Software Engineering Methodology

Chapter 10.0
Software Maintenance

Table of Contents

Chapter: **10.0**

Software Maintenance

Description: This chapter describes an iterative process for conducting software maintenance activities. The process prescribes a minimal set of criteria that are necessary for project management and quality assurance processes; control; and management of the planning, execution, and documentation of software maintenance activities. The use of automated tools to facilitate requirements definition, design, coding, and system documentation is encouraged. The selection and implementation of tools varies among sites and organizations.

The basic maintenance process model includes input, process, output, and control for software maintenance. It is based on the same software engineering principles and preferred practices that lower risk and improve quality during the planning and development stages of the lifecycle. The process model supports the concept that planned software changes should be grouped and packaged into scheduled releases that can be managed as projects. This proven approach allows the maintenance team to better plan, optimize use of resources, take advantage of economies of scale, and better control outcome in terms of both schedule and product quality.

Each organization performing software maintenance activities should have a local documented procedure for handling emergency changes that cannot be implemented as part of a scheduled release. Generally, these changes include fixes to correct defects and updates to meet unscheduled business or legal requirements. Emergency changes should be integrated into the next release for full regression testing and documentation updates.

Stages: The activities to be performed during software maintenance are grouped into logically related segments of work called "stages." These stages are similar to those referenced in the planning and development stages of the software lifecycle. The stages are presented in the sections listed below.

10.1 Problem/Modification Identification Stage
10.2 Analysis Stage
10.3 Design Stage
10.4 Programming Stage
10.5 System Test Stage
10.6 Acceptance Stage
10.7 Delivery Stage

A matrix depicting the maintenance process model stages is provided in *Exhibit 10.0-1, Process Model for Software Maintenance.*

Note: The maintenance process model does not presume the use of any particular software development methodology (e.g., waterfall or spiral). This process is valid regardless of size, complexity, criticality, application of the software product, or usage of the software in the system to be maintained.

The software maintenance stages can be tailored (i.e., logically combining stages or outputs) as appropriate. Stages may be combined to more effectively manage the project. Decisions to combine stages are agreed to by the designated approvers during the Analysis Stage. Factors that can influence the number of stages include level of effort, complexity, visibility, and business impact. Guidance to assist with decisions to combine stages is presented in *Exhibit 10.0-2, Tailoring For Size*.

Project Management: To the extent possible, all software maintenance activity should be managed as a project to gain the benefits inherent in project management and to enable tracking of activities and costs. The extent of project management activity will vary, and should be tailored according to the size, complexity, and impact of the change or enhancement.

Review Processes: In each stage, one or more structured walkthroughs are conducted to validate work products. *Appendix C, Conducting Structured Walkthroughs,* provides a procedure and sample forms that can be used for structured walkthroughs.

In software maintenance, and especially for major modifications, one or more In-Stage Assessments are conducted as part of the quality assurance activities for each stage. This process is documented in *Appendix D, In-Stage Assessment (ISA) Process Guide*.

A Stage Exit is conducted at the end of each stage of software maintenance. This process, which includes definition of participant roles and the review and approval process, is documented in *Appendix E, Stage Exit Process Guide*.

Metrics: Metrics/measures and associated factors for each stage should be collected and reviewed at appropriate intervals. *Exhibit 10.0-3, Process Model Metrics for Software Maintenance,* provides metrics for each stage of software maintenance. Metrics/measures captured for maintenance should enhance the implementation and management of this process.

Conventions: The following convention is used in each exhibit depicting a software maintenance stage.

```
                      Control
                         T
                         *
                         ▾
                   +)))))))))))),
    Input S))))▸ *Process Name* S)))▸ Output
                   .))))))))))))-
                         T
                         *

                 Associated Process
```

An "associated process" is one that is executed in support of software maintenance, but is itself not defined in this chapter (e.g., Stage Exit).

Exhibit 10.0-1. Process Model for Software Maintenance

	Problem Identification Stage	Analysis Stage	Design Stage	Programming Stage	System Test Stage	Acceptance Stage	Delivery Stage
Input	Modification Request (MR)	Project/system document Project file information Validated MR	Project/system document Source code Database(s) Analysis stage output	Source code Product/system document Results of design stage	Updated software Documentation Test Readiness Review Report Updated system	Test Readiness Review Report Integrated system Acceptance test: Plans Cases Procedures	Tested/accepted system
Process	Assign change number Classify Accept or reject change Preliminary effort estimate	Feasibility analysis Detailed analysis Redocument, if needed	Revise: Requirements System doc. Module doc. Project plan Create test cases	Code Unit test Test Readiness Review	Functional test Interface testing Regression testing Test Readiness Review	Acceptance test Interoperability test Functional Configuration Audit (FCA)	Physical Configuration Audit (PCA) Install Training
Output	Validated MR Process determinations	Feasibility Report Detailed Analysis Report Updated: Requirements Modification list Test strategy Project plan	Revised: Modification list Detailed analysis Updated: Design baseline Test plans Project plan	Updated: Software Design documents Test documents User documents Training materials Project plan Test readiness review report	Tested system Test reports Updated project plan	New system baseline Acceptance Test Report FCA Report Updated project plan	PCA Report Version Description Document (VDD)
Review Assurance Approve	Peer review(s)	Structured Walkthrough(s) In-Stage Assessment(s) Stage Exit	Structured Walkthrough(s) In-Stage Assessment(s) Stage Exit	Structured Walkthrough(s) In-Stage Assessment(s) Stage Exit	Structured Walkthrough(s) In-Stage Assessment(s) Stage Exit	Structured Walkthrough(s) In-Stage Assessment(s) Stage Exit	Structured Walkthrough Stage Exit
Metrics	*See Exhibit 10.0-3, Process Model Metrics for Software Maintenance*						

Exhibit 10.0-2. Tailoring For Size

```
                                SOFTWARE MAINTENANCE STAGES
 SIZE[1]
                     ▼        ▼        ▼        ▼        ▼        ▼        ▼

            *        *        *        *        *        *        *        *
LARGE       /))))))))3))))))))3))))))))3))))))))3))))))))3))))))))3))))))))1
            *Prob. ID*Analysis*Design  *Program.*Sys.Tst.*Accept. *Delivery*


                     ▼             ▼                ▼             ▼

            *        *             *                *             *
MEDIUM      /))))))))3)))))))))))))3))))))))))))))))3)))))))))))))1
            *Prob. ID*Anal./Design *Program./Test   *Accep./Deliv.*

                                 ▼                       ▼

            *                    *                       *
SMALL       /))))))))0)))))))))))3)))))))))))))0)))))))))1
            *Prob. ID*Anal./Desi.*Program/Test *Acc./Del.*
            R        R           R             R         R


           LESS )))))))))))))))))))))))))))))))))))))))))))))))))))))))Q‣ MORE
                      DEGREE OF PROJECT MANAGEMENT REQUIRED
```

▼ = Stage Exit occurs at this point.

[1] Size is used as a guide to help determine the appropriate degree of project management, and which stages may be combined for a given effort. Within this context, size is a combination of level of effort required (all activities) and complexity of the modification. Attributes of complexity include technology, team skills, interfaces, and level of understanding of requirements.
Other factors that can influence tailoring include risk, visibility, and business impact.

Exhibit 10.0-3. Process Model Metrics for Software Maintenance

	Problem Identification Stage	Analysis Stage	Design Stage	Programming Stage	System Test Stage	Acceptance Stage	Delivery Stage
Factors	Correctness Maintainability	Flexibility Traceability Usability Reusability Maintainability Comprehensibility	Flexibility Traceability Reusability Testability Maintainability Comprehensibility Reliability	Flexibility Traceability Maintainability Comprehensibility Reliability	Flexibility Traceability Verifiability Testability Interoperability Comprehensibility Reliability	Flexibility Traceability Interoperability Testability Comprehensibility Reliability	Completeness Reliability
Metrics	No. of omissions on Modification Request (MR) No. of MR submittals No. of duplicate MRs Time expended for problem validation	Requirement changes Documentation error rates Effort per function area (e.g., SQA) Elapsed time (schedule) Error rates, by priority and type	S/W complexity Design changes Effort per function area Elapsed time Test plans and procedure changes Error rates, by priority and type Number of lines of code, added, deleted, modified, tested	Volume/ functionality (function points or lines of code) Error rates, by priority and type	Error rates, by priority and type Generated Corrected	Error rates, by priority and type Generated Corrected	Documentation changes (i.e. version description documents, training manuals, operation guidelines)

Note: The above level of metrics is a goal. Each organization responsible for software maintenance activities should establish an individual plan to achieve this level over time.

Bibliography: The following materials were used in the preparation of the Software Maintenance chapter.

1. DeMarco, Tom, *Controlling Software Projects*, New York, 1989.

2. Frame, Davidson J., *Managing Projects in Organizations*, San Francisco, 1987.

3. Frame, Davidson J., *The New Project Management*, San Francisco, 1994.

4. Page-Jones, Meilir, *Practical Project Management*, New York, 1985.

5. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Software Maintenance*, IEEE Std 1219-1992, New York, 1993.

6. U.S. Department of Commerce/National Bureau of Standards, *Guideline on Software Maintenance*, Springfield, Virginia, 1984.

Stage: **10.1**
Problem/Modification Identification Stage

Responsibility: Maintenance Team

Description: In this stage, software changes are identified, classified, and assigned an initial priority ranking. Each request for a software modification (i.e., Modification Request) is evaluated to determine its classification and handling priority. The classification should be identified from the following types of maintenance.

- Corrective - Change to a software product after delivery to correct defects.
- Adaptive - Change to a software product after delivery to keep it functioning properly in a changed or changing environment.
- Emergency - Unscheduled corrective maintenance required to keep a system operational.

The need for software modifications can be driven by any number of factors, including:

- Report of system malfunction
- Mandatory changes required by new or changed federal or state law
- New requirements to support business needs
- Major enhancement or redesign to improve functionality or replace an obsolete system component
- Operational system upgrades and new versions of resident software (e.g., COBOL, CICS, Oracle)

These factors should be considered when assigning a priority to the modification request.

Exhibit 10.1-1 (provided at the end of this section) summarizes the input, process, control, and output for the Problem/Modification Identification Stage of software maintenance.

Input: Input to the Problem/Modification Identification Stage of software maintenance is one or more Modification Requests.

Process: If a modification to the software is required, the following activities must occur within the maintenance process:

- Assign an identification number

- Classify the type of maintenance

- Analyze the modification to determine whether to accept, reject, or further evaluate

- Prioritize the modification according to the following categories:

  - Emergency (follow emergency change procedure and integrate into the next scheduled release or block of modifications)
  - Mandatory (e.g., legal, safety, payroll)
  - Required (has associated benefits; e.g., productivity gains, new business drivers)
  - Nice to have (lower priority)

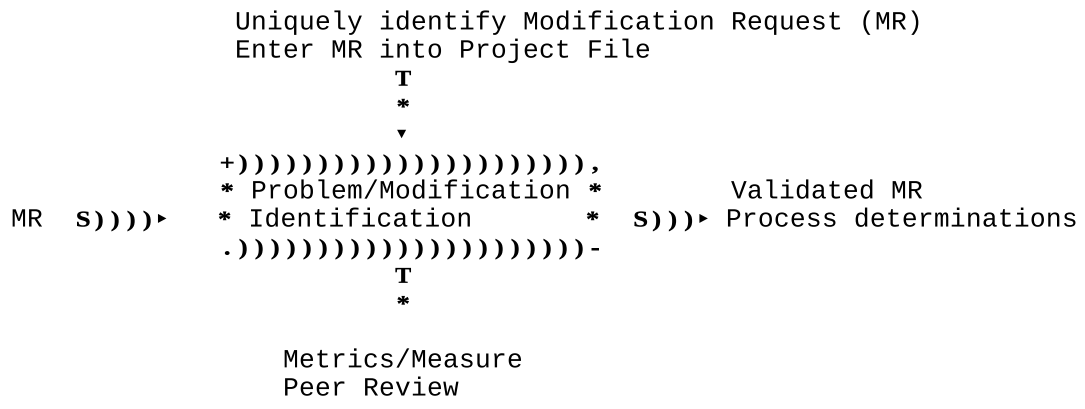
Control: Modification Requests and process determinations are uniquely identified and entered into the Project File.

Work Products: The output of this stage is the validated Modification Request and the following process determinations. Place a copy of all work products in the Project File.

- Statement of the problem or new requirement
- Problem or requirement evaluation
- Classification of the type of maintenance required
- Initial priority
- Verification data (for corrective modifications)
- Initial estimate of resources required

Review Process: Conduct peer review(s) as appropriate.

Exhibit 10.1-1. Problem/Modification Identification Stage


Uniquely identify Modification Request (MR)
Enter MR into Project File
MR
Problem/Modification Identification
Validated MR
Process determinations
Metrics/Measure
Peer Review

Stage: **10.2**
Analysis Stage

Responsibility: Project Team and System Owner

Description: During the Analysis Stage, the Project File information, the Modification Request(s) validated in the Problem/Modification Identification Stage, and the system and project documentation are used to study the feasibility and scope of the modification, and to develop a preliminary Project Plan for design, test, and delivery.

If the documentation is not available or is insufficient and the source code is the only reliable representation of the software system, reverse engineering is recommended to ensure the overall integrity of the system. In those cases where long-lived systems have overgrown the initial base system and have poorly updated documentation, reverse engineering may be required and would evolve through the following steps:

For a smaller scope, or for local analysis on a unit level:

- Dissection of source code into formal units
- Semantic description of formal units and declaration of functional units
- Creation of input/output schematics of units

For a larger scope, or for global analysis on a system level:

- Declaration and semantic description of linear flows
- Declaration and semantic description of system applications (functions grouped)
- Creation of anatomy of the system (system architecture)

Modifications of a similar nature (i.e., affecting the same program(s)) should be grouped together whenever possible, and packaged into releases that are managed as projects. A release cycle should be established and published.

Exhibit 10.2-1 (provided at the end of this section) summarizes the input, process, control, and output for the Analysis Stage.

Input: Input to the Analysis Stage of software maintenance includes the following:

- Validated Modification Request
- Initial resource estimate and associated information
- Project and system documentation, if available

Process: Preliminary analysis activities include the following:

- Make a preliminary estimate of the modification size/magnitude

- Assess the impact of the modification

- Assign the Modification Request to a block of modifications scheduled for implementation

- Coordinate the modifications with other ongoing maintenance tasks

Once modifications are agreed to, grouped if appropriate, and packaged, analysis progresses and includes the following:

- Define firm requirements for the modification

- Identify elements of the modification

- Identify safety and security issues

- Devise a test and implementation strategy

In identifying the elements of the modification (creating the preliminary modification list), examine all work products (e.g., software, specifications, data bases, and documentation) that are affected. Each work product is identified, and generated, if necessary, specifying the portion of the product to be modified, the interfaces affected, the user-noticeable changes expected, the relative degree and kind of experience required to make changes, and the estimated time to complete the modification.

The test strategy is based on input from the previous activity identifying the elements of modification. Requirements for at least three levels of testing, including individual unit tests, integration tests, and user-oriented functional tests are defined. Regression test requirements associated with each of these levels of testing are identified as well. The test cases to be used for testing to establish the test baseline are revalidated.

Control: Control of the Analysis Stage activities includes the following:

- Retrieval of the relevant version of project and system documentation from the configuration control function of the organization.
- Review of the proposed changes and engineering analysis to assess technical and economic feasibility, and correctness.
- Identification of safety and security issues.
- Consideration of the integration of the proposed change within the existing software.
- Verification that all appropriate analysis and project documentation is updated and properly controlled.
- Verification that the test function of the organization is providing a strategy for testing the change(s), and that the change schedule can support the proposed test strategy.
- Review of resource estimates and schedules; verification of accuracy.
- Technical review to select the problem reports and proposed enhancements to be implemented in the new release.

Work Products: The output of the Analysis Stage includes the following:

- Feasibility report for modification requests
- Detailed analysis report
- Updated requirements (including traceability list)
- Test strategy
- Project Plan

A written assessment, generally called a Feasibility Report, is prepared and contains the following:

- Short and long term costs
- The value of the benefit of making the modification (usually provided by the system owner)
- Solution approach, including prototyping if applicable

Work Products, continued:

- Safety and security implications
- Human factors

A project plan states how the design, implementation, testing, and delivery of the modification is to be accomplished with a minimal impact to current users.

Review Processes: Conduct structured walkthrough(s), In-Stage Assessment(s), and a Stage Exit.

At the end of the Analysis Stage, a risk analysis is performed. Using the output of the Analysis Stage, the preliminary resource estimate is revised, and a decision is made on whether to proceed to the Design Stage.

Exhibit 10.2-1. Analysis Stage

```
                         Conduct technical review
                         Verify that documentation is updated
                         Verify test strategy
                         Identify safety and security issues
                                      T
                                      *
                                      ▾              Feasibility report
Validated Mod. Request         +))))))))))))),       Detailed analysis report
Project/system document S))))‣ *  Analysis  * S)))‣  Updated requirements
Project File information       .)))))))))))))-       Modification list
                                      T              Test strategy
                                      *              Project Plan
                                      R
                              Metrics/measures
                              Structured walkthrough
                              In-Stage Assessment
                              Stage Exit
```

Stage: **10.3 Design Stage**

Responsibility: Project Team

Description: In the Design Stage, all current system and project documentation, existing software and data bases, and the output of the Analysis Stage are used to design the modification to the system. *Exhibit 10.3-1* (provided at the end of this section) summarizes the input, process, and output for the Design Stage of software maintenance.

Input: Input to the Design Stage of software maintenance includes the following:

- Analysis Stage output, including:
  - Detailed analysis
  - Updated statement of requirements
  - Preliminary modification list (identification of affected elements)
  - Test strategy
  - Project Plan

- System and project documentation

- Existing source code, comments, and data bases

Process: The process steps for the Design Stage include the following:

- Identify selected software modules

- Modify software module documentation (e.g., data and control flow diagrams, schematics)

- Create test cases for the new design, including safety and security issues

- Identify/create regression tests

- Identify documentation (system/user) update requirements

- Update modification list

- Document any known constraints that influence the design, and any risks that have been identified. Where possible, actions, taken or recommended, that mitigate risk should also be documented.

Control: The following control activities are performed during the Design Stage of a modification.

- Conduct structured walkthrough(s) of the design
- Verify that the new design/requirement is documented as an authorized change
- Verify the inclusion of new design material, including safety and security issues
- Verify that the appropriate test documentation has been updated
- Complete the traceability of the requirements to the design

Work Products: The output of the Design Stage of software maintenance includes the following:

- Revised modification list
- Updated design baseline
- Updated test plans
- Revised detailed analysis
- Verified requirements
- Updated Project Plan
- Documented constraints and risks

Review Process: Conduct structured walkthrough(s), In-Stage Assessment(s), and a Stage Exit.

Exhibit 10.3-1. Design Stage

```
                          Conduct structured walkthrough(s)
                          Verify that design is documented
                          Ensure traceability of requirements to design
                                       T
                                       *
                                       ▾              Revised modification list
      System/project document        +))))))))))))),       Updated design baseline
      Analysis Stage output   S))))▸  *   Design   * S)))▸ Updated test plans
      Source code, data base         .)))))))))))))-       Validated requirements
                                       T              Updated Project Plan
                                       *              Constraints and risks
                                       R
                               Metrics/measures
                               Structured walkthrough(s)
                               In-Stage Assessment(s)
                               Stage Exit
```

Stage: **10.4**
Programming Stage

Responsibility: Project Team

Description: In the Programming Stage, the results of the Design Stage, the current source code, the project and system documentation, (i.e., the entire system as updated by the prior stages) is used to drive the programming effort. *Exhibit 10.4-1* (provided at the end of the section) summarizes the input, control, and output for the Programming stage.

Input: Input to the Programming Stage of software maintenance includes the following:

- Results of the Design Stage
- Current source code, comments, and data bases
- Project and system documentation

Process: The Programming Stage includes the following tasks, which may be conducted in an incremental, iterative approach:

- Coding and unit testing
- Integration
- Revisit project risk
- Test readiness review

Coding and Unit Testing: Implement the change into the code and perform unit testing. Other quality assurance and verification and validation processes may be required for safety-related code. The Quality Assurance Team can help with specific issues.

Integration: After the modifications are coded and unit tested, or at appropriate intervals during coding, the modified software is integrated with the system, and integration and regression tests are refined and performed. All effects (e.g., functional, performance, usability, safety) of the modification on the existing system are assessed and noted. A return to the coding and unit testing tasks is made to remove any unacceptable impacts.

Risk Analysis and Review: Risk analysis and review are performed periodically during the Programming Stage rather than at the end, as in the Design and Analysis Stages. Metrics/measurement data should be used to quantify risk analysis.

Test Readiness Review: To assess the team's preparedness to enter system testing, a Test Readiness Review is conducted. This is a self assessment to determine if items including code, documentation, libraries, hardware, telecommunication lines, and schedules are ready for system test to begin on the scheduled date.

Control: Control of the Programming Stage includes the following activities:

- Conduct structured walkthroughs of the code
- Ensure that unit and integration testing are performed and documented in the Project File
- Ensure that test documentation (e.g., test plans, test cases, and test procedures) are either updated or created
- Identify, document, and resolve any risks exposed during software and test readiness reviews
- Verify that the new software is placed under software configuration management control
- Verify that the training and technical documentation have been updated
- Verify the traceability of the design to the code

Work Products: The output of the Programming Stage includes the following:

- Updated software
- Updated design documentation
- Updated test documentation
- Updated user documentation
- Updated training material
- Statement of risk and impact to users
- Test Readiness Review report

Review Process: Conduct structured walkthrough(s), In-Stage Assessment(s), and a Stage Exit.

Exhibit 10.4-1. Programming Stage

```
                         Conduct structured walkthrough
                         Ensure testing performed and documented
                         Verify:
                           New software placed under configuration management
                           Documentation has been updated
                           Traceability of design to code
                                      T
                                      *
Results of Design Stage               ▾              Updated:
Source code                    +)))))))))))),         Software
Project documentation   S))))▸ *Programming * S)))▸   Design documents
System documentation           .))))))))))))-         Test documents
                                      T               User documents
                                      *               Training material
                                      R               Project Plan
                                                     Test Readiness Rev. rpt.
                              Metrics/measures
                              Structured walkthrough(s)
                              In-Stage Assessment(s)
                              Stage Exit
```

Stage: **10.5**
System Test Stage

Responsibility: Independent Tester(s) or Project Team

Description: System testing is performed on the modified system. Regression testing is a part of system testing and is performed to validate that the modified code does not introduce faults that did not exist prior to the maintenance activity. *Exhibit 10.5-1* (provided at the end of the section) summarizes the input, process, control, and output for the System Test Stage.

Input: Input to the System Test Stage of software maintenance includes the following:

- Test Readiness Review report

- Documentation, which includes:
  - System test plans(s)
  - System test cases
  - System test procedures
  - User manuals
  - Design

- Updated system

- Updated Project Plan

Process: System tests are conducted on a fully integrated system. Testing shall include the performance of:

- System functional test
- Interface testing
- Regression testing
- Test readiness review to assess preparedness for acceptance testing

Results of tests conducted prior to the test readiness review should not be used as part of the system test report to substantiate requirements at the system level. This is necessary to assure that the test organization does not consider that testing all parts (one at a time) of the system constitutes a "system test."

Control: System tests should be conducted by an independent party for maximum results. Prior to the completion of system testing, the test function is responsible for reporting the status of the activities that had been established in the test plan for

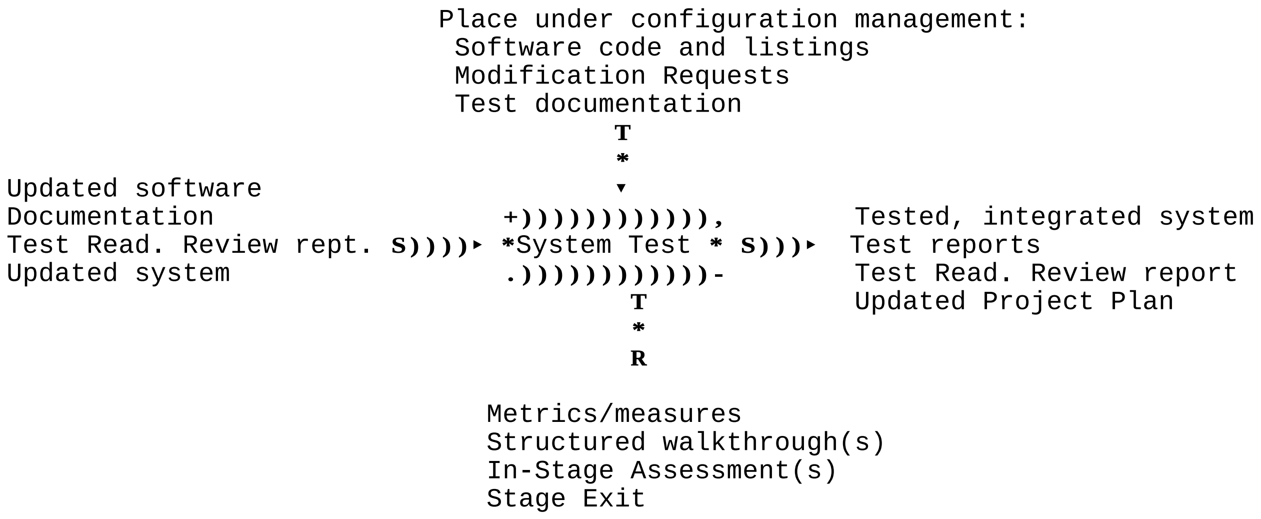
Control, continued: satisfactory completion of system testing. The status is reported to the appropriate reviewers prior to proceeding to acceptance testing. Software code listings, Modification Requests, and test documentation are placed under configuration management. The system owner shall participate in the review to ascertain that the maintenance release is ready to begin acceptance testing.

Work Products: The output for the System Test Stage of software maintenance includes the following:

- Tested and fully integrated system
- Test report
- Test Readiness Review report
- Updated Project Plan

Review Process: Conduct structured walkthrough(s), In-Stage Assessment(s), and a Stage Exit.

Exhibit 10.5-1. System Test Stage



Stage: **10.6**
Acceptance Stage

Responsibility: System Owner/User or other designated individuals

Description: Acceptance tests are conducted on a fully integrated system. Acceptance tests are performed by either the system owner, the user of the modification package, or a third party designated by the system owner. An acceptance test is conducted on the modified system, with software that is under software configuration management in accordance with the application's Software Configuration Management Plan. *Exhibit 10.6-1* (provided at the end of this section) summarizes the input, process, control, and output for the Acceptance Stage.

Input: Input for the Acceptance Stage of software maintenance includes the following:

- Test Readiness Review report
- Fully integrated system
- Acceptance Test Plan
- Acceptance test cases
- Acceptance test procedures

Process: The following steps form the process for acceptance testing:

- Perform acceptance tests at the functional level

- Perform interoperability testing (to validate the functionality of any input and output interfaces)

- Perform regression testing

- Conduct a Functional Configuration Audit (FCA)

The purpose of a FCA is to verify that all requirements specified and agreed to have been met. The FCA compares the system's software elements (programs/modules) to the software requirements documented in the current version of the Software Requirements Specification to assure that the modification addresses all, and only, those requirements. The results of the FCA should be documented, identifying all discrepancies found, and the plans for their resolution.

Control: Control of acceptance tests includes the following:

- Execute acceptance tests
- Report results for the Functional Configuration Audit conducted to ensure that all of the functionality that has been agreed to is in fact present in the system
- Establish the new system baseline
- Place the acceptance test documentation under software configuration management control

Work Products: The output of the Acceptance Stage includes the following:

- New system baseline
- Functional Configuration Audit Report
- Acceptance Test Report
- Updated Project Plan

Review Processes: Conduct structured walkthrough(s), In-Stage Assessment(s), and a Stage Exit.

Exhibit 10.6-1. Acceptance Stage

```
                          Execute acceptance tests
                          Report test results
                          Conduct functional audit
                          Establish new baseline
                            Acceptance test documentation under configuration
      management
                                         T
                                         *
Test Readiness Review report             ▾
Fully integrated system        +))))))))))))),      New system baseline
Acceptance Test Plan    S))))▸ *Accept. Test* S)))▸  Functional configuration
Acceptance test cases          .))))))))))))-        audit report
Acceptance test procedures               T          Acceptance test report
                                         *          Updated Project Plan
                                         R

                             Metrics/measures
                             Structured walkthrough(s)
                             In-Stage Assessment(s)
                             Stage Exit
```

Stage: **10.7**
Delivery Stage

Responsibility: Project Team

Description: This stage describes the requirements for the delivery of a modified software system. *Exhibit 10.7-1* (provided at the end of the section) summarizes the input, process, control, and output for the Delivery Stage.

Input: Input to the Delivery Stage of software maintenance is the fully tested version of the system as represented in the new baseline.

Process: The tasks for delivery of a modified system include the following:

- Conduct a Physical Configuration Audit (PCA).
- Notify the user community.
- Develop an archival version of the system for backup.
- Perform installation and training at the user facility.

The purpose of a PCA is to verify that the software associated with the modification and its documentation are internally consistent and are ready for delivery. The PCA compares the software components (programs/modules) with its supporting documentation to assure that the documentation to be delivered correctly describes the system components. All discrepancies noted during the PCA, along with plans for their resolution should be documented.

Control: Control for the Delivery Stage includes the following:

- Arrange and document a Physical Configuration Audit
- Provide access to system materials for users, including replication and distribution
- Complete the version description document
- Place under software configuration management control

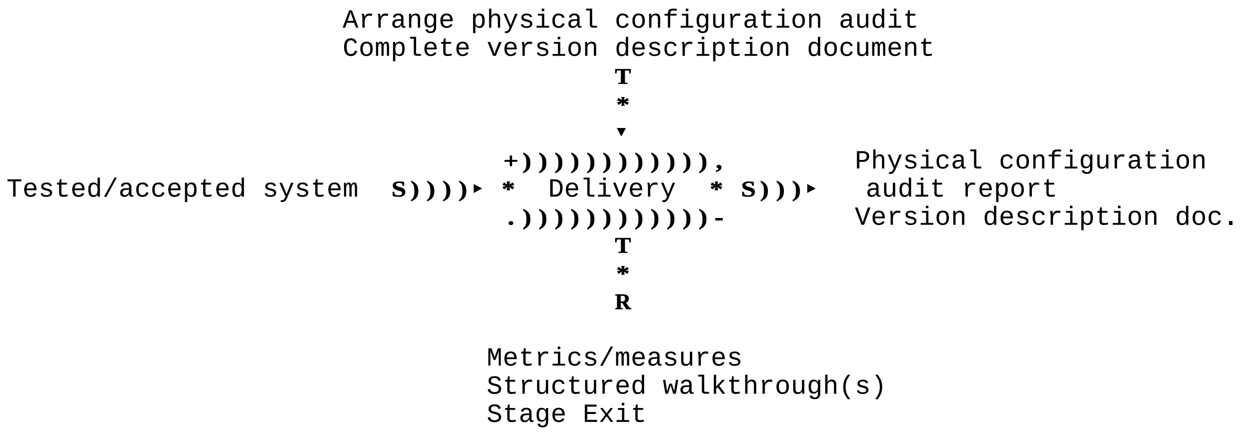
Work Products: The output of the Delivery Stage includes the following:

- Physical Configuration Audit report
- Version Description Document (VDD).

The VDD contains information pertinent to the version or release of the system that is being delivered. Information provided includes system name, date delivered, version number, release number, brief description of functionality delivered in the modification, and prerequisite hardware and software with its associated version and release number. The current VDD is placed together with VDDs from previous versions/releases to form a complete chronology of the system from its initial implementation or Version 1,
Release 1.

Review Process: Conduct structured walkthrough(s) and a Stage Exit.

Exhibit 10.7-1. Delivery Stage


Arrange physical configuration audit
Complete version description document
Tested/accepted system
Delivery
Physical configuration audit report
Version description doc.
Metrics/measures
Structured walkthrough(s)
Stage Exit