LA-UR- 98-4611

APT-PPO-0013, Rev 0
June 1998

Accelerator Production of Tritium

Software Management Plan

Los Alamos National Laboratory, Burns and Roe Enterprises Inc., General Atomics,
Westinghouse Savannah River Company, Brookhaven National Laboratory,
Lawrence Livermore National Laboratory, Sandia National Laboratories

DISCLAIMER

Accelerator Production of Tritium

PPO 2175 (3/98)

RELEASE SUMMARY

DOC CODE	DOCUMENT NUMBER	REVISION
PLN	APT-PPO-0013	0

TITLE: Software Management Plan

CM ACCEPTANCE/ DATE	REV	PREPARED BY	APPROVAL(S)		REVISION DESCRIPTION/ WBS NO.
			RESOURCE/ SUPPORT	PROJECT	
6 RELEASED JUN 2 2 1998	0	B. Olsen	L. Parme R. Baumbach	C. Rodriguez	Initial Release 1.17

Accelerator Production of Tritium

Document No.	APT-PPO-0013	Document Title	Software Management Plan

LIST OF EFFECTIVE PAGES

Page Number	Number of Pages	Revision
i through vi	6	0
1 through 18	18	0
A-1 through A-3	3	0
TOTAL NUMBER OF PAGES:	27	

PPO 4053 (6/98)

Table of Contents

1. Purpose and Scope 1
2. Requirements Documents 2
3. Software Management 2
3.1 Team Member Organizations 2
3.2 PPO Responsibilities 3
4. Software Engineering 4
4.1 Computing Environment 4
4.2 New Software Development 4
4.3 Documentation Requirements 6
5. Standards, Practices and Conventions 8
5.1 Design Standards 8
5.2 Coding Standards 8
6. Verification and Validation Activities 9
6.1 Verification Approach 9
6.2 Validation Approach 10
7. Software Configuration Management 11
7.1 Configuration Identification 11
7.2 Configuration Control 11
7.3 Configuration Status Accounting 12
8. Problem/Change Reporting and Corrective Action 12
9. Software Quality Assurance Activities 13
10. Code Control – The Controlled Program Library 13
11. Software Procurement 14
12. Existing Software 14
12.1 Use of Existing Software 14
12.2 Documentation Requirements for Existing Software 15
12.3 Verification of Existing Software 17
12.4 Validation of Existing Software 17
12.5 Configuration Management of Existing Software 18
Appendix A - Definitions A-1

List of Tables

None

List of Figures

Figure 1. APT Design Team 3
Figure 2. PPO Groups Impacted by *Software Management Plan* 3
Figure 3. Software Life Cycle Phases. 5
Figure 4. Verification and Validation Process. 9

Acronyms

ANS	American Nuclear Society
ANSI	American National Standards Institute
APT	Accelerator Production of Tritium
ASME	American Society of Mechanical Engineers
CAD	Computer Aided Design
CFR	Code of Federal Regulations
CAE	Computer Aided Engineering
DOE	Department of Energy
GFI	Government Furnished Information
OPO	Operations Project Office
PDO	Project Director's Office
PPO	Plant Project Office
QA	Quality Assurance
QAPP	Quality Assurance Program Plan
QAR	Quality Assurance Requirements
SCCB	Software Configuration Control Board
SSC	Structures, Systems, and Components
TPO	Technology Project Office
V&V	Verification and Validation

Software Management Plan

1. Purpose and Scope

This *Software Management Plan* documents the process used to manage computer software-related activities pertaining to the Accelerator Production of Tritium (APT) design. These activities include development, acquisition, leasing, modification, configuration management, and controlled use of Engineering & Scientific software and Real Time software on the APT project. The management of software by the APT design and development team members and others, including suppliers, performing tasks during the preliminary and final design, construction, startup and testing of the APT plant shall be compatible with the approach developed in this Plan.

The requirements of this Plan are mandatory for all software used for design, design analysis, process control and operations involving Safety Class and Safety Significant structures, systems, and components (SSCs).

The requirements of this Plan apply whether the software is developed in-house, leased, licensed, procured, obtained from another organization, obtained from a commercial vendor, or otherwise acquired. Specific software categories considered by this Plan include:

- Engineering & Scientific computer codes: software used to define, analyze, evaluate or verify APT functional and physical characteristics
- Plant operating and process control & monitoring software (Real Time software)
- Plant protection software (Real Time software)
- Simulator software (Real Time software)
- Database and spreadsheet application templates for multiple use used to generate input data to engineering analysis and design. The general purpose utility software itself is excluded from this Plan.

Types of software not covered by this Plan includes:

- Applications of Computer Aided Design (CAD), Computer Aided Engineering (CAE) software which are verified by the normal review and approval of engineering documents and are not within the scope of this Plan
- Single use engineering software approved as part of a calculation and not intended for reuse
- Software used for administrative functions

This *Software Management Plan* will also serve as a plan for software quality assurance by using the requirements herein as a measure for quality control.

The *Software Management Plan* is divided into the following twelve sections:

Section 1 introduces the Plan.

Section 2 lists sources of requirements.

Section 3 describes the software management approach, including project organization and responsibilities with respect to software activities and interfaces with other team members and subcontractors.

Section 4 describes the software engineering approach, including the software engineering environment, software development techniques, and software documentation requirements.

Section 5 describes standards, practices, and conventions for software activities.

Section 6 describes software verification and validation.

Section 7 describes the software configuration management approach, including configuration identification, configuration control, and configuration status accounting.

Section 8 describes the software problem/change reporting and corrective action process.

Section 9 describes software quality assurance activities to monitor and ensure compliance with requirements stated in this Plan.

Section 10 describes the controlled program library, including computer program access control, computer program execution auditing, and document retrieval.

Section 11 describes the software procurement approach including documentation requirements and verification and validation requirements for procured software.

Section 12 describes use of existing software, including documentation requirements, verification and validation activities, and configuration management.

A definition of software terms is provided in Appendix A.

2. Requirements Documents

The following documents were used in developing this plan:

ANSI/ANS-10.4-1987 – *Guidelines for the Verification and Validation of Scientific and Engineering Computer Programs for the Nuclear Industry*, American National Standards Institute, Inc.

ASME NQA-1-1994 – *Quality Assurance Program Requirements for Nuclear Facilities*, American Society of Mechanical Engineers.

DOE Order 5700.6C – *Quality Assurance*, US Department of Energy, Washington DC.

10 CFR 830.120 - Quality Assurance Requirements, US Government, 1996.

3. Software Management

3.1 Team Member Organizations

The APT design and development team, shown in **Figure 1**, includes staff from the Project Director's Office (PDO), Technical Project Office (TPO), Plant Project Office (PPO), and Operation Project Office (OPO).

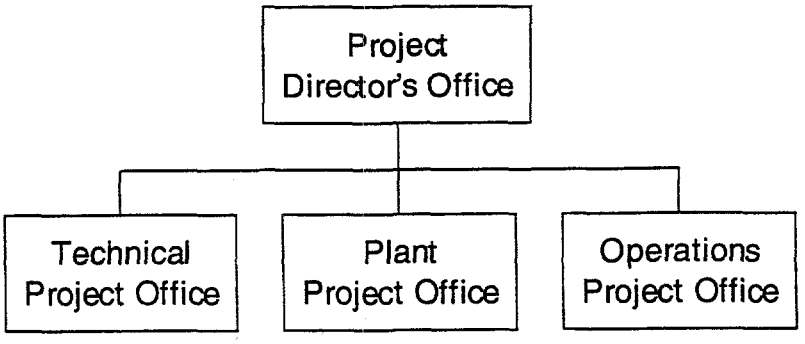

Figure 1. APT Design Team

3.2 PPO Responsibilities

Figure 2 shows the relationship of the PPO groups impacted by the *Software Management Plan*.

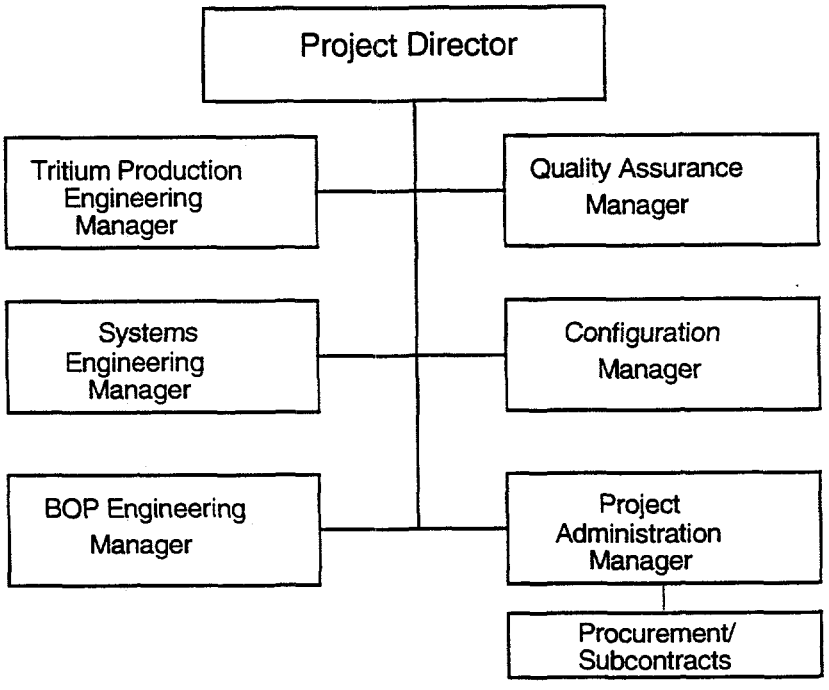

Figure 2. PPO Groups Impacted by the *Software Management Plan*

The Engineering groups are responsible for the software engineering, verification and validation (V&V) activities, and Software Configuration Control Board (SCCB) activities described in this Plan.

The Configuration Management group has overall responsibility software configuration control, the controlled program library, maintaining a master list of all approved PPO computer codes, and administration of the software problem/change reporting and corrective action program.

The Quality Assurance (QA) group is responsible for verifying that the processes throughout the software life cycle described in this Plan are carried out.

The Project Administration group has responsibility for software procurement.

4. Software Engineering

4.1 Computing Environment

The provisions of this Plan apply to the software categories described in Section 1, independent of any hardware and software environment, that is, they apply to all operating systems and hardware platforms chosen by the APT project team members.

4.2 New Software Development

4.2.1 Basis for Life Cycle Approach

Development of new software and modifications of existing software should proceed in a planned, traceable, and orderly manner, as shown by the generic software life cycle model described in Section 4.2.2. Division of the development process into phases, with documentation and verification performed in each phase, provides logic and process control, aiding in early detection of problems and errors. The multiple-phase life cycle model is especially suited for software development projects where costs of rework could be significant.

The life cycle phases and documentation can be combined, depending on the significance, criticality, and complexity of the software. The developer and the sponsor of the software jointly determine the actual number of development phases and the documentation that will be used in a particular software development project before the project begins. This determination should take into account the cost of the documentation and reviews, as well as the risk of proceeding without interim verification. If the planned development process is different from the generic software life cycle described below, the process used should be documented.

4.2.2 Generic Software Life Cycle

Software developed or modified for uses indicated in Section 1 is controlled through use of the software life cycle phases, shown in Figure 3.

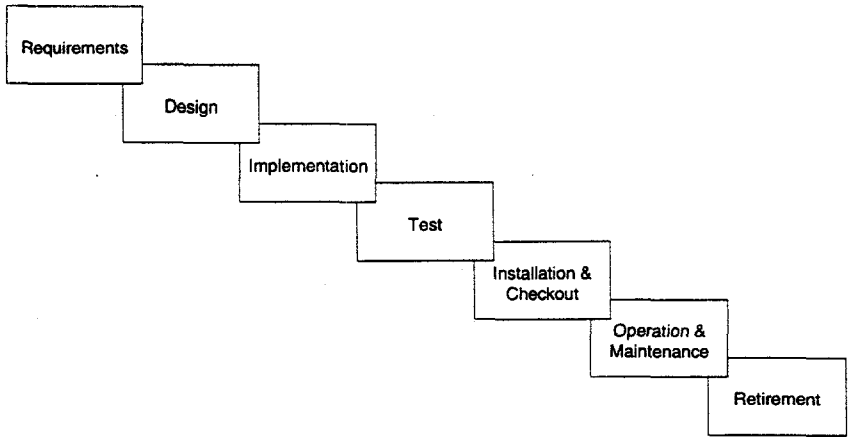

Figure 3. Software Life Cycle Phases

The activities and documentation prepared during each phase are as follows:

- Requirements Phase – This phase defines the requirements that the software must satisfy, including functionality, performance, accuracy, design constraints, attributes, and external interfaces. During this phase, the *Software Requirements Specification* and *Software Verification and Validation Plan* are prepared. On completion of the *Software Verification and Validation Plan*, a software requirements review is performed. After approval, the documents are placed under formal configuration control.

- Design Phase – This phase involves design of the software and preparation of planning documentation for ensuring that the final product performs as desired. During this phase, a *Software Design Description* is prepared. When complete, the *Software Design Description* undergoes a design review. After approval, the document is placed under formal configuration control.

- Implementation Phase – During this phase the design is translated into computer code. A *User's Manual* is developed . A test readiness review is performed upon completion of computer coding and documentation. After approval, all documentation is placed under formal configuration control.

- Test Phase – During this phase the software is tested in accordance with the *Software Verification and Validation Plan* to detect and correct failures. Depending on the complexity of the computer program being tested, testing may range from a single test of the completed computer program to a series of tests performed first on individually developed modules, followed by an overall computer program test. For Real Time software this may involve integration testing with simulator hardware to meet parametric accuracy and response time requirements. All testing should cover the multitude of test cases and ranges of parameters expected for production runs or operation of the computer software.

  Integration testing of Real Time software may require the preparation of separate documents such as a *Test Plan* and *Test Procedure*. Depending on the magnitude of the task, these may be contained in the *Software Verification and Validation Plan*.

  During this phase, a *Software Verification and Validation Report* or a separate *Test Report* and/or *Test Evaluation Report* is prepared, and undergoes qualification

review upon completion. After approval, the document is placed under formal configuration control.

- Installation and Checkout Phase – During this phase the software becomes part of a system incorporating applicable software components, hardware, and data. The final software is integrated into the operational environment and tested in accordance with the *Software Verification and Validation Plan.* Real Time software may be finally tested during system preoperation and commissioning. This may involve the integrated testing with other real time modules (software, hardware, or network) to check out total system performance.

  Separate *Test Plan* and *Test Procedure* may be developed for preoperational checkout of Real Time software or reference made to preoperational testing procedures in the *Software Verification and Validation Plan.* The results of the testing should be included in the *Software Verification and Validation Report* or provided in separate *Test Report* and/or *Test Evaluation Report.*

  During this phase the software is entered into the controlled program library, which provides status of the software for operational use. The software is also added to the PPO list of approved computer codes.

- Operation and Maintenance Phase – During this phase the software is maintained, which may involve corrective, adaptive, and perfective modifications. This process is discussed in Section 7, Software Configuration Management and Section 8, Problem/Change Reporting and Corrective Action.

- Retirement Phase – During the retirement phase, support for a software product is terminated, and routine use of the software is prevented.

4.3 Documentation Requirements

4.3.1 Basis for Documentation Requirements

As with the life cycle phases, documents may be combined, as long as the contents of the individual documents as described in Section 4.3.2 are included in the combined document. The developer and the sponsor of the software jointly determine the actual number and contents of the documents that will be developed in a particular software development project before the project begins. If the planned number or contents of documents is different from that described in Section 4.3.2, the combination of documents is indicated in the title of the resultant combined document.

Section 4.3.2 contains generic requirements for software documentation that correspond to the generic software life cycle described in Section 4.2.2.

4.3.2 Generic Documentation Requirements

Required documents listed below are described with specific titles. Alternative titles may be used if the equivalence is documented.

Documentation for new software, and for modified portions of existing software, consists of the following:

- *Software Requirements Specification*

- *Software Verification and Validation Plan*
- *Software Design Description*
- *Software Verification and Validation Report*
- *User's Manual*

Additional documentation for Real Time software may include:

- *Test Plan*
- *Test Procedure*
- *Test Report*
- *Test Evaluation Report*

Each document has numerical document identification and conforms to format requirements for technical documents indicated elsewhere.

Software Requirements Specification – This document is prepared for each new computer program and addresses functionality, performance, range of accuracy, design constraints, attributes, and external interfaces, as applicable. This document specifies the requirements allocated to a computer program and translates the functional requirements into more detailed software requirements.

Software Verification and Validation Plan – This plan describes the tasks for verification and validation of the software, including methods used for verification and validation and acceptance criteria. The plan specifies hardware and software configurations pertinent to the software verification and validation. The plan is organized in a manner that allows traceability to both software requirements and software design.

Software Design Description – This document contains the following: a description of major components of the software design as they relate to software requirements; a technical description of the software with respect to the theoretical basis, mathematical model, control flow, data flow, control logic, and data structure; a description of allowable or prescribed ranges for inputs and outputs; and the design described in a manner that can be translated into code.

Software Verification and Validation Report – This document describes the results of each verification action performed in reference to the *Verification and Validation Plan*, including software testing. It identifies remaining deficiencies, serving as an exception report to highlight potential problem areas. This report also documents the results of validation activities and shows comparisons with numerical criteria.

Software User's Manual – This document provides a brief software design description and detailed instructions on using the completed computer program. At a minimum it includes the following: identification of subroutine, function, and variable names; user instructions; input and output specifications; input and output formats; a description of system limitations; a description of expected problems and how the user can respond; and information for obtaining user and maintenance support.

Test Plan – This document describes the approach to be followed for testing a computer system. Typical contents identify the items to be tested, test cases, test sequences, requirements and acceptance criteria, and responsibilities for the testing activities.

Test Procedure – This document provides the detailed step-by-step process to complete the test.

Test Report – This document describes the test results in detail.

Test Evaluation Report – This document expounds upon the test results from a software design perspective and documents the effect or consequences of the results of the test on the design.

5. Standards, Practices and Conventions

The following design and coding standards should be applied to any software development.

5.1 Design Standards

Software should be designed using a structured methodology that identifies the specific types of calculations or operations to be performed, input and output requirements, hardware on which the software should execute, hardware interfaces, and interfaces to the user and other software. Design constraints and acceptance criteria should be dependent on the complexity, criticality, and significance of the software being developed.

For design of control software, the hardware solution may be specified later in the development project to achieve optimum results.

Any one or a combination of the following accepted software design methods may be used: flow charts, data flow diagrams with real time extensions, data dictionaries, process specifications, entity-relationship diagrams, and state transition diagrams.

Names for processes, flows, stores, and terminators should be meaningful.

A theoretical basis should be developed to provide a mathematical model of the physical world and numerical methods to approximate the solution of the mathematical model. This theoretical basis should be contained in the *Software Design Description* (see Section 4.3.2) or be a separate document referenced by the *Software Design Description.*

5.2 Coding Standards

The following standards apply to all software developed or modified.

Each module is adequately commented to include the following: a statement of purpose that indicates the function of the module; an interface description; a discussion of pertinent data such as important variables and their use, restrictions and limitations, and other information; and a development history. Source code also contains embedded comments, which describe processing functions. Embedded comments provide extra information, rather than simply paraphrasing the source code.

6. Verification and Validation Activities

Verification of software is the process of determining whether the product of a given phase of the software development cycle fulfills the requirements imposed by the previous phase. Validation of software is the testing and evaluation of completed software to ensure compliance with its requirements. During the software life cycle, verification activities are performed in the requirements, design, and implementation phases, and validation activities are performed in the test, installation, and checkout phases. Figure 4 presents an overview of the software development cycle and the verification and validation (V&V) process.

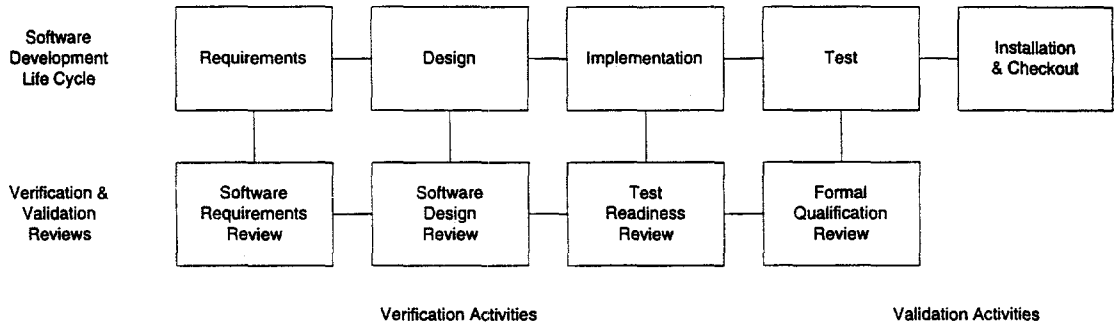

Figure 4. Verification and Validation Process

A *Software Verification and Validation Plan* is prepared during the requirements phase to describe the tasks and acceptance criteria for V&V of the software. The level of V&V is consistent with the complexity, criticality, and significance of the software. Minimally, the *Software Verification and Validation Plan* describes the following: all V&V tasks required; tools, techniques, methods, and acceptance criteria; input and output including reports; schedules; resources required; risks and assumptions; and roles and responsibilities for accomplishing the software V&V. The *Software Verification and Validation Plan* also identifies all required procedures and test documentation.

The reviews, described in Sections 6.1 and 6.2, are performed by individuals who have sufficient technical expertise to competently evaluate the documents and activities under review. Review participants do not participate in development of the area of the software that is under review.

The results of software V&V activities are formally documented in the *Software Verification and Validation Report*. This report includes the results of all reviews and tests identified in the *Software Verification and Validation Plan*; specifies hardware and software configurations pertinent to the V&V process; summarizes status of the software; describes deficiencies and their resolution; and states whether the software is ready for operational use. This report is prepared during the test phase of the software life cycle.

6.1 Verification Approach

In general, verification is accomplished by conducting reviews at various stages of the development processes, outlined in Section 4.2.2. The required reviews are as follows:

- Software Requirements Review – This review assures that the software requirements stated in the *Software Requirements Specification* are complete, consistent, technically feasible, and testable. The review also ensures that the requirements will result in feasible and usable code. Both the *Software Requirements Specification* and *Software Verification and Validation Plan* are reviewed.
- Software Design Review – The *Software Design Description* is examined for technical adequacy of the approach, to assure logic, completeness, consistency, clarity, and correctness of the software design, and to verify that the software design is traceable to its requirements.
- Test Readiness Review – This review verifies that the software is ready for normal testing. More specifically, this review assures that the coding is consistent with the design, for example, that mathematical equations are solved correctly. In addition, this review verifies that approved changes have been incorporated into the *Software Requirements Specification* and the *Software Design Description*, evaluates the *User's Manual* for completeness, consistency, and correctness, and verifies that *code* reviews have been conducted.

Upon satisfactory completion of each of the reviews, all documentation is placed under formal configuration control.

6.2 Validation Approach

In general, validation testing of the complete and integrated software is the primary method of software validation. Validation testing usually includes a comparison of the computed results or functional behavior against expected results.

Engineering computer codes are validated by comparison of calculations with experimental or plant operating data, comparison against another validated code, or when none of the above are reasonably achievable, such as in computer simulation of a severe accident, independent review by experts. For engineering computer codes, comparison with experimental data is the preferred method of validation.

The experimental data selected for validation must have known uncertainties that are acceptably small. If the statistically combined uncertainties of the code calculations and the experimental data are within the accuracy (confidence) limits specified in the software requirements, then the code is validated as sufficiently accurate for its intended purpose.

Real Time software is validated by testing separate modules to provide assurance of correct translation between activities and proper function according to specified design requirements. In addition, testing of integrated modules and integration testing with hardware in the operating environment are performed to demonstrate that the software adequately and correctly performs all intended functions according to design requirements and demonstrates that the software:

- properly handles abnormal conditions and events as well as credible failures
- does not perform adverse unintended functions, and
- does not degrade the system either by itself, or in combination with other functions or configuration items.

At the end of the test phase, a formal review of the *Software Verification and Validation Report* including any separately developed documentation such as *Test Plan, Test Report* etc. is performed. Upon completion of this review, the documents are placed under configuration control.

7. Software Configuration Management

Software configuration management is the set of activities that provide control and traceability of software. The PPO ensures performance of the software configuration management activities as described in this Plan. Within PPO, software configuration management activities are performed by the PPO Configuration Management organization.

7.1 Configuration Identification

A computer software configuration item is a collection of software components, including the computer code and documentation, treated as a single entity for the purpose of configuration control.

A baseline is the approved computer software configuration items at a given time in its life cycle. The baseline is identified as the list of approved software components. The baselines, plus approved changes from those baselines, constitute the current configuration of the computer software configuration item. Baselines are used to assure revision control and to provide traceability of computer software configuration items.

Baselines are established by satisfactory completion of a document or code review or Software Configuration Control Board action.

Computer software configuration items are assigned unique identifiers by the Configuration Control Manager. The identification system does the following:

- Uniquely identifies each computer software configuration item;
- Identifies changes to computer software configuration items by revision;
- Provides the ability to uniquely identify each configuration of the revised software available for use.

This identification system is designed to prevent inadvertent or improper use of an incorrect version of a software product.

7.2 Configuration Control

Computer software configuration items, including items in their preliminary stages of development, are placed under configuration control following formal review and approval.

Software problem reports and software change proposals are used by the users and the Software Configuration Control Board to document problem and change request evaluations and their resolutions. Section 8 discusses these documents. Changes to computer software configuration items are formally documented using software change proposals. This documentation includes a description of the change, rationale for the change, and identification of affected software components.

If the cognizant manager determines that the change could have significant safety, mission, performance, cost or schedule impact, then evaluation and approval of the change becomes the responsibility of the Software Configuration Control Board. Changes to a computer software configuration item are evaluated and approved by at least the same level of management that approved the previous version of the configuration item to assure that the impact of the change is carefully assessed. Only approved changes are made to software baselines and subsequently processed by the Configuration Control Manager. Approved changes to computer software configuration items are subjected to the same verification and validation as the previous version of the computer software configuration item. Information concerning approved changes is sent to sponsors, users and other affected organizations.

The PPO Software Configuration Control Board consists of the following individuals or their designated representatives:

- Systems Engineering Manager
- Tritium Production Engineering Manager
- Balance of Plant Engineering Manager
- Configuration Control Manager
- Quality Assurance Manager
- Others, as designated by the chairperson of the Software Configuration Control Board

The Software Configuration Control Board is chaired by the Systems Engineering Manager or his designee.

7.3 Configuration Status Accounting

The following configuration status reports are maintained by the Configuration Control Manager as indicated:

- Engineering database listing all documents making up the configuration baseline for each computer software configuration item.
- Current status of all computer software configuration items and their related software components under configuration control.
- Complete list of all outstanding software problem reports and their status.
- Complete list of all outstanding software change proposals and their status.
- PPO list of approved computer codes and their approval status (i.e., approval for preliminary and final design).

8. Problem/Change Reporting and Corrective Action

Responsibility for administration of a software problem/change reporting and corrective action program is vested in the PPO Configuration Manager.

Requests for software modifications is presented in a standardized manner via completion of a uniquely-identified software problem report or software change proposal.

Software problem reports are used to report any error found in a computer program, its documentation, design, or specification. The software problem report identifies the error and includes a description of the circumstances leading to the error, such as input data, listings, and other supporting material.

Software change requests are used to request a modification to a computer program to correct errors found or to request an enhancement to a program or its documentation. The software change request identifies the change being requested and includes a brief requirements specification for the change.

Problems or changes are analyzed and reported by the proposing organization, and subsequently acted upon by the Software Configuration Control Board.

The cognizant manager and the Software Configuration Control Board use software problem reports and software change proposals to document evaluations of problems and change requests and their resolutions. PPO implementing procedures explain in detail the software problem report and software change proposal.

9. Software Quality Assurance Activities

PPO Software quality assurance (QA) activities ensure that processes throughout the software life cycle, such as documentation reviews and code reviews, are performed in accordance with this *Software Management Plan.* The actual performance of these QA activities is the responsibility of the organizations developing or modifying the software and is assigned to qualified individuals who did not participate in development, validation, or documentation of the software.

The responsible QA organization performs verification activities such as audits and surveillance, and participates in the Software Configuration Control Board.

10. Code Control – The Controlled Program Library

The PPO controlled program library, maintained by the Configuration Control Manager, is the controlled collection of software, documentation, and associated tools to facilitate the orderly development, support, and use of software. It provides storage of and controlled access to software and documentation in both human-readable and machine-readable form. In addition, the controlled program library provides for archived backup storage.

The controlled program library includes, either physically or by reference, all records necessary to provide evidence that software used on the APT Project by the PPO meets the design intent and satisfies contractual requirements. The controlled program library documents the status of configuration control and corrective action processes for each software configuration item. Records show proof that testing has been performed in accordance with an approved *Verification and Validation Plan.* For historical traceability, approved changes and reference data are retained. Record identification, collection, retention, and storage are in accordance with the PPO *Quality Assurance Program Plan* (QAPP).

The controlled program library provides version control for all configuration items from their initial entry throughout their life. Inadvertent modification of items placed under configuration control in the library is prevented by submitting new versions of the modified item and retaining old versions. Old versions may be retired/archived if no longer referenced in the design documentation.

The controlled program library maintains the electronic files for each executed computer program used as a basis for the design. This information includes, as applicable, the input data file, the output data file, and any other specific data such as employee name, task/work package name, date of execution, and relevant comments.

11. Software Procurement

Software procurement responsibility for PPO is vested in the Project Administration Manager and/or the Procurement/Subcontracts Manager.

Procurement procedures at PPO ensure appropriate review and approval by engineering and QA organizations. In any case involving procured software, the Procurement Manager, with the assistance of the cognizant manager, determines the adequacy of vendor policies and procedures in meeting requirements of the PPO QAPP. This may include an evaluation to determine the adequacy of the supplier to support software operation and maintenance, and to identify activities and documents needed for the software to be placed under configuration control. The number of documents may be limited to essential information, including a statement of requirements, V&V documentation, and a *User's Manual.* This documentation may be obtained from the supplier or reviewed at the supplier facility.

After the procured software passes evaluation it is placed under configuration control. This is done to track the different versions of the software as it is revised by the supplier and to provide access control for the software.

Acquisition documents specify that the supplier of software or related services report software errors or failures to the PPO Procurement Manager.

Software acquired from National Laboratories may be used in support of the design and/or safety review of APT structures, systems and components. The PPO Project Director ensures the processes used by the Laboratory to develop and control such software are adequate. Software provided by a National Laboratory to the PPO is considered Government Furnished Information (GFI) for this purpose.

12. Existing Software

Existing PPO software is defined as software or computer programs which were developed or procured prior to June 1, 1998. Requirements for existing software are given below. Decision authority, regarding documentation for software developed prior to this date, rests with the cognizant engineering manager.

12.1 Use of Existing Software

All existing software that has been approved for interim or final design calculation is placed under configuration management and is included in the PPO list of approved computer codes.

All software is placed under configuration control before it can be approved for interim use. Results obtained with software that has been approved only for interim design calculations are identified as "PRELIMINARY" and are confirmed as "FINAL" only after using fully verified and validated software.

After verification and validation testing has been completed and the (modified) software has been placed under configuration control, the software is approved for final design calculations. For software that has not yet been formally verified and validated, a determination for its use in final design calculations is made and documented, after assessment of potential risks, by the cognizant engineering manager.

12.2 Documentation Requirements for Existing Software

The software documentation requirements discussed in Section 4.3, applicable to newly developed software for PPO, are to be tailored for existing software. This determination is made and documented by the cognizant engineering manager, as appropriate to the software in question.

12.2.1 Acquisition of Existing Software

Acquisition of existing software includes a review by the user organization to formally evaluate each existing computer program (and its related documentation) intended for use by the PPO and for inclusion in the PPO list of approved computer codes. The evaluation takes into account the quality of existing documentation, the proposed user application requirements, and the nature and previous usage of the software. The determination regarding software documentation is made and documented by the cognizant engineering manager, as appropriate to the software in question.

At a minimum, documentation for existing software includes the following:

- Documentation of user application requirements either in a tailored version of a *Software Requirements Specification* or in existing documents
- A *Software Verification and Validation Plan* or equivalent existing documentation that identifies the items to be tested, testing to be performed, test sequences, responsible organizations, and evaluation criteria
- A *Software Verification and Validation Report* or equivalent existing documentation describing the results of the test cases specified in the *Software Verification and Validation Plan*
- A *User's Manual*

The evaluation is documented, and specific PPO documentation requirements for each software program are determined.

If existing software is to be used without modification, the documentation described above is sufficient. Additional documentation requirements pertaining to existing software undergoing maintenance are described in Sections 12.2.2 through 12.2.4.

12.2.2 Perfective Modifications of Existing Software

For existing software undergoing perfective modifications, the modifications must adhere to the same requirements as new software, however, only the modifications themselves have to be documented to new software standards. As a result, tailored versions of the following documents are allowed for existing software undergoing perfective modifications:

- The *Software Requirements Specification* may consist of existing documents plus the software change proposals used to implement the perfective modifications.
- The *Software Design Specification* may be abbreviated to encompass the design of modified portions of the existing software and to describe any changes to the external interface of the program's modules and functions.
- The *Software Verification and Validation Plan* or *Test Plan* and *Test Procedure* should include test cases designed to test, at a minimum, modified portions of the existing software and sufficient regression testing of unmodified portions to ensure that unintended side effects have been avoided. These documents should also address verification of changes against software change requests and validation of any program performance modified by the changes.
- The *Software Verification and Validation Report* or *Test Report* and/or *Test Evaluation Report* is prepared to document the validation testing.
- The *User's Manual* is modified, if necessary, to reflect changes.

12.2.3 Adaptive Modifications of Existing Software

For existing software undergoing adaptive modifications, the modifications adhere to the same requirements as new software, however, only the modifications themselves have to be documented to new software standards. As a result, tailored versions of the following documents are allowed for existing software undergoing adaptive modifications:

- The *Software Requirements Specification* may consist of existing documents plus the software change requests used to implement the adaptive modifications.
- The *Software Design Specification* is changed only if the code is modified during adaptation (e.g., translations do not require this document).
- The *Software Verification and Validation Plan* or *Test Plan* and *Test Procedure* and the *Software Verification and Validation Report* or *Test Report* and/or *Test Evaluation Report* are modified to reflect verification and validation of the adapted software through comparison of outputs to given input data for the original version of the software (prior to adaptation).
- The *User's Manual* is modified, if necessary, to reflect changes.

12.2.4 Corrective Modifications of Existing Software

For corrective modifications, the required documentation consists of changes to a copy of the original documentation when required by corrective actions:

- The *Software Requirements Specification* may consist of the existing document plus changes made to reflect corrective action taken.
- A *Software Verification and Validation Plan* or *Test Plan* and *Test Procedure* is prepared to test the corrections, and a *Software Verification and Validation Report* or *Test Report* and/or *Test Evaluation Report* is prepared to document the testing.
- The *User's Manual* is modified, if necessary, to reflect corrections.

12.3 Verification of Existing Software

The general verification requirements discussed in Section 6.1, applicable to all newly developed software, are tailored for existing software. Different methods for tailoring verification requirements are implemented depending on whether the existing software is used with or without modifications. The determination regarding software verification documentation is made and documented by the cognizant engineering manager, as appropriate to the specific software in question.

For all existing software, upon acquisition, it is shown that the software meets users' application requirements. Section 12.2.1 defines the process for the evaluation of existing software upon acquisition. The verification of documentation required by this evaluation process follows the approach outlined in Section 6.1.

Modifications to existing software require document changes as specified in Sections 12.2.2 through 12.2.4. The required documentation is verified according to the approach outlined in Section 6.1.

12.4 Validation of Existing Software

The general validation requirements discussed in Section 6.2, applicable to all newly developed software for PPO work, are tailored for existing software. Different validation requirements may result depending on whether the existing software is used with or without modification. The determination regarding validation and its documentation is made and documented by the cognizant engineering manager, as appropriate to the specific software in question.

For existing software used without modifications, if the software has been previously validated and is being used in a manner consistent with the prior validation, then it does not need to be revalidated. If the software has not been previously validated, then it is validated following the guidelines established in Section 6.2. Where software must remain unvalidated until system start-up testing, decisions to use the unvalidated software are made once programmatic risks have been assessed and appropriately documented.

Modifications to existing validated software are subjected to the same validation requirements as would apply to new, unmodified software, although the validation can be abbreviated to consider only the effects of the modifications themselves. Validation of software modifications includes selective regression testing to detect errors introduced during the modification of systems or system components. This testing ensures that the modifications have not caused unintended adverse affects and ensures that modified systems or system components still meet specified requirements.

12.5 Configuration Management of Existing Software

Existing software used for design work is placed under configuration control as follows:

- The initial configuration baseline consists of executable code, source code, and documentation.
- At this point selected computer software may be approved for interim use by the cognizant engineering manager (Refer to Section 12.1 regarding the approval of software for interim use).
- The *Software Requirements Specification*, *Software Design Description*, *User's Manual*, and *Software Verification and Validation Plan* or *Test Plan* and *Test Procedure*, if applicable, are made part of the configuration baseline.
- As codes are verified and validated, the *Software Verification and Validation Report* or *Test Report* and/or *Test Evaluation Report* for each software is made part of the configuration baseline. A *User's Manual* is prepared, issued, and included in the configuration baseline. After reaching this stage, the software is considered under configuration control and authorized for use in final design calculations or operation.
- During the preparation for modifying existing computer software, the *Software Verification and Validation Plan* or *Test Plan* and *Test Procedure* and the *Software Design Description* for the modified portion of the existing software is made part of the configuration baseline.
- Following the testing of modified computer software, the *Software Verification and Validation Report* or *Test Report* and/or *Test Evaluation Report* for the modified portion of the existing software is made part of the configuration baseline.

Appendix A- Definitions

Adaptive Maintenance: Maintenance activity to adapt the software to changes in its operating environment (e.g., different computers or operating systems).

Baseline: The formally reviewed and approved computer software configuration item at a given time in its life cycle. The baseline can only be changed through formal change control procedures.

Computer Program: A sequence of instructions suitable for processing by a computer. Processing may include the use of an assembler, a compiler, a linker, an interpreter, or a translator to prepare the program for execution as well as to execute it.

Computer software configuration item: A software component treated as a unit for the purpose of configuration control.

Computer System: Integrated software and hardware designed to perform calculations or enable the operation, control, and monitoring of components and processes.

Configuration: The functional and/or physical characteristics of hardware/software as set forth in technical documentation and achieved in a product.

Configuration Control: The process of identifying and defining the configuration items in a system, controlling the release and change of these items throughout the system life cycle, recording and reporting the status of configuration items and change requests.

Configuration Identification: The process of assigning a unique identifier to software for the purpose of configuration control.

Configuration Management: A discipline applying technical and administrative direction and surveillance to identify and document the functional and physical characteristics of a configuration item; control changes to those characteristics; and record and report change processing and implementation status.

Configuration Status Accounting: The recording and reporting of the information that is needed to manage the configuration effectively, including a listing of the approved configuration identification, the status of proposed changes to configuration, and the implementation status of approved changes.

Controlled Program Library: A controlled collection of software, documentation, and associated tools to facilitate the orderly development, support, and use of software.

Corrective Maintenance: Maintenance activity to remove errors located in software during the operations and maintenance phase of the software life cycle.

Developer: The individual or group responsible for constructing or modifying the software.

Engineering and Scientific Software: Software used to perform engineering and scientific calculations.

Functional Classification: The following safety and QA classifications for Structures, Systems, and Components (SSCs) are used to determine the applicability of software related activities:

- Safety Class - SSCs providing protection for the public from uncontrolled releases of hazardous materials.
- Safety Significant - SSCs providing protection for on-site workers and defense-in-depth from hazardous materials.
- Production Support – SSCs protecting the facility from events impacting its mission.

Hardware: Physical equipment used in data processing, as opposed to computer programs, procedures, rules, and associated documentation.

Maintenance: Modification of software. See also Adaptive Maintenance, Corrective Maintenance, and Perfective Maintenance.

Nonconformance: A deficiency in characteristic, documentation, or procedure that renders the quality of an item or activity unacceptable or indeterminate.

Perfective Maintenance: Maintenance activity to respond to new or revised requirements.

Quality Assurance: All those planned and systematic actions necessary to provide adequate confidence that a software shall perform satisfactorily in service.

Real Time Software: Software applications that must respond within a specified time frame and that usually form an integral part of the plant configuration. Real time software includes operational software and control software. Operational software is used to operate or monitor components within the plant and/or plant systems. Control software is a set of programs, associated data, procedures, rules, documentation and material concerned with the development, use, operation, and maintenance of a computer system used for a control system.

Regression Testing: Selective retesting to detect faults introduced during modification of a system or system component, to verify that modifications have not caused unintended adverse effects, or to verify that a modified system component still meets its specified requirements.

Review: An evaluation of software component(s) or project status to ascertain discrepancies from planned results and to recommend improvement. This evaluation follows a formal process (i.e. management review process, technical review process, or software inspection process.)

Software: Computer programs, procedures, rules, and associated documentation and data pertaining to the operation of a computer system. Software can be divided into a variety of functional types including Scientific and Engineering software, Real Time software, and Utility software.

Software Category: An indicator of the time period during which PPO computer software was developed.

- Existing Software: Computer software developed prior to June 1, 1998.
- New Software: Computer software developed on or after June 1, 1998.
- Modified Software: Existing or new software that is changed, revised, or altered on or after June 1, 1998.

Software Component: An item associated with a software product, such as the source code, the executable code, a data file, a procedure, a document, etc.

Software Development Cycle: The period of time that begins with the decision to develop a software product and ends when the product is delivered. This cycle typically includes a requirements phase, design phase, implementation phase, test phase, and sometimes, an installation and checkout phase.

Software Documentation: Recorded information that documents the requirements and the design, explains the capabilities of the software, provides requirements for testing and installation of the software, results of testing, operating and maintenance instructions, and may include source code listings.

Software Life Cycle: The period of time that starts when a software product is conceived and ends when the software product is no longer available for routine use. The software life cycle typically includes a requirements phase, a design phase, an implementation phase, a test phase, an installation and checkout phase, an operation and maintenance phase, and sometimes a retirement phase.

Software Quality Assurance: Those verification activities associated with the development, maintenance, and procurement of software products necessary to provide adequate confidence that the software conforms to established requirements.

Sponsor: The individual or group requiring the software for use.

Systems Software: A collection of computer programs written to service other computer programs. Some examples of systems software include compilers, editors, and file management utilities.

Software Testing: The process of operating a computer system, observing and recording the results, and making an evaluation to verify that specified requirements are met or identify errors.

Test Case: A specific set of test data and associated procedures developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement.

Traceability: The ability to trace the history, application, or location of an item and like items or activities by means of recorded identification.

Utility Software: Software used for computer aided software engineering including software to produce drawings, diagrams, spreadsheets, and databases.

Validation: The test and evaluation of completed software to ensure compliance with software requirements.

Verification: The process of determining whether or not the product of a given phase of the software development cycle fulfills the requirements imposed by the previous phase.