrangement information.

The SBD is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

5.2.4 Instrument Performance Specification (with Software Requirements)

A complete and clearly detailed Instrument Performance Specification (IPS) shall be generated for each instrument identified in the HSS. The Instrument Performance Specification (IPS) shall contain the hardware and software performance requirements for a microprocessor-based system. The IPS shall contain the requirements that the software and/or hardware be designed with sufficient margin to perform as designed under conditions of maximum stress.

1. The IPS shall include, as a minimum, the following hardware requirements:
    a. Inputs/Outputs, including ranges, accuracy, and data rate capabilities,
    b. Design Features that provide administrative control,
    c. Initialization requirements,
    d. Features for handling diagnostic and other faults and failures in the instrument and in external devices,
    e. Self test and/or on-line test requirements,
    f. HSI interface requirements,
    g. In-service test features and/or diagnostics,
    h. Interrupt features,
    i. Features that support calibration and surveillance testing, and
    j. Calibration procedures.
2. The IPS shall include, as a minimum the, following *product specific* software requirements:
    a. Process input definition including ranges, accuracy, sampling intervals, limit checks, and range bounding,
    b. Algorithms to be programmed with consideration of abnormal inputs,
    c. All data required by algorithms,
    d. All outputs, including ranges, accuracy, and update intervals,
    e. Initialization requirements,
    f. Self test and/or on-line test requirements (including power-on diagnostic testing as applicable),

g. HSI interface requirements,

h. In-service test features and/or diagnostics,

i. Timing and response time requirements including overall system response time, and

j. Security requirements to limit changes to critical functions and data. In addition, methods used to protect the information from inadvertent or malicious alternation.

3. When addressing software requirements, the IPS shall include the testing criteria and quality assurance requirements specified in Item 4 of the Software Requirements Specification section (see Section 5.2.2) of this plan. The IPS shall also include, at a minimum, the following *generic* software requirements:

   a. independence, separation, diversity, traceability, and defensive design techniques or features to be employed in the safety-related software design as described in Reg. Guide 1.172 [4.2.4(2)],

   b. safety-related software shall reside in read-only memory that retains its contents without external or internal power (e.g., ROM, EPROM), and

   c. system tuning parameters shall reside in changeable, non-volatile memory and that such parameters be bounded to their allowable ranges.

The software requirements section of the IPS shall identify which software requirements are safety-related (i.e., which software requirements are necessary to implement the safety-related, Quality Class Q, instrument functions specified in the HSS). Appendix A.3 presents one acceptable format for the IPS. The IPS is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

5.2.5 Sub-System Schematic

Where appropriate, a sub-system schematic shall be prepared to sketch the instrument layout for the microprocessor-based system. The Sub-system Schematic includes a block diagram of the sub-system showing the hardware modules to be used (including their physical locations and inter-connections) and instrument inputs and outputs. Additional information may be included, as necessary. The Sub-system Schematic is not applicable for the design and development of a software-based system. The Sub-system Schematic is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

5.2.6 Data Communications Protocol

The Communication Protocol Specifications may be divided into:

1. External Data Communication Protocol Specification(s) (see Subsection 5.2.6.1), and

2. Internal Data Communication Protocol Specification(s) [Note: This Specification(s) shall be prepared during the Design Phase (see Section 5.3.2)].

5.2.6.1 External Data Communications Protocol Specification(s)

An External Data Communication Protocol Specification shall be prepared for each communication link established with a system external to the software-based product (i.e., inter-system communication).

Each specification shall identify and define the data interfaces used, including message structure and format. The specification(s) shall include a timing analysis. The specification(s) shall include a definition of the error detection and recovery processes required for this communication link. The specification is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

5.2.7 User's Manual

A User's Manual shall be prepared to provide the system operator with the necessary instructions for system use and operation. The User's Manual is prepared from the HSS, and if applicable, the SRS, SBD, and IPS, and other design documents necessary to describe the finished systems.

The preparer shall design the manual contents and format specifically to meet the needs of the intended users (e.g., GEEN engineers and utility Operations, Maintenance, Engineering, Installation, and Test staff). The User's Manual shall specify and describe the required data and control inputs, input sequences, options, program limitations, and other activities or items necessary for successful execution of the software (or operation of the instrument, in the case of microprocessor-based systems). All error messages shall be identified and corrective actions described, or, for software-based systems, reference shall be provided to vendor's manuals that provide additional platform-specific error messages. At a minimum, the User's Manual shall describe and provide:

- definition and explanation of all the system displays (including displayed data, and, if applicable, input devices functions such as keyboards, and user-entered data),

- definition of Display/Control hierarchy (including operational details),

- description of all operating modes and entry, exit, and state transition requirements and restrictions for each ,
- definition of all symbols, abbreviations, and acronyms, and
- definition of all error messages and their criticality.

The development of the User's Manual is initiated during the Requirements Phase to provide:

- a means of verifying the requirements for software-based product,
- guidance for the design of the software-based products,
- guidance for the evaluation of the human factors engineering techniques employed, and
- assistance in writing the Validation Test Procedures.

The development of a User's Manual is expected to evolve through the Design and Implementation Phases, as design requirements mature.

The User's Manual is a part of the System/Instrument Operation and Maintenance Manual. The User's Manual is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

5.2.8 Support Software and Tools

Support software (including software tools) is the software used to "support" the development of the software-based products throughout the software development process. These items support the development of the software, but the vendor might choose not to deliver these tools as part of the final product. Examples of the support software and software tools would include:

- the software tools used to map IO/FX/Setpoint Data to generate data applications, build data files/databases, or generate graphic displays, and
- compilers, linkers, and other software tools (i.e., emulation tools, flow signal simulators, test tools, software safety analysis tools, block ware tools) used to support, develop, and/or execute the application software.

In general, support software is purchased through a commercial outlet. Purchased software is categorized as follows:

- Commercial - Software distributed in large volume to perform general purpose application such as spreadsheet, and database applications, and

- Special Purpose - Software distributed in relatively small volume to perform special purpose type of functions (e.g., emulation tools, flow signal simulators).

At the option of the RE, the support software/tools may be tested to verify that it is adequate for the intended use. If this option is adopted, then evaluation criteria must be established and documented in the Support Software/Tool Report. The Responsible Engineer shall ensure that the verification level is commensurate with the system's degree of safety importance.

At a minimum, a report shall be required for each Support Software/Tool used for Quality Class Q software and shall contain:

- the identity of the support software/tool and the version being evaluated,
- description of the purpose of the support software/tool,
- the acceptance criteria for the support software/tool,
- description of the verification methods/review activities used to confirm the applicability of the support software/tool,
- the evaluation results, and
- a summary evaluation of the tool's acceptability, along with any restrictions on use of the tool.

In general, commercial software is widely used throughout many industries and is thoroughly checked by both vendor regression testing as well as extensive user testing. Therefore, less rigorous evaluations are acceptable. However, Special Purpose software may require more rigorous and specific testing to ensure proper operation and suitability for use in its intended application.

If testing of the support software/tools is not performed, then the software or analysis produced using the support software/tools must be subjected to V&V activities (see SVVP [4.1.2(2)]).

If "in house" development of support software is necessary, information and data required to design the support software shall be defined, documented (e.g., in a Application Data Specification), and developed in accordance with a process equivalent to the process outlined in GEEN EOP 40-3.00, Engineering Computer Programs [4.1.2(3e)].

For Quality Class Q Software, Support Software/Tool Reports are configurable items, and are subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively. The Support Software/Tool Report and the associated documentation of the support software/tool shall be included in the Support Software/Tool Documentation Package and filed in the software project DRF. The support

software/tool and its documentation package shall be included as part of the appropriate software life cycle phase baseline review and placed under configuration control {see SCMP [4.1.2(1)]}.

5.2.9 Third Party Software[5]

Third party software is software procured from outside sources and incorporated into the final software-based product. Examples of such software would include special hardware test algorithms, standard communication protocol stacks, and linkable software libraries.

Prior to the procurement, the RE shall review the third party software documentation to ensure that the software meets the applicable standards stated in Appendix B.

Upon acquisition of the software, and prior to its incorporation into the final software package, the performance capabilities of third party software shall be evaluated to ensure that it meets the requirements and constraints defined in the HSS and other applicable design documents. The level of detail of the evaluation shall commensurate with the system's degree of quality and safety importance. At a minimum, the following tasks shall be performed for Quality Class Q software:

1. evaluate the third party software, including all relevant documentation, to ensure that it performs as intended (including all software interfaces),
2. evaluate the software quality assurance program followed by the vendor during software design and development,
3. evaluate problem/error report and tracking procedures, and
4. analyze all reported problems/errors and resolution to these problems/errors to ensure that there is no known unresolved problem or nonconformance, which may impact the safety functional or performance requirements in the application.

The result of the evaluation shall be documented in the Third Party Software Evaluation Report and all detected discrepancy(s) and/or error(s) shall be reported to and analyzed by the software vendor.

At a minimum, the Third Party Software Evaluation Report for Quality Class Q software shall:

1. identify the third party software and the version being evaluated,
2. describe the purpose of the Third Party Software,
3. specify the acceptance criteria for the evaluated Third Party Software, and
4. describe the evaluation methods and/or procedures used to confirm the applicability of the third party software.

[5] Third party software refers to the purchased software libraries or packages used as part of the development of a software-based product, not the software procured from vendors, such as the Non Essential DCIS vendor.

Prior to the incorporation of any Third Party Software into the final software package, the evaluation report shall be verified in accordance with the methods defined in the SVVP [4.1.2(2)], and the SSP [4.1.2(4)]. The Third Party Software Evaluation Report and the associated documentation shall be included in the Third Party Software Documentation Package and shall be kept in the software project DRF.

The Third Party Software Documentation Package shall be included as part of the appropriate software life cycle phase baseline review, and placed under configuration control {see SCMP [4.1.2(1)]}.

5.2.10 Adaptation of Previously Developed Software

Application of previously developed software packages may be acceptable if the following acceptance criteria are met.

The RE shall prepare an adequacy evaluation report to document the basis for the package's acceptance for Quality Class Q software. At a minimum, the report shall:

1. describe the function of the software package and identify the previous application(s) of the package,

2. describe the operating experience of the software-based product(s) identified in Item 1, such as locations of installation, number of units installed, and a conservative estimate of the total operating hours accumulated,

3. describe the licensing experience with the software-based product(s),

4. identify the relevant engineering documents that are available and determine their status and repository, accompanied by a description of differences to what is required for new software,

5. identify the relevant problem reports, including Field Deviation Disposition Requests [4.1.2(3e)], service requests from customer(s) due to equipment failure or the result of non-conformances, and the associated resolution reports,

6. ensure that there is no unresolved problem or nonconformance in the PDS which impacts the safety function in the proposed application,

7. describe the verification performed to validate that the software package meets the functional requirements identified in the HSS, and if necessary, identify any modification(s) needed to meet the proposed I&C application,

8. identify any additional activities to be performed,

9. demonstrate that all requirements are met,

10. compare the previously applied software quality assurance requirements to the requirements outlined in the SMP, address the differences, and evaluate if operating experience compensates for any identified weaknesses,

11. A system integrity investigation for Quality Class Q equipment shall be done per requirements in IEEE 7-4.3.2 Section 5.5.1 [4.2.2(4)], and

12. Identify the support tools necessary to implement the required changes, and determine their status and repository.

If it cannot be demonstrated that the criteria above are met, the software package may be applied to the project only after the product has been re-engineered and all lifecycle documents, peer reviews, and testing prescribed by this plan for new products have been generated.

The adequacy evaluation report shall be verified in accordance with the methods defined in the SVVP [4.1.2(2)] and filed in the software project DRF. The implementation of new requirements imposed on the previously developed software package (i.e., package modifications) shall follow the engineering process outlined in this plan. The supplemental documentation shall be verified in accordance with the methods defined in the SVVP [4.1.2(2)] and the SSP [4.1.2(4)], controlled, filed in the software project DRF, and included in the baseline review.

5.2.11 Safety Analysis of Requirements

Software safety issues must be addressed at each phase of the software development for Quality Class Q software. For the Requirements Phase, safety requirements are identified and documented in the HSS and IPS. The HSS identifies the instrument functions having safety-related implications [see Section 5.2.1(5)] and the software requirements section of the IPS identifies and documents the software requirements that are necessary to implement those safety-related functions [see Section 5.2.4(3)]. Quality Class N requirements shall be evaluated by the software safety team (part of the SPE) to ensure that these non –safety-related functions have no means to falsely initiate or prevent the system from performing the required safety function(s). The Software Safety Plan [4.1.2(4)] outlines the methods and procedures to be used in assuring that the design as described in these documents adequately address safety.

5.2.12 Requirements Baseline Review Record

The BRT shall perform a Requirements Baseline Review on the documents generated during this phase. This review ensures that the design requirements have been documented, and are traceable to the hierarchy documents. The Requirements Baseline Review also serves to formally review and confirm the completion status of all output documentation. The BRT Task Lead and reviewers shall prepare the Requirements Baseline Review Record to provide a record of the review results of the items that comprise the Requirements Baseline Review and, if necessary, document all non-conformances identified during the baseline review process {see SCMP [4.1.2(1)]}. If changes are made to any of the documents after the Baseline Review has

been completed, they shall be controlled through the formal change control procedure {see SCMP [4.1.2(1)]}. The Requirements Baseline Review Record shall be shall be filed in the software project DRF {see SCMP [4.1.2(1)]}.

5.3 Design Phase

The purpose of the Design Phase is to:

- configure the software requirements into well structured components,
- identify the software modules that must be developed (through either the writing of new software modules, or the modifications of the previously developed software modules),
- define the software coding conventions and guidelines, and
- develop test procedure and test cases required to test all software functions.

Input Documents:

- SRS,
- SBD,
- IPS,
- External Data Communication Protocol Specification,
- User's Manual,
- SMP,
- SCMP,
- SVVP,
- SSP,
- SIntP.

Output documents, verification documents and verification methods:

Table 3 Design Phase Output Documents

Output Documents	Verification	Method
1. Software Design Specification	User's Manuals, External Data Communication Protocol Specification, and either SRS (software-based system), or SBD, Sub-system Schematic and software section of IPS (microprocessor-based system)	Design Verification and Traceability Analysis
2. Validation Test Procedures and Test Cases Specification	HSS, User's Manuals and Software Integration Plan, for software-based systems, SRSs, for microprocessor-based systems, SBD, External Communications Protocol Specification, and IPSs.	Design Verification and Traceability Analysis
3. Software Conventions and Guidelines Document	Performed according to the instructions in SMP, SCMP, SVVP and SPP	Design Verification
4. Internal Data Communication Specification Protocol	IPS	Design Verification
5. if applicable, Support Software/Tool and its documentation package	Performed according to the instructions in SMP, SCMP, SVVP and SSP	Design Verification
6. if applicable, Third Party Software and its documentation package	Performed according to the instructions in SMP, SCMP, SVVP and SSP	Design Verification
7. if applicable, supplemental documentation for Previously Developed Software	Performed according to the instructions in SMP, SCMP, SVVP and SSP	Design Verification
8. Design Baseline Review Record		None

Quality goals:

- All software modules defined, including their interfaces and internal logic,
- All software modules traceable to a requirement,
- All verifications are performed in accordance with the SVVP, and
- Software safety analysis is performed on all outputs and verified in accordance with the SSP.

5.3.1 Software Design Specification

The Software Design Specification (SDS) documents how the system software will be structured to implement the software requirements. It shall describe the software architectural layout, software structure, algorithms, data design and structure, data flows and error handling techniques. The SDS shall be developed using the design requirement techniques defined in the

Requirements Phase. This information shall be described in sufficient detail to allow an experienced software engineer to write the software programs. For the implementation of previously developed software, the SDS shall identify all required modifications.

The SDS shall identify all software modules and, if applicable, Third Party Software that will become part of the final software package. The SDS shall also identify all software modules having safety-related implications (i.e., those software modules necessary to implement the safety-related functions described in the HSS and the safety-related requirements specified in the IPS). The SDS is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

5.3.2 Internal Data Communication Protocol Specification

An Internal Data Communication Protocol Specification(s) shall be prepared for each communication link established between components within the same software-based product (i.e., intra-system communication; between instruments in the same system or modules within the same instrument).

Each specification shall identify and define the data interfaces used, including message structure and format. The specification shall include a timing analysis. The specification is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

5.3.3 Validation Test Procedures and Test Cases Specification

The Validation Test Procedures and Test Cases Specifications define the procedures and individual test cases that test the completed software package for compliance with overall system requirements. These test cases define the inputs, predicted results, and a set of execution conditions required to perform each test. The Validation Test Procedures and Test Cases Specification consists of the following elements:

1. the definition of test cases (test steps) that exercise the software functions,
2. the acceptance criteria for each test case, and
3. the documentation and review of all test results.

The Validation Test Procedures and Test Cases Specification shall be traceable to the requirements in HSS, User's Manuals, and either the SRSs for software-based systems or the SBD and IPSs for microprocessor-based systems. The Validation Test Procedures and Test Cases Specification is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

5.3.4 Software Conventions and Guidelines Document

The Software Conventions and Guideline document establish the software coding guidelines. This document provides guidance how to write software source code that is efficient, understandable, testable, maintainable, safe, reliable, and consistent throughout the software-based product. The software conventions and guideline document shall specify, at a minimum:

1. the coding conventions, color conventions, code formats, and code documentation formats to be used,
2. the modular design of the software modules,
3. the standard software structure to be used,
4. that the software design will include self-supervision of control flow and data,
5. that the final source program must be readable from start to end,
6. the common regions and techniques for the declaration of public and global data variables (e.g., the requirement that constant parameters not be hardcoded within the body of functions or procedures),
7. the naming convention of the computer-based software configuration items (e.g., source code listings, test data and databases), and
8. the operation mode portion of Quality Class Q software must be deterministic.

The Software Conventions and Guideline document is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

5.3.5 Safety Analysis of Software Design

Software safety issues must be addressed at each phase of the software development for Quality Class Q software. For the Design Phase, the SDS and the Validation Test Procedures and Test Cases Specification track the implementation and test of the software safety requirements. The SDS documents the software modules that have safety-related implications (i.e., that are necessary to implement the safety-related functions specified in the HSS and the software safety requirements specified in the IPS). The Validation Test Procedures outline the test steps required to test the instrument, with special emphasis on the normal and off-normal operation of the safety-related functions. The Software Safety Plan [4.1.2(5)] outlines the methods and procedures to be used in assuring that the design as described in these documents adequately address software safety.

5.3.6 Software Design Baseline Review Record

The BRT shall perform a Design Baseline Review on documents generated during this phase. This review ensures that the requirements have been incorporated into the design, that the peer reviews considered the requirements and the design, and that the testing document validate the requirements, and that the design is traceable to the requirements documents. The Design Baseline Review also serves to review and confirm the completion status of all output documentation formally. The BRT Task Lead and reviewers shall prepare the Design Baseline Review Record to provide a record of the review results of the items that comprise the Design Baseline Review, and, if necessary, document all non-conformances identified during the baseline review process. If changes are made to any of the documents after the Baseline Review has been completed, they shall be controlled through the formal change control procedure {see SCMP [4.1.2(1)]}. Non-conformances and their resolutions are tracked through an external tracking system. The Design Baseline Review Record shall be filed in the software project DRF {see SCMP [4.1.2(1)]}.

5.4 Implementation Phase

The Implementation Phase is divided into two sections - the software module coding section and the software module test section. Once the two sections are completed a Baseline Review is performed.

The purpose of the Implementation Phase is to:

- transform the software design into software source code using the defined Software Conventions and Guidelines Document,
- perform a review of the completed software modules for coding style conformance and the discovering of potential errors,
- conduct module test of the software modules to uncover errors, and
- verify that the code created performs properly, that it correctly implements the design as specified in the SDS, that error detection is appropriate, and that the design is testable.

Input Documents:

- SDS,
- Internal Data Communication Protocol Specification,
- External Data Communication Protocol Specification,
- Software Conventions and Guidelines Document,
- SMP,

- SCMP,
- SVVP,
- SSP,
- SIntP.

Output documents, verification documents and verification methods:

Table 4 Implementation Phase Output Documents

Output Documents	Verification	Method
1. Source Code	SDS, Software Conventions and Guidelines Document, and for microprocessor-based system, Internal Data Communication Protocol Specification	Code Review, Traceability Analysis and Module Test
2. Module Test Report	SDS, Software Integration Plan	Design Verification
3. If applicable, Support Software/Tool and its documentation package	Performed according to the instructions in SMP, SCMP, SVVP and SSP	Design Verification
4. If applicable, Third Party Software and its documentation package	Performed according to the instructions in SMP, SCMP, SVVP and SSP	Design Verification
5. If applicable, supplemental documentation for Previously Developed Software	Performed according to the instructions in SMP, SCMP, SVVP and SSP	Design Verification
6. Implementation Baseline Review Record		None

Quality goals:

- Meet coding and design standards outlined in the Software Conventions and Guidelines Document,
- Successful implementation of the software design outlined in SDS,
- Successful completion of all software modules testing,
- All verifications are performed in accordance with the SVVP, and
- Software safety analysis performed on all outputs, documented during the verification, and in accordance with the SSP.

5.4.1 Coding

5.4.1.1 Software Source Code

Software Coding is the translation of the detailed software design (as specified in the SDS) into machine-readable code, using a predetermined programming language. The software programs must be implemented in accordance to the appropriate Software Conventions and Guidelines Documents (see Section 5.3.4). Each software module shall contain comments of sufficient detail to allow a qualified software engineer to understand the design intent and logic flow. At a minimum, each software module shall contain a statement of the procedure's purpose or function, and a brief description of the algorithm implemented. Each software module shall identify whether or not the software module is qualified for use in safety-related (Q) or nonsafety-related applications (N).

For safety-related applications, the standards, conventions, and techniques used to implement the software design for the support software shall be the same as the process required for the software installed in the plant. The storage and configuration management requirements of the software source code are outlined in the SCMP [4.1.2(1)].

5.4.1.2 Code Reviews

A code review (code analysis) of each software module shall be performed to assure that the software correctly implements the design specified in the SDS (see Section 5.3.1), and does so in a manner that is compliant with the guidelines outlined in the Software Conventions and Guidelines document (see Section 5.3.4). Each review shall be performed by a qualified software engineer other than the individual responsible for code implementation. The code review function looks for incomplete or incorrect code, inefficient and un-maintainable implementation, and poor documentation. For Quality Class Q software, code reviews are a formal verification of the source code, and the first bastion in the defense-in-depth validation of the actual implementation, and shall be performed by an independent reviewer who does not work for the same manager as the engineer implementing the code. For Quality Class N software, reviews may be more informal. At a minimum, the code review shall evaluate the code for:

1. Compliance to code conventions and standards,
2. Maintainability,
3. Readability,
4. Portability and re-usability,
5. Proper syntax,
6. Required logical functionality,
7. Reliability,

8. Testability,
9. Adherence to specified requirements, and
10. Security.

These elements shall also serve as the minimum standards by which to measure and gauge the effectiveness and quality of software at this phase. As part of individual work packages, required by the SDP, the minimum and maximum acceptance criteria shall be specified.

All discrepancies and the reviewer's comments shall be documented in individual Code Review Data Sheets. The Code Review Data Sheet shall at least include the following information:

1. The identity of the responsible software module designer,
2. The identity of the code reviewer,
3. The software modules reviewed its version/revision level and quality assurance classification, i.e. Quality Class Q or N,
4. Discrepancies found and the reviewers comments, and
5. Resolution to each discrepancy and comments by the responsible software module designer.

The software module designer shall address and resolve each discrepancy and/or comment. Completed source code which has been reviewed and tested (i.e., Module Test and/or Integration Test has been completed), shall be re-reviewed each time a modification is made. All individual Code Review Data Sheets shall be included as part of the Module Test Report (see Section 5.4.2.1).

5.4.2 Software Module Testing

Software module testing shall be performed to ensure that the software module or a group of software modules executes as specified in the SDS. Tools (e.g. debuggers) provided in the development environment may be utilized for software module testing.

The Software Integration Plan shall identify the elements to consider for all module testing activities. These elements shall also serve as the minimum standards by which to measure and gauge the effectiveness and quality of software at this phase. As part of individual work packages, required by the SDP, the minimum and maximum acceptance criteria shall be specified.

Module tests are typically done on single modules or on logical groups of modules (sometimes referred to by vendors as "Unit testing" or "Functional testing") that will test how well the module or group of modules performs their collective function(s).

Quality Class N Software

The depth of module testing on Quality Class N software shall be determined by the individual responsible for the test. The testing shall be performed to, at a minimum:

1. debug code,
2. verify that each software module satisfies the functionality allocated to it by the SDS, and
3. test logic paths, boundary conditions, and error paths.

Summary of test activities shall be prepared by the Module Tester and included in the Module Test Report.

Quality Class Q Software

For Quality Class Q software, testing shall follow the program outlined in the Software Integration Plan (see Section 5.2.7).

Module tests shall evaluate:

1. the software module interface,
2. the local data structures,
3. normal processing paths,
4. error paths, and
5. boundary conditions.

Testing shall be conducted (i.e., test cases shall be generated) for input data that is typical, at design limits, and beyond design limits. Logic accuracy and, for time critical software modules, processing time shall be tested. All module tests shall employ "white box" testing techniques. It is not necessary to develop software module test procedures, but each test step (i.e., test execution) must be documented. Module test approach, features, and test results shall be documented in individual Module Test Data Sheets and included as part of the Module Test Report. All software modules in the software package shall be addressed. Module testing shall be re-performed on all modified software modules (i.e., software modules that have been tested, and the test records filed in the Module Test Report).

5.4.2.1 Module Test Reports

The Module Test Reports shall document the results of code reviews (see Subsection 5.4.1.2) and Module Testing (see Section 5.4.2).

The Module Test Report is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively. The Module Test Report is part of the baseline review for this phase.

Quality Class N Software

For Quality Class N software, the Module Test Report shall include a summary of the test activities, such as test preparation, test execution, and test results.

Quality Class Q Software

For Quality Class Q software, the Module Test Report shall describe the extent of testing performed, including the steps involved and the test results obtained. The test results shall be described in sufficient detail to allow for a review of testing adequacy. The Module Test Report shall contain the complete set of individual Module Test Data Sheets and Code Review Data Sheets. Records shall be retained for all software modules so that the as-tested software is fully documented. The Module Test Report shall also describe the extent of re-review performed on the source code when modification of a software module becomes necessary during test. The Module Test Report shall identify the software modules having safety-related application and shall provide an evaluation of the adequacy of the testing performed.

5.4.3 Safety Analysis of Software Coding

Software safety issues must be addressed at each phase of the software development for Quality Class Q software. For the Implementation Phase, the software source code modules and the software Module Test Report track the implementation and test of the software safety requirements. Each software module shall identify its safety-related usage. The Module Test Report identifies all software modules having safety-related application, and provides an evaluation of the adequacy of the testing. The Software Safety Plan [4.1.2(4)] outlines the methods and procedures to be used in assuring that the design as described in these documents adequately address safety.

5.4.4 Implementation Baseline Review Record

The BRT shall perform an Implementation Baseline Review on the documents described above. This review is conducted to ensure that all tasks required for the Implementation Phase, including the required verification reviews and testing, have been completed successfully. The BRT Task Lead shall prepare the Implementation Baseline Review Record to provide a record of the review results, and, if necessary, document all non-conformances identified during the baseline review process. The Implementation Baseline Review Record shall be filed in the software project DRF {see SCMP [4.1.2(1)]}.

5.5 Integration Phase

The purpose of the Integration Phase is to:

- verify by testing that the software/hardware interface requirements are met, and
- prove the functional correctness of software packages installed in each instrument (through functional integration testing).

Input Documents:

- HSS,
- SRS,
- IPS,
- External Data Communication Protocol Specification
- User's Manual
- SMP,
- SCMP,
- SVVP,
- SSP,
- SIntP.

Output documents, verification documents and verification methods:

Table 5 Integration Phase Output Documents

Output Documents	Verification	Method
1. Integration and Installation Test Report, including Errors Checklist and system build description	Software Integration Plan HSS and SRS for software-based systems, Software Integration Plan, IPS and User's Manual for microprocessor-based systems.	Design Verification and Traceability Analysis
2. if applicable, Support Software/Tool and its documentation package	Performed according to the instructions in SMP, SCMP, SVVP and SSP	Design Verification
3. Integration Baseline Review Record		None

Quality goals:

- Meet the functional requirements as outlined in the HSS and SRS for software-based systems, or the IPS and User's Manual for microprocessor-based systems.
- All verifications are performed in accordance with the SVVP
- Software safety analysis is performed on all outputs, documented during the verification, and in accordance with the SSP.

5.5.1 Integration Testing

The Software Integration Plan shall identify the elements to consider for all integration testing activities. These elements shall also serve as the minimum standards by which to measure and gauge the effectiveness and quality of software at this phase. As part of individual work packages, required by the SDP, the minimum and maximum acceptance criteria for these standards shall be specified.

Integration Testing shall be performed to confirm that:

- the integrated software modules perform as intended,
- faults and failures, and the associated the error handling and restoration processes work as intended,
- all operating modes work as intended
- the software, installed in the prototype or target hardware, performs as intended, and
- the interfacing entities operate as specified in the HSS and SRS for software-based systems, and in the IPS and User's Manual for microprocessor-based systems .

Integration Testing shall be designed to uncover errors that result from:

- incorrect interface (hardware and software) implementation,
- contention caused by improper global data access,
- file and data structure violations, and
- inappropriate or incorrect error-handling, including restoration from faults and failure conditions.

Software integration testing shall be performed in accordance with procedures outlined in the Software Integration Plan (see Section 5.2.7). The scope of the integration testing shall include testing of inter-module logic paths. For Quality Class Q software, functional integration testing

shall evaluate each function over its entire process range, including boundary conditions and timing considerations. Integration testing employs a combination of "white box" and "black box" testing techniques and shall utilize both the physical system hardware and software emulation tools to test internal and interface aspects of the design. As part of integration testing, detailed testing shall be performed on the interfaces of all software modules having safety-related application (i.e., module-to-module interface and module-to-hardware interfaces). Test cases, approaches, features, and results shall be documented in individual Integration Test Data Sheets and included as part of the Integration and Installation Test Report.

5.5.2 Integration and Installation Test Report

The Integration and Installation Test Reports shall document the results of integration activities.

The Integration and Installation Test Report is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

Quality Class N Software

The Integration and Installation Test Report shall include a summary of test activities, such as test preparation, test execution, and test results.

Quality Class Q Software

The Integration and Installation Test Report shall, at a minimum, contain:

1. A description of the integration testing performed, including test objectives, test equipment, software installation, input stimulus, etc.
2. Document the results, including any discrepancies found during integration test
3. Identify all source code, libraries, or any other file used to build the item subject to test, by name, location revision, and date.

An Errors Checklist shall be included to track the errors encountered during integration testing. The following classes of software errors shall be defined and tracked:

1. Data reference errors - errors that occur when data items are referenced improperly,
2. Data declaration errors - errors resulting from conflicts between intended and actual usage,
3. Computation errors - errors resulting from improper analysis or computational precision,
4. Comparison errors - errors resulting from improper or imprecise condition expressions,

5. Control flow errors - errors resulting from incorrect branching targets,
6. Interface errors - errors resulting from improper passage of data between software modules, and
7. Input/Output errors - errors resulting from incorrect data formats or invalid interface specification.

The Integration and Installation Test Report shall provide an evaluation of the testing performed on the interfaces of the software modules having safety-related application.

As part of the verification of the Integration and Installation Test Report, traceability shall be performed for Quality Class Q software to show how the hardware-software integration testing adequately tests the system requirements as outlined in the HSS and SRS for software-based systems, and in the IPS and User's Manual for microprocessor-based systems.

5.5.3 Safety Analysis of Integration Test

Software safety issues must be addressed at each phase of the software development for Quality Class Q software. For the Integration Phase, the software Integration and Installation Test Report documents the testing performed on the safety-related software modules as they are integrated with the hardware platform. The Software Safety Plan [4.1.2(4)] outlines the methods and procedures to be used in assuring that the design as described in these documents adequately address safety.

5.5.4 Integration Baseline Review Record

The BRT shall perform an Integration Baseline Review on the documents described above. This review is conducted to ensure that all tasks required for the Integration Phase have been completed, and the test results verified. The Integration Baseline Review Record shall be prepared by the BRT Task Lead to provide a record of the review results of the items that comprise the Integration Baseline Review, and, if necessary, document all non-conformances identified during the baseline review process. The Integration Baseline Review Record shall be filed in the software project DRF {see SCMP [4.1.2(1)]}.

5.6 Validation Phase

The purpose of the Validation Phase is to:

- demonstrate that the software-based product is operational and conforms to all functional and performance requirements contained in the HSS.

Input Documents:

- Validation Test Procedure and Test Cases Specification,
- integrated software package,

- SMP,
- SCMP,
- SVVP,
- SSP,
- SIntP.

Output documents, verification documents and verification methods:

Table 6 Validation Phase Output Documents

Output Documents	Verification	Method
1. Validation Test Report	HSS, LDs, P&IDs, Software Integration Plan, SRSs for software-based system, SBD and IPS, for microprocessor-based system	Design Verification and Traceability Analysis
2. Build Release Description	Performed according to the instructions in the SVVP and SSP.	Design Verification
3. If applicable, Support Software/Tool and its documentation package	Performed according to the instructions in SMP, SCMP, SVVP and	Design Verification
4. Validation Test Baseline Review Record		None

Quality goals:

- Meet the functional requirements as outlined in the HSS and other applicable documents.

5.6.1 Validation Testing

The purpose of validation testing is to verify that the completed product meets the requirements outlined in the HSS. The system validation test is guided by the Software Integration Plan (see Section 5.2.7) based on the requirements contained therein. The Validation Test Procedures and Test Cases Specification (see Section 5.3.3) contain steps that demonstrate that all functional and performance characteristics of the software-based product are operational, and that design requirements outlined in the HSS and the User's Manuals have been met. The validation testing activity shall follow the steps outlined in the Validation Test Procedures and Test Cases Specification to successful completion.

5.6.2 Validation Test Report

The Validation Test Report shall document the results of the Validation Test, including test anomalies and their resolutions. The results of the Validation Test shall be evaluated to verify that the system requirements have been met. The Validation Test Report is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

5.6.3 Build Release Description

The RE shall generate a "system build" description to document the final validated and released software. This document shall contain all information required to describe the "build" parameters of the software, such that a duplicate version of the object code can be recreated. The Software Integration Plan defines the requirements needed to develop the system build description. The Build Release Description is a configurable item, and is subject to verification and configuration management as specified in the SVVP [4.1.2(2)], the SSP [4.1.2(4)], and SCMP [4.1.2(1)] respectively.

5.6.4 Safety Analysis of Validation Test

Software safety issues must be addressed at each phase of the software development for Quality Class Q software. For the Validation Phase, the Validation Test Report documents the testing performed on the entire software package to demonstrate that the safety-related functions, identified in the design input requirements, are operational. The Software Safety Plan [4.1.2(5)] outlines the methods and procedures to be used in assuring that the design as described in these documents adequately address safety.

5.6.5 Validation Baseline Review Record

The BRT shall perform a Validation Test Baseline Review on the output documents described for this phase. This review ensures completion of all tasks required for the Validation Phase, including the required verification reviews. The BRT Task Lead shall prepare the Validation Test Baseline Review Record to provide a record of the review results of the items that comprise the Validation Test Baseline Review, and, if necessary, document all non-conformances identified during the baseline review process. The Validation Test Baseline Review Record shall be filed in the software project DRF {see SCMP [4.1.2(1)]}.

5.7 Installation Phase

The purpose of the Installation Phase is to:

- Prepare the software-based product installation, operation, maintenance, and training manuals.

Input Documents:

- HSS
- IPS or SRS

- SDS
- User Manual
- SMP,
- SCMP,
- SVVP,
- SIP
- SOMP
- STrngP
- SSP,

Output documents, verification documents and verification methods:

Table 7 Installation Phase Output Documents

Output Documents	Verification	Method
1. Software Installation Procedure	Build Release Description	Design Verification
2. Software Installation Report	Software Installation Procedure	Design Verification
3. Installation Configuration Tables	HSS, IPS, SRS, SDS, User Manual	Design Verification
4. Operations Manual	HSS, IPS, SRS, SDS, User Manual, SIP, SOMP	Design Review
5. Maintenance Manual	HSS, IPS, SRS, SDS, User Manual, SIP, SOMP	Design Review
6. Training Manual	Operation & Maintenance Manual, STrngP	Design Review

Quality goals:

- Transcribe the requirements and design information described in the design documentation to Operations, Maintenance, and Training Manuals.
- Successful installation of non-microprocessor based software systems.

5.7.1 Software Installation Report

An installation report for each package or system shall be produced upon the completion of the installation effort and shall report as a minimum;

1. Summary of installation activities

2. All anomalies discovered during installation, which shall be provided to the developer and resolved prior to placing the software into operation.

3. Any associated data sheets generated during the installation,

4. Installation test summary,

5. Any user configurable parameter values,

6. Indication of customer approval and acceptance of the installation activities.

For software-based systems, part of the system acceptance testing shall be the successful rebuild and installation of the rebuilt software.

5.7.2 Installation Configuration Tables

When applicable, configuration tables shall be developed for each software package. Each user configurable function shall be defined, along with each configurable mode, which includes the function, safety, and security of the overall application.

For software based systems with modifiable configurations, part of the system acceptance testing shall be the successful initialization and re-installation of the configuration data.

5.7.3 Operations Manuals

Operations Manuals shall be developed in accordance with EOP 70-5.00 Operation and Maintenance Instruction Manuals.

5.7.4 Maintenance Manuals

Maintenance Manuals shall be developed in accordance with EOP 70-5.00 Operation and Maintenance Instruction Manuals.

5.7.5 Training Manuals

Training Manuals shall be complete, which requires that:

1. All actions available to the operator are described fully for all operating modes, including error recovery.
2. Operator actions shall be specified in terms of inputs supplied by users and equipment, actions initiated by the operation, and responses to the user.
3. The Training Manuals should describe the operational environment within which the software will operate, including precautions and limitations that must be observed during operations to avoid exposing personnel or the plant to hazards.
4. All variables in the physical environment that the software must monitor and control should be fully described. User interfaces should be fully described for each category of user.

5.8 Operations and Maintenance Phase

The objective of the Operations and Maintenance Phase is to:

- provide a controlled path through the design process (operation) that may be invoked when software modification is required.

Input Documents:

- I&C Software Problem Report {see SCMP [4.1.2(1)]},
- Engineering Change Notice (ECN).

Output documents, verification documents, and verification methods:

Table 8 Operation & Maintenance Phase Output Documents

Output Documents	Verification	Method
1. ECN(s)	I&C Software Problem Report	Design Verification
2. Revised/impacted documentation[6]	Hierarchy documentation of the Revised/impacted documentation	Design Verification
3. Revised Source and Executable Code	Revised SDS and/or Internal Communication Protocol Specification	Design Verification
4. Module Test Report[7]	Revised SDS	Design Verification and Traceability Analysis
5. Integration and Installation Test Report[6]	For software-based system, revised HSS and SRS, For microprocessor-based system, revised SBD and IPS	Design Verification and Traceability Analysis
6. Validation Test Report[6]	HSS, Software Integration Plan, for software-based system, SRSs, for microprocessor-based system, SBD and IPS	Design Verification and Traceability Analysis
7. Revised Baseline Review Records		None

Quality goals:

- Changes designed and implemented using the complete process.
- Identified error resolved and no new errors induced.

[6] The revised documentation may include an abbreviated version of the Software Validation Test Procedure to test the modified sections of the software package.

[7] May be a supplemental or "delta" report.

5.8.1 Maintenance Objectives and Scope

The Maintenance Phase begins with the completion of the Validation Test Baseline Review (see Section 5.6.4), and effects changes to previously validated (baselined) software. Software engineering change control is the process used to control, authorize, and implement changes to engineering controlled documents to:

1. assure that the total impact of a change is considered before a change is approved,
2. assure re-evaluation of the software safety analysis and the documentation of any re-evaluation performed,
3. assure that all affected documents are identified and changed as part of the change package,
4. provide authority for a change, and identify all organizations responsible for the engineering inputs {see SCMP [4.1.2(1)]},
5. provide accurate and traceable records of a change, and
6. assure scheduled implementation.

Changes made to the software-based product, including the software, shall be handled in accordance with configuration change control process defined in the SCMP [4.1.2(1)].

I&C Software Life Cycle Phases:	Planning	Requirements	Design	Implementation	Integration	Validation	Installation	Operation & Maintenance
SRP Software Life Cycle Phase	Planning	Requirements	Design	Implementation	Integration	Validation	Installation	Operation & Maintenance

Figure 1 I&C Software Life Cycle Phases vs. SRP Software Life Cycle Phases

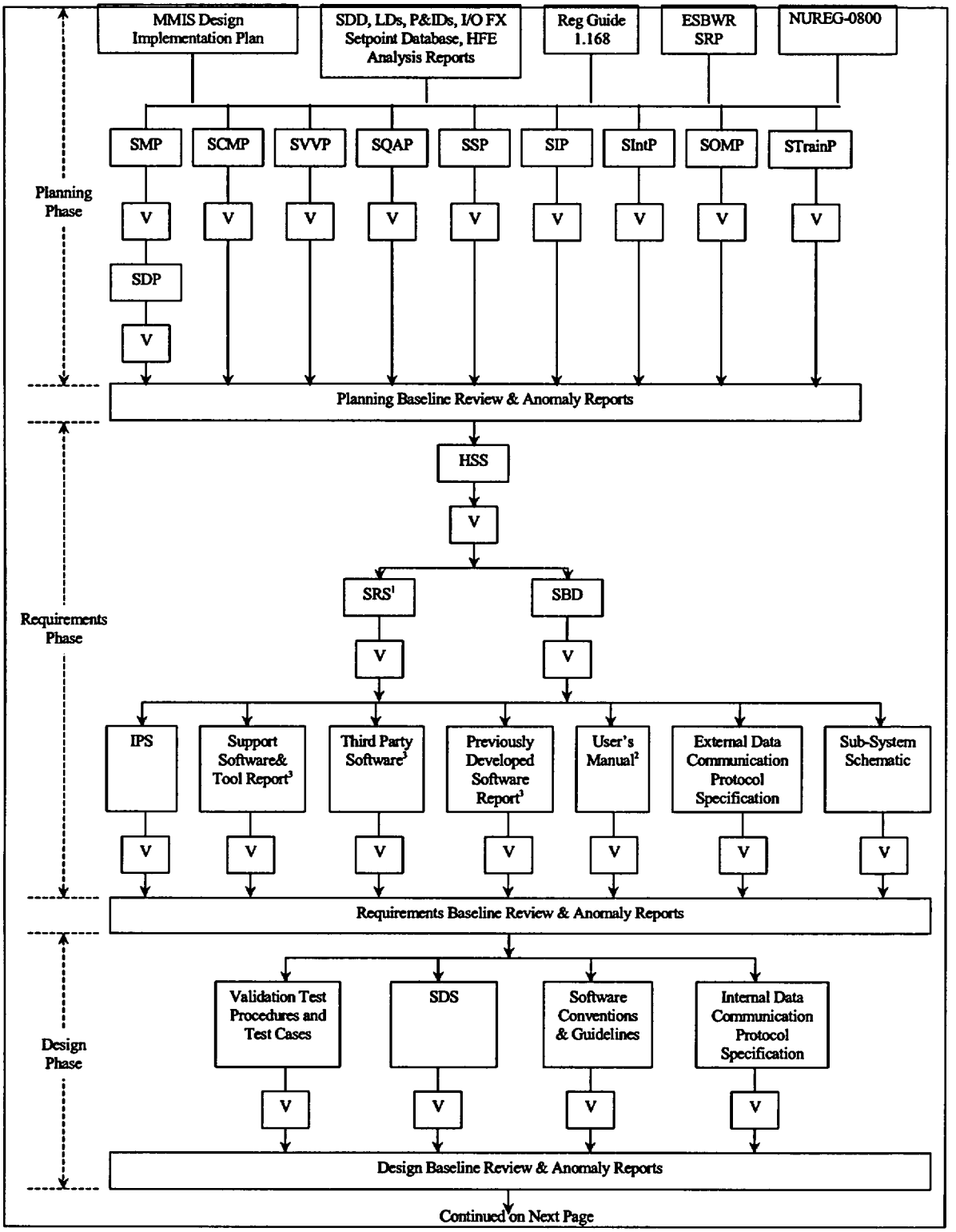
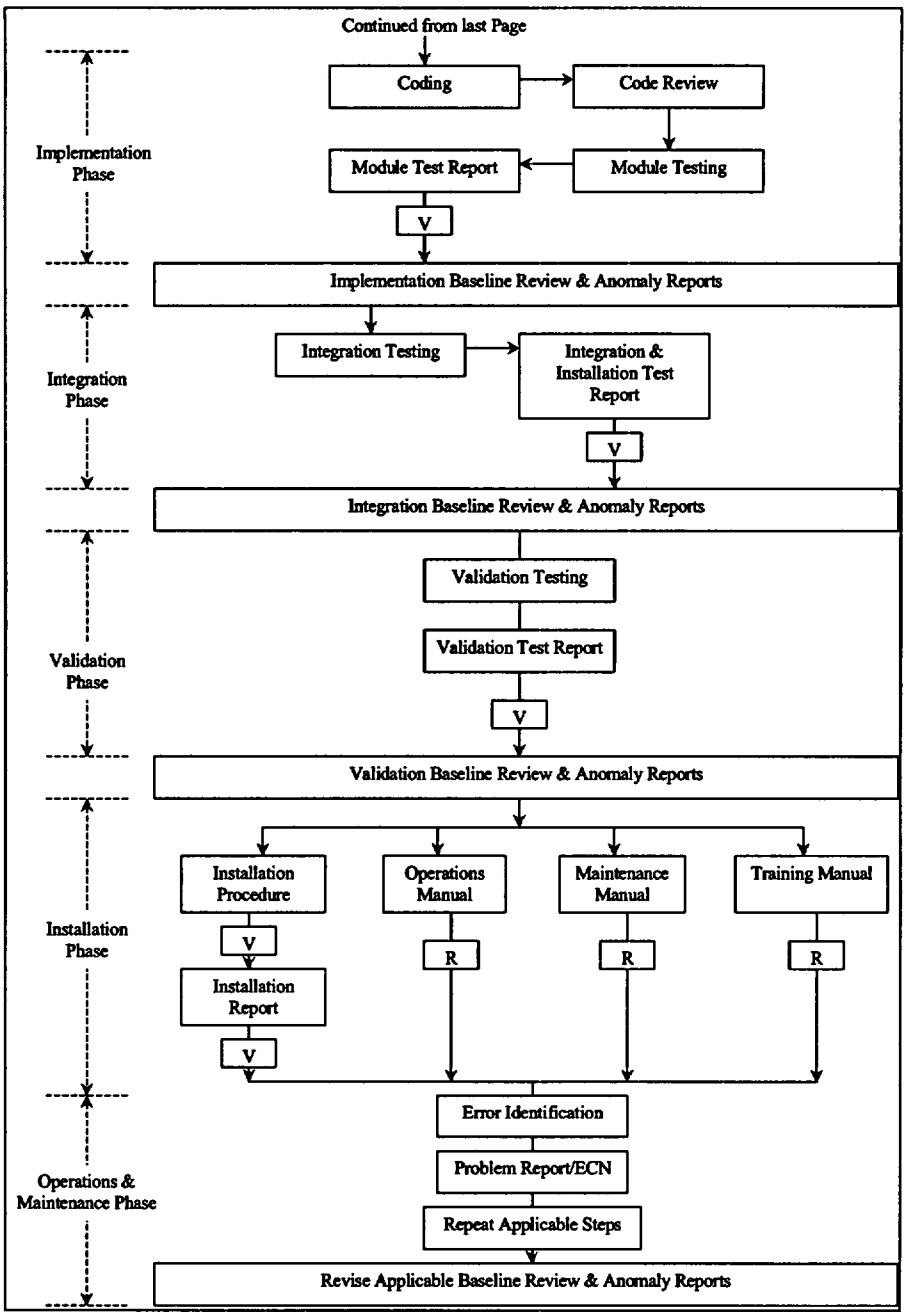


Figure 2 I&C Software Life Cycle Phases and Hierarchy of Documentation Development.



If App licable
1 - SR S is not required for Microprocessor Based Systems
2 - Development of the User's manual may develop through l ater phases
3 - These do cuments are developed in th e appropriate step
4 – No t App licable for microprocessor based sys tems

Figure 2 I&C Software Life Cycle Phases and Hierarchy of Documentation Development (Continued)

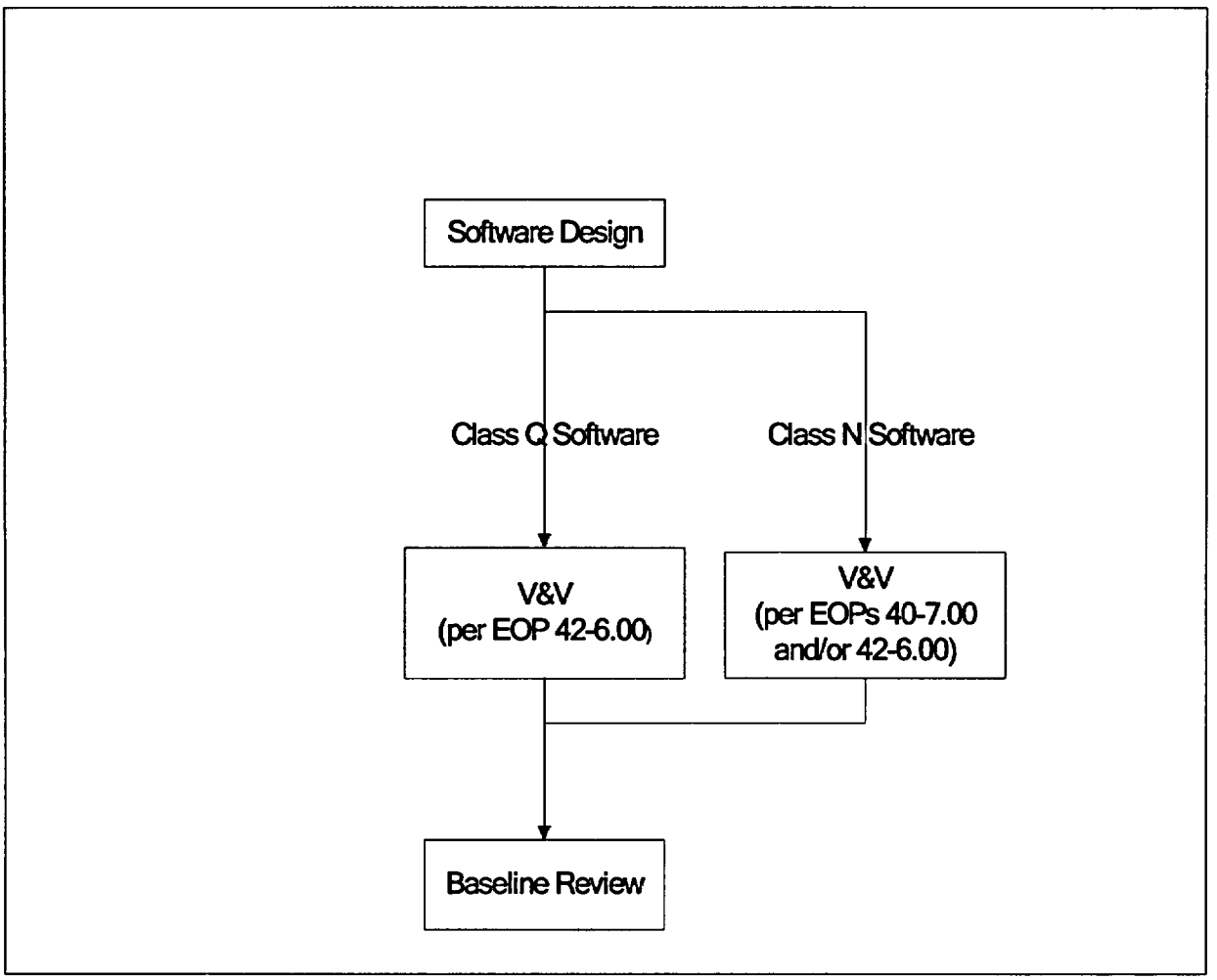


Figure 3 Verification and Validation Process

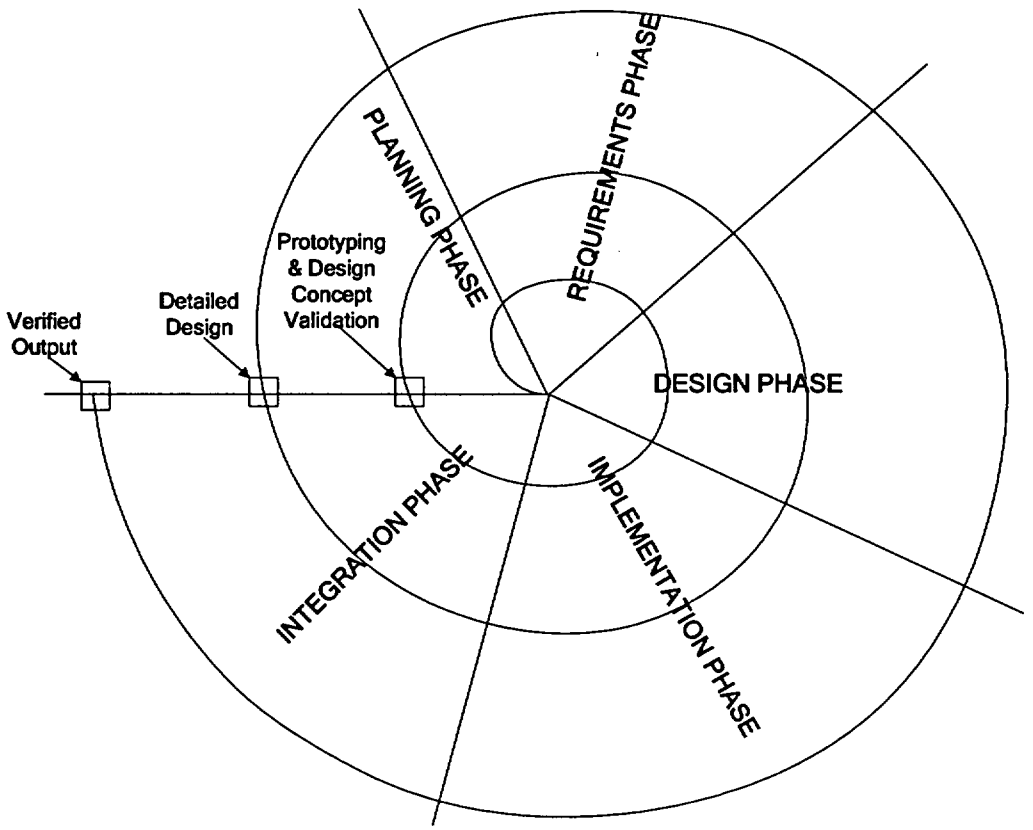


Figure 4 Evolutionary Life-Cycle Model

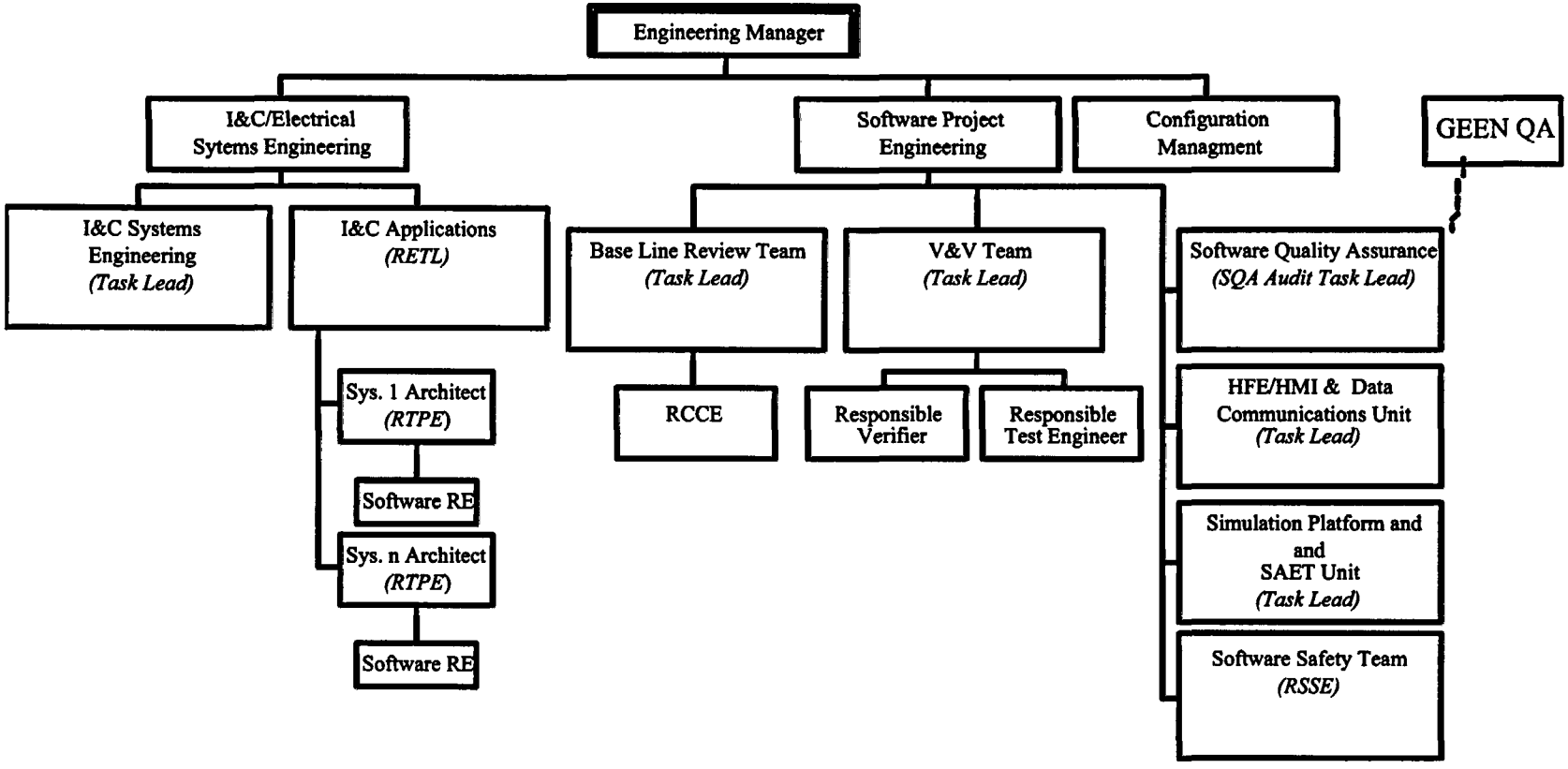


Figure 5 I&C Organization

67

Appendix A: Document Formats

A.1 Hardware/Software Specification (HSS)

The following is one acceptable format for the Hardware/Software Specification (HSS). Sub-system definitions may be added as required. Changes to the format must be approved by the RTPE.

1 Scope

2 Applicable Documents

- 2.1 Supporting and Supplemental Documents
  - 2.1.1 Supporting Documents
  - 2.1.2 Supplemental Documents
- 2.2 Codes and Standards
  - 2.2.1 American Society of Mechanical Engineers (ASME) Codes
  - 2.2.2 Institute of Electrical and Electronic Engineers (IEEE) Standards
- 2.3 Regulation and Regulatory Requirements
  - 2.3.1 U.S. Nuclear Regulatory Commission Regulations

3 System Overview

- 3.1 System Description
- 3.2 System Classification
- 3.3 System Architecture
- 3.4 Interfaces to Other Systems

(Subsections may be arranged or added to clarify system specific requirements.)

4 Hardware/Software Requirements

- 4.1 System Functional Requirements
- 4.2 Hardware Requirements
- 4.3 Software Requirements
- 4.4 Design Constraints

(Subsections may be arranged or added to clarify system specific requirements.)

5 V&V Requirements

- 5.1 System Testing Requirements
- 5.2 Integrated Test Requirements (as applicable)

6 Appendix A

- 6.1 Definitions and Acronyms

A.2 Software Requirements Specification (SRS)

The following is one acceptable format for the Software Requirements Specification (SRS). Changes to the format must be approved by the RTPE.

1 Introduction
- 1.1 Purpose
- 1.2 Scope
- 1.3 Definitions, Acronyms and Abbreviations

2 Applicable Documents
- 2.1 Supporting and Supplemental Documents
- 2.1.1 Supporting Documents
- 2.1.2 Supplemental Documents
- 2.2 Codes and Standards
- 2.3 References

3 Description
- 3.1 Subsystem Perspective
- 3.2 Subsystem Functions
- 3.3 Interfaces
  - 3.3.1 Subsystem Interfaces
  - 3.3.2 Hardware Interfaces
  - 3.3.3 Software Interfaces
  - 3.3.4 User Interfaces
  - 3.3.5 Communication (I/O) Interfaces
- 3.4 Operations
- 3.5 User Characteristics

4 Specific Requirements
- 4.1 Subsystem Features
  - 4.1.1 Feature 1
  - 4.1.2 Feature 2
  - 4.1.3 … through Feature *n*
- 4.2 User Displays
- 4.3 Reports
- 4.4 Performances Requirements
- 4.5 Site Installation/System Initialization Requirements
- 4.6 Security
- 4.7 Redundancy
- 4.8 Alarm/Error Presentation
- 4.9 Design Constraints
- 4.10 Assumptions and Dependencies
- 4.11 System Engineering Attributes
  - 4.11.1 Reliability
  - 4.11.2 Availability

A.3 Instrument Performance Specification (IPS)

The following is one acceptable format for the Instrument Performance Specification (IPS). Changes to the format must be approved by the RTPE.

1 Scope
2 Applicable Documents
  2.1 Supporting and Supplemental Documents
    2.1.1 Supporting Documents
    2.1.2 Supplemental Documents
  2.2 Codes and Standards
    2.2.1 American Society of Mechanical Engineers (ASME) Codes
    2.2.2 Institute of Electrical and Electronic Engineers (IEEE) Standards
  … etc.
  2.3 Regulation and Regulatory Requirements
    2.3.1 U.S. Nuclear Regulatory Commission Regulations
  … etc.
3 Description
  3.1 General Description
  3.2 [*System name*] Performance Description
  3.3 [*Instrument name*] Performance Description
    3.3.1 User Interface
    3.3.2 Instrument Modes
    3.3.3 Instrument Bypass
    3.3.4 Instrument Calibration
    3.3.5 Response Times
    3.3.6 Communications
    3.3.7 Physical Description
  3.4 Functional Description
    3.4.1 General
    3.4.2 [*Module name*] Description
    (Subsections may be added to clarify requirements.)
  3.5 Instrument Safety Classification
4 Software Controlled Functions
  4.1 Software Overview
    4.1.1 Computer 1 (embedded processor 1)
  … Computer n (embedded processor n)
  4.2 Hardware/Software Initialization
  4.3 Periodic Functions
    4.3.1 Instrument Functions
    4.3.2 Input Signal Processing
    4.3.3 Inputs from Communications Links
    4.3.4 Process Calculations

Appendix B: Requirement Standards

Application	Requirement Standards
1. Quality Class Q Application or Systems	Reg. Guide 1.152, Criteria for Digital Computers in Safety Systems of Nuclear Power Plants [4.2.4(1)]
2. Quality Class N Application or Systems	ISO-9001-2000, Quality management and quality assurance standards - Guidelines for the application of ISO 9001-2000 to the development, supply and maintenance of software [4.2.3(1)]