

NUCLEAR WASTE MANAGEMENT PROGRAM

CONTROLLED COPY NO. _____

0089

No.: 033-YMP-R Appendix H

Revision: 0

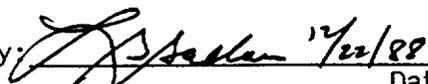
Date: December 15, 1988

Page: 1 of 10

 Subject: APPENDIX H - REQUIREMENTS FOR COMPUTER SOFTWARE USED
TO SUPPORT A HIGH-LEVEL NUCLEAR WASTE
REPOSITORY LICENSE APPLICATION

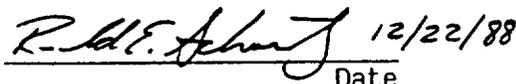
Approved: FEB 10 1988

Approved by:


 12/22/88
Date

Yucca Mountain Project Leader

Approved by:


 12/22/88
Date

Quality Assurance Manager

1.0 OBJECTIVES

The purpose of this appendix is to establish requirements for the development, management, control, and documentation of software used to support the Yucca Mountain Project (YMP). The software requirements are intended to ensure software quality and to provide part of the basis on which YMP will evaluate the soundness of the software used.

This appendix supplements 033-YMP-R 3 of this QAPP and is used in conjunction with that section as applicable.

2.0 APPLICABILITY

The requirements set forth in this appendix apply to computer software used to produce or manipulate data that is used directly in site-characterization and performance assessment analyses and in the design, analysis, and operation of repository structures, systems, and components.

Written procedures are established that assure the requirements of this appendix are implemented in a consistent and systematic manner. The extent to which these requirements apply are defined in the software QA plan and is related to the nature, complexity, and importance of the software applications.

3.0 TERMS AND DEFINITIONS

Terms and definitions used in this appendix for software are defined below:

Baseline: As used for computer software: (1) The stage of computer software at a completed and reviewed phase of the software lifecycle; (2) Approved documentation generated within or as a result of completing a phase of the software life cycle.

Computer Code: A set of computer instructions for performing the operations specified in a numerical model.

No.: 033-YMP-R Appendix H	Revision: 0	Date: December 15, 1988	Page: 2 of 10
------------------------------	----------------	----------------------------	------------------

Configuration Management: As used for computer software: (1) A system for orderly control of software, including methods used for labeling, changing, and storing software and its associated documentation. (2) The systematic evaluation, coordination, approval or disapproval, and implementation of all approved changes in an item of software after establishment of its configuration.

Computer Code Verification: Assurance that a computer code correctly performs the operating specified in a numerical model (NUREG-0856). Usually accomplished by comparing code results to (1) a hand calculation, (2) an analytical solution or approximation, or (3) a verified code designed to perform the same type of analysis (benchmarking).

Discrepancy: Condition adverse to quality; reference to any of the following: failures, malfunctions, deficiencies, defective items, and nonconformances.

Life Cycle: See software development life cycle.

Model: A representation of a physical system, based on scientific principles and laws, that transforms a set of input information or data into another set of output information or data.

Model Validation: Assurance that a model as embodied in a computer code is a correct representation of the process or system for which it is intended (NUREG-0856). Usually accomplished by comparing code results to (1) physical data, or (2) a verified and validated code designed to perform the same type of analysis (e.g., benchmarking with a validated code). Peer review may be used for model validation if it is the only available means for validating a model.

Numerical Method: A procedure for solving a problem primarily using numerical methods.

Numerical Model: A representation of a process or system using numerical methods.

Software: A set of computer operations specified in any programming language that can be translated unambiguously into machine language. (Operations specified in machine language are also software).

Software-development Life Cycle: A method of project planning and documentation for the development of a software product. Life cycle allows optimal traceability regarding the goals, restrictions, decisions made, and current progress of a code.

4.0 SOFTWARE VERIFICATION AND MODEL VALIDATION

Software verification and model validation activities are performed as described in the software QA plan.

No.:	Revision:	Date:	Page:
033-YMP-R Appendix H	0	December 15, 1988	3 of 10

4.1 SOFTWARE VERIFICATION

Verification plans employ methods such as inspections, analyses, demonstrations, and tests to assure that the software adequately and correctly performs all intended functions and that the software does not perform any function that, either by itself or in combination with other functions, can degrade the entire system.

Verification activities are performed according to written procedures relative to specific hardware configurations. The amount of verification activity is determined by the type and complexity of the software. The results of verification are documented in accordance with Section 6.0 and reviewed in accordance with Section 7.0 of this appendix.

4.2 MODEL VALIDATION

Model validation activities are performed according to written procedures to demonstrate that models embodied in computer software are correct representations of the process or system for which they are intended. This is accomplished by comparing software results against verified and traceable data obtained from laboratory experiments, field experiments or observations, or in-situ testing. Specific sets of data used in the validation process are identified, and justification is documented for their use. When data are not available from the sources mentioned above, alternative approaches may be used and are documented. Alternative approaches may include peer review and comparisons with the results of similar analyses performed with verified software. The results of the model validation are documented according to Section 6.0 and reviewed according to Section 7.0.

5.0 SOFTWARE CONFIGURATION MANAGEMENT

Software configuration management system is established to assure positive identification of software and control of all software baseline changes.

5.1 CONFIGURATION IDENTIFICATION

Software configuration baseline items are identified at the appropriate phase of each software lifecycle. Approved changes to a baseline are added to the baseline as updates. A baseline plus updates specify the most recent software configuration. A labeling system for configuration items is implemented that:

- o Uniquely identifies each software configuration item or version identifier.
- o Identifies changes to software configuration items by revision identifiers.
- o Facilitates placement of the software configuration item in a relationship with other configuration item.

No.:	Revision:	Date:	Page:
033-YMP-R Appendix H	0	December 15, 1988	4 of 10

5.2 CONFIGURATION CHANGE CONTROL

Changes to software configuration items are formally controlled and documented. This documentation contains a description of the change, the identification of the originating organization, the rationale for the change, and the identification of affected baseline and software configuration items. Assurance is provided that only authorized changes are made to software baselines and software configuration items.

5.3 CONFIGURATION STATUS ACCOUNTING

The information that is needed to manage software configuration items is recorded and reported. The information includes the approved configuration identification, the status of formal proposals for changes to software configuration items, the implementation status of approved changes, and all information to support the functions of configuration identification, and configuration control.

6.0 DOCUMENTATION

Documentation is required as defined by the software QA plan. The following is acceptable documentation of computer software used on the Yucca Mountain Project. Additional documentation may also be identified in the software QA plan.

6.1 SOFTWARE LIFE CYCLE DOCUMENTATION

6.1.1 SOFTWARE LIFECYCLE REQUIREMENTS SPECIFICATION

Software requirements documentation outlines the requirements that the software must fulfill. A specific capability of software is called a requirement only if its achievement can be verified by a prescribed method. The requirements address the following as applicable to the software application:

- o Functionally - the functions the software are to perform.
- o Performance - the time-related issues of software operation such as speed, recovery time, response time, etc.
- o Design constraints imposed on implementation - any element that will restrict design options.
- o Attributes - non-time-related issues of software operation such as portability, correctness, security, maintainability, etc.
- o External Interface - interactions with other participants, hardware, and other software.

No.:	Revision:	Date:	Page:
033-YMP-R Appendix H	0	December 15, 1988	5 of 10

6.1.2 SOFTWARE LIFECYCLE DESIGN DOCUMENTATION

Software design documentation addresses the following as applicable to the software application:

- o A description of the major components of the software design as they relate to the requirements of the software requirements specification.
- o A technical description of the software with respect to control flow, data flow, control logic, and data structure.
- o The description of the allowable and tolerable ranges for inputs and outputs.
- o The design described in a manner that is easily traceable to the software requirements.
- o A description of life cycle verification activities.

6.1.3 SOFTWARE LIFECYCLE IMPLEMENTATION DOCUMENTATION

Software implementation documentation addresses the following as applicable:

- o Source code listing.
- o Revised requirements documents.
- o Revised design documents.

Any design changes made to the requirements and design phase document are assessed as to the impact to the design. The revised requirements and design phase documents are reviewed at the same review level as the original documents.

6.1.4 SOFTWARE LIFECYCLE TESTING DOCUMENTATION

Life cycle testing activities are documented. Software testing documentation includes a plan that describes the tasks and criteria for accomplishing the verification of the software in this phase. The documentation also specifies the hardware and system software configuration(s) for which the software is designed. In those cases where testing is used to ensure that requirements were met in the software design, test documentation provides traceability from requirements to design as implemented in the code. This documentation also includes a report on the results of the execution of the life cycle verification activities. This report includes the results of all reviews, audits and tests, and a summary of the status of the software.

6.2 MANDATORY DOCUMENTATION

The following mandatory documentation (consistent with NUREG-0856) is provided to meet the requirements of Section 3.2 of this QAPP, as applicable:

- o Software Summary
- o Mathematical and Numerical Models

No.:	Revision:	Date:	Page:
033-YMP-R Appendix H	0	December 15, 1988	6 of 10

- o User's Manual
- o Code Assessment and Support
- o Continuing Documentation and Code Listing

7.0 REVIEWS

Documentation produced during software development, acquisition, implementation, testing, and use is subject to appropriate reviews as described in the software QA plan.

7.1 SOFTWARE LIFE CYCLE REVIEWS

Reviews of software life cycle activities are performed for each life cycle phase completed. The procedures used for reviews identify the reviewers and their responsibilities.

The documentation for all reviews contains a record of review comments and the personnel responsible for comment resolution. After review comments are resolved, the approved documents are updated and placed under configuration management.

The following reviews are performed as applicable:

7.1.1 SOFTWARE LIFECYCLE REQUIREMENTS REVIEW

The review of software requirements is performed at the completion of the software requirements documentation. This review assures that the requirements are complete, verifiable and consistent. The review assures that there is sufficient detail available to facilitate definition of the software design or acquisition.

7.1.2 SOFTWARE LIFECYCLE DESIGN REVIEW

The software design review is held at the completion of the software design documentation. This review evaluates the technical adequacy of the design approach and assures that the design satisfies all the requirements in the requirements documentation. The complexity of the software design may require the performance of multiple design reviews.

7.1.3 SOFTWARE LIFECYCLE IMPLEMENTATION REVIEW

The software implementation review is an evaluation of the completed software lifecycle requirements, design, and implementation processes.

7.1.4 SOFTWARE LIFECYCLE TESTING REVIEW

The software testing review is an evaluation of the adequacy of completed software lifecycle verification activities.

No.:	Revision:	Date:	Page:
033-YMP-R Appendix H	0	December 15, 1988	7 of 10

7.2 MANDATORY REVIEWS

Mandatory documents (consistent with Section 6.2 of this appendix) are reviewed and documented in accordance with review procedures established in the software QA plan.

The adequacy of verification activities is reviewed. Also the adequacy of model-validation activities is reviewed.

8.0 DISCREPANCY REPORTING AND CORRECTIVE ACTION

Formal procedures are established for software discrepancy reporting and corrective action. This discrepancy reporting system is integrated with the configuration management system to assure formal processing of discrepancy resolutions.

Software discrepancy procedures assures that, as a minimum:

- o Defects are documented and evaluated for possible corrective action.
- o Defects are assessed for impact on previous applications.
- o Corrections are reviewed and approved before changes to software configuration items are entered into baselines.
- o Preventive and corrective actions provide for appropriate notification of organizations to which controlled copies have been distributed.

9.0 MEDIA CONTROL AND SECURITY

Physical media containing the images of software are physically protected to prevent their inadvertent damage, degradation, or loss.

10.0 SOFTWARE ACQUISITION, PROCUREMENT, AND TRANSFER

Procedures are established for controlling the acquisition or procurement of computer software from an outside organization and for the transfer of computer software to an outside organization.

Software requests by participating organizations includes appropriate criteria to enable the software received to comply, as much as possible, with the requirements of this QA plan. Requirements not satisfied at the time when the software is received, are completed by the organization in the appropriate phase of the applicable software life cycle. For those requirements that are not satisfied, the reasons are documented for distribution to the users.

Configuration management requirements apply to acquired or procured software using the product originally received as the initial baseline. Configuration management records document any conversions, modifications, configuration changes, or additional software required to make the software functional.

No.:	Revision:	Date:	Page:
033-YMP-R Appendix H	0	December 15, 1988	8 of 10

11.0 SOFTWARE QUALITY ASSURANCE PLAN

A software QA plan is prepared that describes the software development, acquisition and applications undertaken. Individual software QA plans may additionally be prepared for specific software products. The software QA plan identifies the:

- o Organizational responsibilities for the management and control of software.
- o Software products to which the software QA plan applies.
- o Criteria for meeting the requirements set forth in this appendix to the applicable software.
- o Software life-cycle model used.
- o Required documentation.
- o Software configuration-management system.
- o Verification and validation methodologies.
- o Discrepancy reporting and corrective actions.
- o Software review procedures.

Software lifecycle management is a requirement, and the software QA Plan presents the specific software lifecycle controls. A generic lifecycle that presents the conceptual lifecycle management steps is presented in Section 11.1.

11.1 SOFTWARE LIFECYCLE

A software life cycle model that requires that software development or acquisition proceed in a traceable, planned, and orderly manner is implemented. The relative emphasis placed on the phases of the software life cycle will depend on the nature, complexity, importance, and intended applications of the software.

The following lifecycle elements apply as appropriate for the specific lifecycle model defined, interpreted, and described in the software QA plan.

11.1.1 LIFECYCLE REQUIREMENTS PHASE

During this phase, requirements that pertain to functionality, performance, design constraints, attributes, and external interfaces of the completed software are specified, documented, and reviewed. These requirements include the following characteristics:

- o A format and language that is understood by the programming organization and the user.
- o Enough detail to allow for objective verification.

- o Adequate definition to provide for the response of the software to the identified input data.
- o The information necessary to design the software without prescribing the software design itself.

11.1.2 LIFECYCLE DESIGN PHASE

During the design phase, a software design based on the requirements is specified, documented, and systematically reviewed. The design specifies the overall structure (control and data flow), and the reduction of the overall structure into physical solutions (algorithms, equations, control logic, and data structures). The design may necessitate the modification of the requirements documentation.

Verification activities during this phase consist of, but are not limited to:

- o The planning for design-based test cases.
- o The review and analysis of the software design.
- o The verification of the software design.

11.1.3 LIFECYCLE IMPLEMENTATION PHASE

During this phase, the design is translated into a programming language and the implemented software is debugged. Only minor, if any, design issues are resolved at this phase.

Verification activities during this phase consist of:

The possible modification of test cases necessary due to design changes made during coding.

The examination of source code listings to assure adherence to coding standards and conventions.

11.1.4 LIFECYCLE TESTING PHASE

The testing phase consists of verification activities. Software verification will be essentially completed during this phase. The verification activities will include:

- o Execution of the test cases and evaluation of the results.
- o Evaluation of the completed software to assure adherence to the requirements.
- o The preparation of a report describing the results of software verification.

Model validation is conducted in accordance with Section 4.2 of this appendix. Because model validation is application dependent, model validation may not be completed at this stage.

No.:	Revision:	Date:	Page:
033-YMP-R Appendix H	0	December 15, 1988	10 of 10

11.1.5 LIFECYCLE INSTALLATION AND CHECKOUT PHASE

During this phase, the software may become part of a system incorporating other software components, the hardware, and production data. The process of integrating the software with other components may consist of installing hardware, installing the program, reformatting or creating databases, and verifying that all components have been included.

Testing activities during this phase consist of the execution of test cases for installation and integration. Test cases from earlier phases are used for installation testing.

11.1.6 LIFECYCLE OPERATIONS AND MAINTENANCE PHASE

During the operations and maintenance phase, the software has been approved for operational use. Maintenance activity consists of identification of latent errors and notification of users. Further activities may consist of maintenance of the software to remove latent errors (corrective maintenance), response to new or revised requirements (perfective maintenance), or adaptation of the software to changes in the software environment (adaptive maintenance). Software modifications are approved, documented, tested, and controlled in accordance with software configuration management requirements.

12.0 SOFTWARE APPLICATIONS

Procedures are established for controlling the application of software that perform technical calculations in support of site-characterization and performance assessment analyses and for the design, analysis, and operation of repository structures, system, and components. These software applications are reviewed and approved to assure that the software selected is applicable to the problem being solved and that input data assumptions are valid and traceable.

Procedures are established for documenting software applications that perform technical calculations to ensure that these applications and the results of these applications can be independently reproduced.

Procedures are established for reviewing these applications to provide reasonable assurance that the software used is appropriate for the intended application and that the results produced are accurate. Documentation appropriate for a given application or analysis includes the computer code, the input data, the assumptions or approximations employed to develop the input data, and appropriate user documentation for performing the application or analysis.