

No.: Y80-101PD

Title: Software Management Program Description

Revision: 09/14/01

The purpose of the Software Management Program is to govern the development, acquisition, modification, and control of software at the Y-12 National Security Complex (Y-12 Complex).

A hard copy of this document is valid only until the document revision number has changed on the web. The hard copy should be dated and signed the day it is printed. If you continue working from the hard copy, you should verify its accuracy on the web.

Date Printed: _____

Verifier: _____

BWXT Y-12, L.L.C.
Management Requirements

Number: Y80-101PD
Rev. Date: 09/14/01
Supersedes: 12/15/00
Page: 1 of 17

BWXT Y-12
Management Control

Subject: Software Management Program Description

S. L. Chapman/L. W. Koester /S/
Program Description Written by

9/14/01
Date

Approvals: Approval Dates and Signatures on File

C. H. Moseley Jr. /S/
Functional Area Manager/Owner

9/17/01
Date

Donald C. Johanson /S/
Executive Manager

9/18/01
Date

Clarence C. Hinton /s/
Requirements and Issues Management

9/19/01
Date

09/30/01
Effective Date

Re-Affirm Date

This document has been reviewed by an Authorized Derivative Classifier and UCNI Reviewing Official and has been determined to be unclassified and contains no UCNI. This review does not constitute clearance for public release.

Original Signed By:

D. O. Colclasure 9/14/01
Name & Date

Subject: Software Management Program Description

REVISION LOG

Revision Date	Description of Change	Pages Affected
9/14/01	DM/R 01-QAO-08 Revise and Clarify contents consistent with Y80-101INS contents.	All
12/15/00	DM/R 00-QAO-15 Place into Management Requirements format. Revise contents consistent with current customer and Y-12 Complex requirements and established standards. Number changed to Program Description.	All
11/01/2000	BWXT Y-12 Blue Sheet for adopting procedure with listed changes	All

Subject: Software Management Program Description

Table of Contents

1.0	Purpose.....	5
2.0	Scope.....	5
3.0	Requirements.....	5
3.1	Instruction.....	6
3.2	Corporate Information Technology Architecture and Strategies.....	7
3.3	Corporate Information Management.....	7
4.0	Software Management.....	7
4.1	Development Methods.....	7
4.2	Life Cycle.....	9
4.3	Applicability.....	9
4.4	Adaptability.....	9
5.0	Implementation.....	10
5.1	Preparation.....	10
5.1.1	Evaluation.....	10
5.1.2	Exemption.....	10
5.2	Life Cycle Stages.....	10
5.2.1	Planning.....	10
5.2.2	Requirements Definition.....	11
5.2.3	Functional Design.....	11
5.2.4	System Design.....	11
5.2.5	Programming.....	11
5.2.6	Software Integration and Testing.....	11
5.2.7	Software Installation and Acceptance.....	11
5.2.8	Software Maintenance.....	11
5.2.9	Retirement.....	11
6.0	Organizational Interfaces.....	12
6.1	Computing and Telecommunications Security Organization (CTSO).....	12
6.2	Corporate Information Office (CIO).....	12
6.3	Quality Organization.....	12
6.4	Product Engineering.....	12
7.0	Roles and Responsibilities.....	12
7.1	Boards and Teams.....	12
7.1.1	Software Review Board.....	12
7.1.2	Software Project Team.....	13
7.2	Individuals.....	13
7.2.1	Software Owner.....	13
7.2.2	User Point of Contact (POC).....	13
7.2.3	Project Manager.....	13
7.2.4	Lead Analyst.....	13
7.2.5	Software Program Manager.....	14
7.2.6	Product Engineer (PE).....	14
7.2.7	Quality Engineer (QE).....	14
7.2.8	Computing and Telecommunications Security Organization (CTSO) Representative.....	14
7.2.9	Information System Security Officer (ISSO).....	14
8.0	Implementing Documents.....	14
9.0	Source Documents.....	14

Subject: Software Management Program Description

Table of Contents (Continued)

10.0	References.....	15
11.0	Documentation and Records	15
	Figure 1	16
	Figure 2	17

Subject: Software Management Program Description
--

1.0 PURPOSE

The purpose of the Software Management Program is to govern the development, acquisition, modification, and control of software at the Y-12 National Security Complex (Y-12 Complex). The Software Management Program:

- provides instruction for identifying consequences of software failure and assigning impact level;
- categorizes and classifies software to allow for graded implementation;
- identifies quality, security, and software engineering requirements; and
- establishes documentation, signatures, and evidence retention requirements.

Compliance with the Software Management Program ensures that Department of Energy (DOE) and Y-12 Complex software acquisition, development, and management requirements are met. New requirements will be incorporated into the program as necessary. Requirements from other programs applicable to software must be identified and addressed by the Software Project Team and documentation established which serves as evidence of compliance with these requirements. This documentation, when appropriately identified, may be incorporated into those deliverables required by Software Management.

Implementation of the Software Management Program, documented in Y80-101INS, "Software Management Instruction," is discussed in more detail in Section 5 of this program description. The instruction allows combination of and selected exclusion of defined life cycle stages and deliverables. The emphasis is on providing flexibility and adaptability in a graded process appropriate for the specific software and data.

2.0 SCOPE

This program description and implementing instruction apply to the following:

- software developed, procured, or otherwise obtained from an external or government source which is implemented directly or modified for use and controlled and/or maintained for use;
- software internally developed and maintained;
- data management (i.e. data generated, collected, stored, managed and used by software; and
- software developed, procured, obtained, or modified for use external to Y-12 in the absence of an accepted alternative Software Quality Assurance Program.

This program description and implementing instruction does not apply to the following:

- software that is developed, procured, obtained, or modified by BWXT Y-12 for a contracting entity where the contracting entity requires the use of an alternate Software Quality Assurance program. For record purposes this software must be entered into the Software Application Manager (SAM) for evaluation.

3.0 REQUIREMENTS

This document, Y80-101PD, "Software Management Program Description," establishes tasks to be performed when implementing identified quality, security, and software engineering, requirements derived from the following three sources. First, facility quality requirements are extracted from the *Quality Program Description*, Y60-101PD, a part of the contractual agreement between the Y-12 Complex and DOE, and weapons software quality requirements are extracted from DOE AL QC-1, *Quality Criteria*, and are correlated to the requirements extracted from Y60-101PD. Second, security requirements are extracted from the *Automated Information System (AIS) Security Handbook*, Y19-401INS, which implements the security requirements from DOE Order 471.2A, DOE M 471.2-2, DOE N 205.1, and Office of Management and Budget

Subject: Software Management Program Description
--

3.0 REQUIREMENTS (Continued)

(OMB) Circular A-130 Appendix III. Third, software engineering methodology requirements are derived from DOE G 200.1-1A, *Software Engineering Methodology*. A compilation of these requirements appears in the Administrative and Technical Basis document for the instruction.

3.1 Instruction

The Y80 Series Instruction documents the activities required to implement the Software Management program. The instruction is organized by section covering introduction, requirements, roles and responsibilities, evaluation, exemption, life cycle stages, and appendices. The introduction presents the general requirements and performance expectations for quality, security, and software engineering. The roles and responsibilities section identifies who performs what actions. The evaluation process identifies the information required for a software project and decisions to be made and documented based on this information. The exemption process identifies the actions and documentation required to exempt an identified deliverable for a specific software project. The "Life Cycle Stages" describes the deliverables required for each stage, the approval reviews to be performed with the requisite signatures, and the requirements for exempting a specific deliverable. Each Y80 Series Instruction section contains a purpose, strategy, and roles and responsibilities. Sequencing of the actions has been specifically avoided to allow for optimum utilization of resources and funding as a software project is implemented. Stage Exit forms must be completed and signed, indicating review of deliverables and concurrence by assigned PT members prior to closing the stage. The appendices contain references, guidance, forms, and definitions to aid in the implementation of a software project.

The Y80 Series Program Description and Instruction are available from Plant Records and electronically on the "Y-12 Internal Home Page" from the Management Requirements home page:

Comments may be submitted to the Manager of Quality Programs . Current versions of all forms having UCN numbers that are referenced in the Y80 Series are available from the Y-12 Internal Web home page for Just-in-Time Forms:

In addition, a Software Application Manager is provided from the Y-12 Internal Web Home Page:

This application provides the necessary tools and forms to collect project information and store it for future use as well as determine the application, type, source, and failure impact. Also provided are the templates to facilitate the generation and retention of the required deliverables for the software project. The information and deliverables entered through this Web-based application are maintained as records for the software project with configuration control as required when the software project is placed in maintenance and receives revisions in the future. Software Project Managers and Teams are expected to utilize the Software Application Manager. Where the contracting entity requires an alternate Software Quality Assurance program only the Evaluation portion of the Software Application Manager is required.

A Glossary and Abbreviations is provided in Appendix A of Y80-100INS.

A failure impact level for each software project will be determined based on the consequences associated with failure of the application. This failure impact level determination will serve as the basis for applying the life cycle stage deliverables to the software project. Life cycle stages are discussed in greater detail in Section 4.1 of this program description. An overall deliverable matrix is provided in Appendix B of Y80-101INS to provide direction on the deliverables for each life cycle stage of a software project.

Subject: Software Management Program Description
--

3.0 REQUIREMENTS (Continued)

3.2 Corporate Information Technology Architecture and Strategies

The Corporate Information Office (CIO) is responsible for establishing information technology architecture strategies and associated standards. These standards and strategies include those for the areas of database, server, desktop, network, and electronic security. Strategies shall encourage the use of Y-12 Complex-wide solutions in all functional areas to obtain integration across Y-12 and maximum cost benefit. Current versions of information technology strategies are available from the CIO or via the "Y-12 Information Technology Portal" on the "Y-12 Internal Home Page."

3.3 Corporate Information Management

Information shall be managed and controlled in a manner that is consistent with overall DOE and Y-12 Complex computer security and information architecture policies and strategies. The CIO is responsible for defining the Y-12 Complex-wide information strategy to which database implementations, products, services, and interfaces shall adhere. The Computing and Telecommunications Security Organization (CTSO) is responsible for overseeing the implementation of computer security policies. Information generated within the Y-12 Complex is considered a corporate resource that may be used as appropriate throughout the company to conduct business. Therefore, information management policies shall encourage the use of standardized database management system (DBMS) products, interfaces, and data formats wherever possible to promote efficient data sharing across the Y-12 Complex.

4.0 SOFTWARE MANAGEMENT

The software development process utilized in the Y-12 Complex is diagrammed in Figure 1. The software project team selects the software development method appropriate for the software project. Development methods that may be used are as follows:

4.1 Development Methods

This section describes some examples of development methods that may be used in implementing the software engineering methodology through implementing instructions. The software project team does not have to formally select a method and document the selection. The methods are provided to demonstrate to the flexibility in adapting the lifecycle stages to produce software using the implementing instruction. Two or more methods may be used simultaneously or in sequence. The methods provided below are a few of the most common examples and are not intended to be a comprehensive list of development methods.

Structured Development: Structured development incorporates the traditional "waterfall" method where each stage is completed prior to moving on to the succeeding stage. This method is very restrictive, but has usefulness in some situations, particularly for minor software projects.

Segmented Development: Segmented development is most often applied to large software engineering projects where the project requirements can be divided into functional segments. Each segment becomes a separate project and provides a useful subset of the total capabilities of the full product. This segmentation serves two purposes: to break a large development effort into manageable pieces for easier project management and control; and to provide intermediate work products that form the building blocks for the complete product. The lifecycle stages, actions, and deliverables are applied to each segment. The overall system and software objectives are defined, the system architecture is selected for the overall project, and a Project Plan for development of the first segment is written and approved by the system owner.

Subject: Software Management Program Description
--

4.0 SOFTWARE MANAGEMENT (Continued)

Spiral Development: Spiral development repeats the planning, requirements, and functional design stages in a succession of cycles in which the project's objectives are clarified, alternatives are defined, risks and constraints are identified, and a prototype is constructed. The prototype is evaluated and the next cycle is planned. The actions and deliverables for the Planning, Requirements Definition, and Functional Design Stages are repeated in each cycle. Once the design is firm, the actions and deliverables for System Design, Programming, and Integration and Testing stages are followed to produce the final software product.

Rapid Prototyping: Rapid prototyping can be applied to any software development methodology (e.g., segmented, spiral). Rapid prototyping is recommended for software development that is based on a new technology or evolutionary requirements. With the rapid prototyping technique, the most important and critical software requirements are defined based on current knowledge and experience. A quick design addressing those requirements is prepared, and a prototype is coded and tested. The purpose of the prototype is to gain preliminary information about the total requirements and confidence in the correctness of the design approach.

Iterative Technique: The iterative technique is normally used to develop software products piece by piece. Once the system architecture and functional or conceptual design are defined and approved, system functionality can be divided into logically related pieces. With each iterative step of the development effort, the project team performs the lifecycle stages actions and deliverables.

Rapid Application Development: Rapid Application Development (RAD) is a method for developing systems incrementally and delivering working pieces every 3 to 4 months, rather than waiting until the entire project is programmed before implementation. Over the years, many information projects failed because by the time the implementation took place, the business had changed.

Joint Application Development: Joint Application Development (JAD) is a RAD concept that involves cooperation between the designer of a computer system and the end user to develop a system that meets the user's needs exactly. It complements other system analysis and design techniques by emphasizing participative development among system owners, users, designers, and builders. During JAD sessions for system design, the system designer will take on the role of facilitator for possibly several full-day workshops intended to address different design issues and deliverables.

Object-Oriented Development: Object-oriented development focuses on the design of software components that mimic the real world. A component that adequately mimics the real world is much more likely to be used and reused. The approach emphasizes how a system operates, as opposed to analysis, which is concerned with what a system is capable of doing. One of the most important advantages in using an object-oriented approach is the ability to reuse components. Traditional practices surrounding software development often mitigate against reuse. Short term goals are stressed because today's milestones must be achieved before any thought can be given to milestones that may be months or years away. Typically, the object-oriented process follows an evolutionary spiral that starts with customer communication, where the problem is defined. The technical work associated with the process follows the iterative path of analysis, design, programming, and testing. The fundamental core concepts in object-oriented design involve the elements of classes, objects, and attributes. Understanding the definition and relationships of these elements is crucial in the application of object-oriented technologies.

Factors to be considered when choosing a life-cycle model for a specific project include the following:

- software failure impact level,
- expected size of the software product,
- intended use of the software,
- software source, and
- knowledge and experience of the project team.

Subject: Software Management Program Description
--

4.0 SOFTWARE MANAGEMENT (Continued)

4.2 Life Cycle

The life cycle used for the Y-12 Complex consists of nine major stages, with each stage consisting of one or more activities that may be further divided into tasks. Life cycle stages, activities, and tasks have deliverable work products and measurable endpoints. The nine life cycle stages are shown in Figure 2 and described in detail in Y80-101INS.

Several activities are common to each life cycle stage. The information and documentation required to perform stage activities and tasks should be available before initiating a stage. Assessments should be conducted near the end of each stage to assess the quality of the work product deliverables, security, state of development, and project status. The final activity in each life cycle stage is the Stage Exit, where the Software Owner, other project stakeholders, and project manager evaluate the deliverable work products to determine whether to close the stage and, in accordance with the Software Project Plan, continue with the current stage (s), return to a previous stage, or terminate the project. The life cycle stages, activities, tasks, and work products are described in detail in the Y80 Series.

4.3 Applicability

The methodology for the Y-12 Complex employ best industry practice project management and software processes. The software life cycle model is highly flexible because life cycle stages, activities, tasks, and related work product deliverables can be readily adapted to meet specific requirements for Y-12 Complex software projects of any size and scope. The life cycle model selected by the Project Team can be tailored to projects of any size, ranging from large enterprise-wide systems involving many people to small projects of a short duration involving a single developer. The model can be applied to manufacturing systems, business or administrative systems, technical applications, information systems, real-time systems, and research and development software projects.

4.4 Adaptability

In order to present its full scope, the complete software life cycle model is defined with discrete stages, activities, tasks, and work product deliverables. However, the life cycle model can be adapted to meet requirements for any software project. The project manager and Software Owner determine adaptations, with final approval granted by the Project Team and/or Software Review Board. Adaptations to life cycle stage, activities, or work products must be identified, described, and justified in the project plan. Stages, activities, tasks, and/or work product deliverables may be compressed or combined to meet specific project needs, especially for small projects. The life cycle can be expanded to include additional activities, tasks, or work products as needed. For example, eliminating the design and programming stages for software provided by one of the design agencies or for Commercial Off The Shelf (COTS) software might be appropriate. The functional design, system design, and/or programming stages might be combined for the rapid prototyping methodology. The programming stage might be compressed when a design tool is used to generate computer code. Life cycle model adaptations must retain the fundamental objectives of the software engineering methodology, and quality and security requirements must be met.

Subject: Software Management Program Description
--

4.0 SOFTWARE MANAGEMENT (Continued)

Some examples of life cycle adaptations include the following:

- Change the order in which life cycle stages are performed.
- Schedule stages and activities in concurrent or sequential order.
- Repeat, merge, or eliminate stages, activities, or work products.
- Include additional activities, tasks, or work products in a stage.
- Change the sequence or implementation of life cycle activities.
- Change the development schedule of the work products.
- Combine or expand activities and the timing of their execution.

5.0 IMPLEMENTATION

Implementation consists of integrating the requirements of quality assurance, security, and software engineering into an instruction that is used to manage software work processes by providing the "how to" to meet identified requirements. The following subsections describe the structure of the instruction that implements this program description.

5.1 Preparation

Software preparation consists of entering the software information into SAM and performing an evaluation to determine the deliverables that are required. An exemption process exists to determine if identified deliverables may be exempted based on identified software source and consequences of failure.

5.1.1 Evaluation

During evaluation the applicable software system information, use, source, consequences of failure, and SFI are identified, determined and documented

5.1.1 Exemption

Potential exemptions of deliverables are determined by the SFI Level of the software. During the exemption process the deliverables identified as potentials for exemption are reviewed by the Project Team (PT). If an exemption is selected, the PT documents the rationale for exemption and presents it to the Project Manager and/or the Software Review Board. If exemption is granted, the status is documented as a part of the project file.

5.2 Life Cycle Stages

A lifecycle model is used that is based on the Department of Energy software engineering methodology. This model partitions the software engineering lifecycle into nine stages. Each stage is divided into activities and tasks, and has a measurable end point, the Stage Exit. The execution of all nine stages is based on the premise that the quality and success of a software product depends on a feasible concept, highly visible project planning, commitments to resources and schedules, complete and accurate requirements, a sound design, consistent and maintainable programming techniques, and a comprehensive testing program. The lifecycle stages and activities are described below.

5.2.1 Planning

During planning, a problem that can potentially be solved by means of a software project is identified or an opportunity for increasing efficiency or productivity via a software project is identified. The project's purpose, objectives, scope, and constraints are described in enough detail to plan the project. An initial project schedule and budgetary plan are developed for the project. An overall life cycle model is selected.

Subject: Software Management Program Description
--

5.0 IMPLEMENTATION (continued)

5.2.2 Requirements Definition

During requirements definition, detailed requirements for the proposed system are gathered and alternative solutions (e.g., enhance existing application, procure COTS application, develop in-house application, etc.) are identified, along with possible host platforms for the application. Software tools, languages, DBMS, and related required infrastructure needs are also identified. Procurement activities are initiated for those items that must be obtained from external suppliers.

5.2.3 Functional Design

During functional design, requirements for the proposed system are expanded into functional specifications and models, and one of the alternatives identified during the requirements definition stage is selected. The functional specifications contain the business rules for the project mapped into functional components.

5.2.4 System Design

During system design, functional specifications are translated into technical, computer-oriented specifications. The design specifications are in sufficient detail to plan and execute the programming, software integration and testing, and installation stages.

5.2.5 Programming

During programming, the system is coded, and individual components are tested, and the Release Version identification initiated. Documentation related to software maintenance is completed, along with material related to user training and user documentation.

5.2.6 Software Integration and Testing

During software integration and testing the system is fully integrated and system testing is conducted to validate that the software will operate in its intended environment, satisfies user requirements, and is supported with complete and accurate operating documentation.

5.2.7 Software Installation and Acceptance

During software installation and acceptance, the system is installed and acceptance tested. These tests ensure that the system meets design requirements and is acceptable to the Software Owner. Once the software is installed and accepted, the system is placed in the Maintenance Stage.

5.2.8 Software Maintenance

During software maintenance, the system is maintained for production use. Configuration Management is formally established for the project by the PM or SO. Changes and enhancements are managed. During this stage, the Release Version identification is incremented and applied. Software maintenance is performed until the system is retired.

5.2.9 Retirement

During retirement, the application and associated data are reviewed for closure action such as storage, destruction, repository placement, future reading and utilization. The Project Team identifies the retention and disposition actions required to address the software system, documentation, and resulting data files.

Subject: Software Management Program Description
--

6.0 ORGANIZATIONAL INTERFACES

Several key organizational interfaces may become involved in a software project. These organizations provide technical direction and oversight or fulfill a key management role.

6.1 Computing and Telecommunications Security Organization (CTSO)

The CTSO provides leadership in the protection of computing and telecommunications hardware, software, and data processed on Automated Information System (AIS) resources. Policy, standards, and requirements related to computer security are developed by CTSO. A member of CTSO is named as a permanent member of the Software Review Board. A CTSO member is named to each Software Project Team, or the Lead Analyst may be appointed as the CTSO Representative by CTSO.

6.2 Corporate Information Office (CIO)

The CIO is responsible for setting Y-12 Complex-wide strategy for database, server, desktop, and network computing. This responsibility includes development of policies, standards, and requirements in these areas, consistent with DOE guidance and good business practice, to ensure cost-effective management of the overall Y-12 Complex information technology resource base.

6.3 Quality Organization

The Quality organization will ensure that current Quality Assurance (QA) requirements are adequately established in the Software Management Program and will provide oversight of implementation of the program. The Quality organization appoints a Quality Engineer/Specialist to serve on the Software Review Board to ensure that software within the Y-12 Complex meets quality requirements. Also, the Quality organization will schedule and perform assessments to ensure that the QA requirements are adequately implemented throughout the Y-12 Complex. The Manager of Quality Programs is responsible for interpreting this program and providing guidance. A Quality Engineer is named to serve on each Software Project Team.

6.4 Product Engineering

The Product Engineering organization ensures that appropriate SQA/SM activities are performed and product related software is managed consistent with QC-1 requirements. A Product Engineer is named as a permanent member of the Software Review Board. A Product Engineer is named to serve on each Software Project Team dealing with Product Software.

7.0 ROLES AND RESPONSIBILITIES

7.1 Boards and Teams

Boards and teams associated with the Software Management Program consist of the Software Review Board (SRB) and the Software Project Team.

7.1.1 Software Review Board (SRB)

The SRB has responsibility for reviewing, approving or rejecting sample selected or submitted Project Plans, and reviewing, granting or rejecting exemption requests. The SRB also reviews the assigned failure impact for software projects and reassigns as necessary. SRB membership is defined in the SRB Charter, Appendix B of Y80-101INS.

Subject: Software Management Program Description
--

7.0 ROLES AND RESPONSIBILITIES (Continued)

7.1.2 Software Project Team

The Software Project Team is responsible for the definition, design, and development or acquisition of an application software project. This team is responsible for the development of the project from the planning stage through Maintenance Stage. The team consists of the Software Owner, User Point of Contact (POC), Project Manager, Lead Analyst, one or more Developers, CTSO Representative, QE, and PE (if product related).

7.2 Individuals

Individual roles shall be designated as needed on a project, depending on the size and scope of that project. In some cases, multiple roles may be assigned to the same person. The following sections describe a complete set of roles that should be considered for all projects. In some cases, new or different roles may be required for a specialized project.

7.2.1 Software Owner

The Software Owner initiates the request for a project. The User POC may also be the Software Owner. The Software Owner is responsible for the functional characteristics of a software project and its use within the plant. This responsibility includes evaluating and obtaining approvals to proposed software enhancements, and executing authorized changes. This role is usually filled by a person in the organization who will also be responsible for the funding of the software project. The Software Owner is responsible for proper execution of all activities contained within Y80-101INS appropriate to the specific software project. The Software Owner has the authority to initiate, cancel, retire, or postpone a software project. The SO ensures that the PT has the necessary expertise and training.

7.2.2 User Point of Contact (POC)

The User POC represents the needs and interests of the users of the system and ensures that the system requirements are defined and met. The User POC has ultimate responsibility for the preparation of a User Acceptance Test Plan during the installation and acceptance stage of the project. During the maintenance stage, the User POC requests changes to an software project, participates in acceptance testing of new or revised software, and accepts or rejects the software for production use.

7.2.3 Project Manager

The Project Manager provides overall leadership of the project. The Project Manager is responsible for preparing and maintaining the Project Plan from the Planning Stage through to the Maintenance Stage. The PM is accountable for proper execution of all activities contained within Y80-101INS appropriate to the specific software project. The Project Manager may be selected from any participating organization on the basis of his or her knowledge of the subject area, leadership skills, etc.

7.2.4 Lead Analyst

The Lead Analyst is responsible for the technical design and implementation of the software project. If no additional programmers/analysts are assigned to the project, the Lead Analyst performs the functions of the programmers/analysts. The Lead Analyst is the CTSO Representative unless otherwise specified by CTSO.

Subject: Software Management Program Description
--

7.0 ROLES AND RESPONSIBILITIES (Continued)

7.2.5 Software Program Manager

The Software Program Manager (1) assists plant personnel in the implementation of software requirements, (2) evaluates the effectiveness of software activities, (3) represents the plant in software matters involving external organizations, and (4) supervises the plant software training and awareness programs. The Program Manager is responsible for evaluating alternative SQA programs against Y80-101INS for acceptability.

7.2.6 Product Engineer (PE)

The PE has oversight responsibility for the weapons program with which the software is associated. The Product Engineering Manager designates the PE for software used on multiple programs. The PE may reside in another organization. The PE is also responsible for assuring that software meets established quality requirements and assists in the determination of whether software is to be categorized as product software. The PE serves on the SRB.

7.2.7 Quality Engineer (QE)

The QE is responsible for assuring that product software meets established quality requirements and assists in the determination of whether software is to be categorized as product software. For non-weapon related activities, a Quality Engineer with facility quality assurance responsibilities participates as the QE representative to projects. The QE serves on the SRB.

7.2.8 Computing and Telecommunications Security Organization (CTSO) Representative

The CTSO Representative represents the computer security organization throughout the life cycle(s) of a protected software project. The CTSO Representative or the CTSO member recommends the certification of a protected software project for production status to the certifying authority. The certifying authority is the software owner for applications residing on unclassified processors or the Information Security Officer (ISSO) for applications residing on classified processors.

7.2.9 Information System Security Officer (ISSO)

The ISSO is an individual assigned the responsibility for ensuring the implementation of and compliance with all policies, standards, and procedures for the AIS Security Program for a single classified or Unclassified National Security Related (UNSR) processor or group of classified or UNSR processors. This individual certifies protected software projects that reside on a classified processor.

8.0 IMPLEMENTING DOCUMENTS

The implementing document for software management is Y80-101INS, "Software Management Instruction."

9.0 SOURCE DOCUMENTS

- BWXT Y-12 Standards/Requirements Identification Document (S/RID), Requirement Units: BWXT Y-12 Identification Documents (ESID) 9940
- Y60-101PD, *Quality Program Description*
- QC-1, DOE-AL, *Quality Criteria*
- DOE Order 471.2A, *Information Security Program*

Subject: Software Management Program Description
--

10.0 REFERENCES

- Y19-401INS, *AIS Security Handbook*
- DOE M 471.2-2, *Classified Information Systems Security Manual*
- DOE N 205.1, *Unclassified Cyber Security Program*
- Office of Management and Budget Circular A-130 Appendix III
- DOE G 200.1-1, *U.S. Department of Energy (DOE) Software Engineering Methodology*, March 1999
- DOE 203.1, *Software Quality Assurance*

11.0 DOCUMENTATION AND RECORDS

The amount, type, and content of the project and system documentation should be tailored to meet project needs. Project documentation should be at a level to ensure the project is properly managed. The system documentation—including user training material and instructions—should be at a level to ensure full system operability, usability, and maintainability. No requirement exists that each work product document be a formal, stand-alone document. Stage documentation requirements may be combined to meet project needs.

Examples of acceptable project documents include the following:

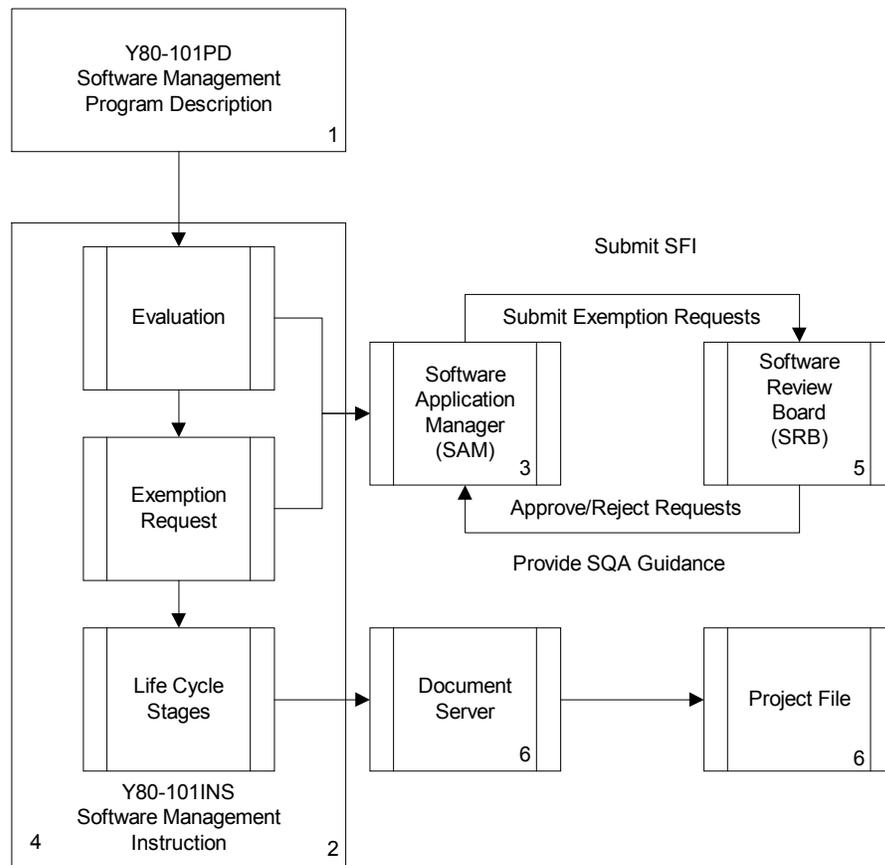
- formal documents;
- identified life cycle stage deliverables;
- electronic documents, including documentation in software tool sets;
- forms, including electronic forms;
- reports;
- electronic mail messages;
- minutes and notes from team meetings;
- hand-written notes; and
- computer code.

Draft documents in one life cycle stage may be expanded or updated in a subsequent stage. The Project Manager and Software Owner should determine the project documentation requirements, and they should be described in the project plan. Examples of factors to consider when determining the project documentation requirements include (1) software failure impact level, (2) product software, (3) protected software, (4), intended use of the software, (5) software source, (6) software development tools to be used, and (7) any specific project requirements.

Subject: Software Management Program Description

Figure 1

Software Management Program Diagram



Notes:

- 1. Y80-101PD provides information on what is to be done to provide Software Management.
- 2. Y80-101INS provides information on how to perform Software Management.
- 3. The Software Application Manager(SAM) implements Y80-101PD and Y80-101PD.
- 4. The SPT serves as the management tool .
- 4. The SPT implements the PD and INS to establish and execute a Software Project.
- 5. The SRB reviews SFI and Exemption submissions and provides guidance to SPTs.
- 6. The SPT utilizes the Documentation Server to maintain SP documentation and Project File.

Subject: Software Management Program Description

Figure 2

Life Cycle Stage Diagram

