Chapter: **8.0 Testing Stage**

Description: Testing activities focus on interfaces between and among components of the product, such as functional correctness, system stability, overall system operability, system security, privacy and sensitive information control, and system performance requirements (e.g., reliability, maintainability, and availability). Testing performed incrementally provides feedback on quality, errors, and design weaknesses early in the integration process.

In this stage, components are integrated and tested to determine whether the product meets predetermined functionality, performance, quality, interface, and security requirements. Once the product is fully integrated, system testing is conducted to validate that the product will operate in its intended environment, satisfies all user requirements, and is supported with complete and accurate operating documentation. User Acceptance Testing (UAT) follows System Testing, and solicits feedback from users to make any final adjustments to the programming before releasing the product for implementation.

Input: The following items provide input to this stage:

SEM Templates:

- Conversion Plan
- Installation Plan
- Maintenance Plan
- Requirements Specification
- Requirements Traceability Matrix
- Test Plan
    - Integration testing (component to component)
    - Performance testing (load, stress, etc.)
    - System testing (end to end)
    - User acceptance testing (UAT)
- Test Reports
    - Integration test reports
    - Performance test report
    - System test reports
    - User Acceptance test reports
- Transition Plan
- Training Plan

PMM Templates:

- Project Plan
- Quality Management Plan
- Security Plan

Other Inputs:

- Development Baselines
- Operating Documentation
  - Users Manual
  - Developer's Reference Manual
- Project Test File
- Software Modules

High-Level Activities:

The remainder of this chapter is divided into sections that describe specific high-level activities performed during this stage. These activities represent the minimum requirements for a large information systems engineering effort. *Notes* are provided, as applicable, to assist in customizing these lifecycle stage requirements to accommodate the different sizes of information systems engineering efforts. The high-level activities are presented in the sections listed below.

8.1 Conduct Integration Testing
8.2 Conduct System Testing
8.3 Conduct User Acceptance Testing

Touch Points:

The following touch points are involved in the Testing Stage:

- Contracts and Procurement
  - Contract Liaison involvement if contract issues arise
- E-Michigan
  - Continue to work with E-Michigan's webmaster, as appropriate, to ensure ADA compliance and Michigan.gov look and feel standards
- Infrastructure Services
  - Infrastructure Specialist involvement as documented in the Infrastructure Services Request (ISR)
- Security
  - Include application testing for security controls

Output:

Several work products are produced during this stage. The work products listed below are the minimum requirements for a large project. Deviations in the content and delivery of these work products are determined by the size and complexity of the project. Explanations of the work products are provided under the applicable activities described in the remainder of this chapter.

SEM Templates:

- Error Reporting and Tracking Checklist (*final*)
- Integration and System Testing Checklist (*final*)
- Pre-acceptance Checklist (*final*)
- Testing Package Checklist (*final*)
- User Acceptance Checklist (*final*)
- Conversion Plan *(revised, if needed)*
- Installation Plan *(final)*
- Maintenance Plan *(revised)*
- Requirements Traceability Matrix *(final)*
- Test Reports *(final)*
    - Integration test reports
    - Performance test report
    - System test reports
    - User Acceptance test reports
- Training Plan *(final)*
- Transition Plan *(revised)*

PMM Templates:

- Project Plan *(revised)*
- Security Plan *(revised)*

Other Outputs:

- Operating Documents *(final)*
    - Users Manual
    - Developer's Reference Manual

A diagram showing the work products associated with each SEM stage is provided in *Exhibit 8.0-1, SEM Overview*. The activities for this stage are emphasized in bold.

Review the Project Plan for accuracy and completeness of all Testing Stage activities and make any changes needed to update the information.

Review Process: Quality reviews are necessary during this stage to validate the product and associated work products. The activities that are appropriate for quality reviews are identified in this chapter and Chapter 2.0, Lifecycle Model. The time and resources needed to conduct the quality reviews should be reflected in the project resources, schedule, and work breakdown structure.

Structured Walkthrough (SWT)
Requirements for a peer review or a more formal structured walkthrough are documented under *Review Process* at the end of each Task, Subtask, or Activity

section in this stage. The State of Michigan guide titled *Structured Walkthrough Process Guide* provides a procedure and sample forms that can be used for SWTs. This document is available on the MDIT SUITE website.

Stage Exit

Schedule a Stage Exit as the last activity of the Testing Stage to enable the project approvers to review project deliverables and provide a concur/non-concur position to the project manager. The State of Michigan guide titled *Stage Exit Process Guide* provides a procedure and sample report form that can be used for stage exits. This document is available on the MDIT SUITE website.

References:

Chapter 2.0, Lifecycle Model, *Quality Reviews* provides an overview of the Quality Reviews to be conducted on a project.

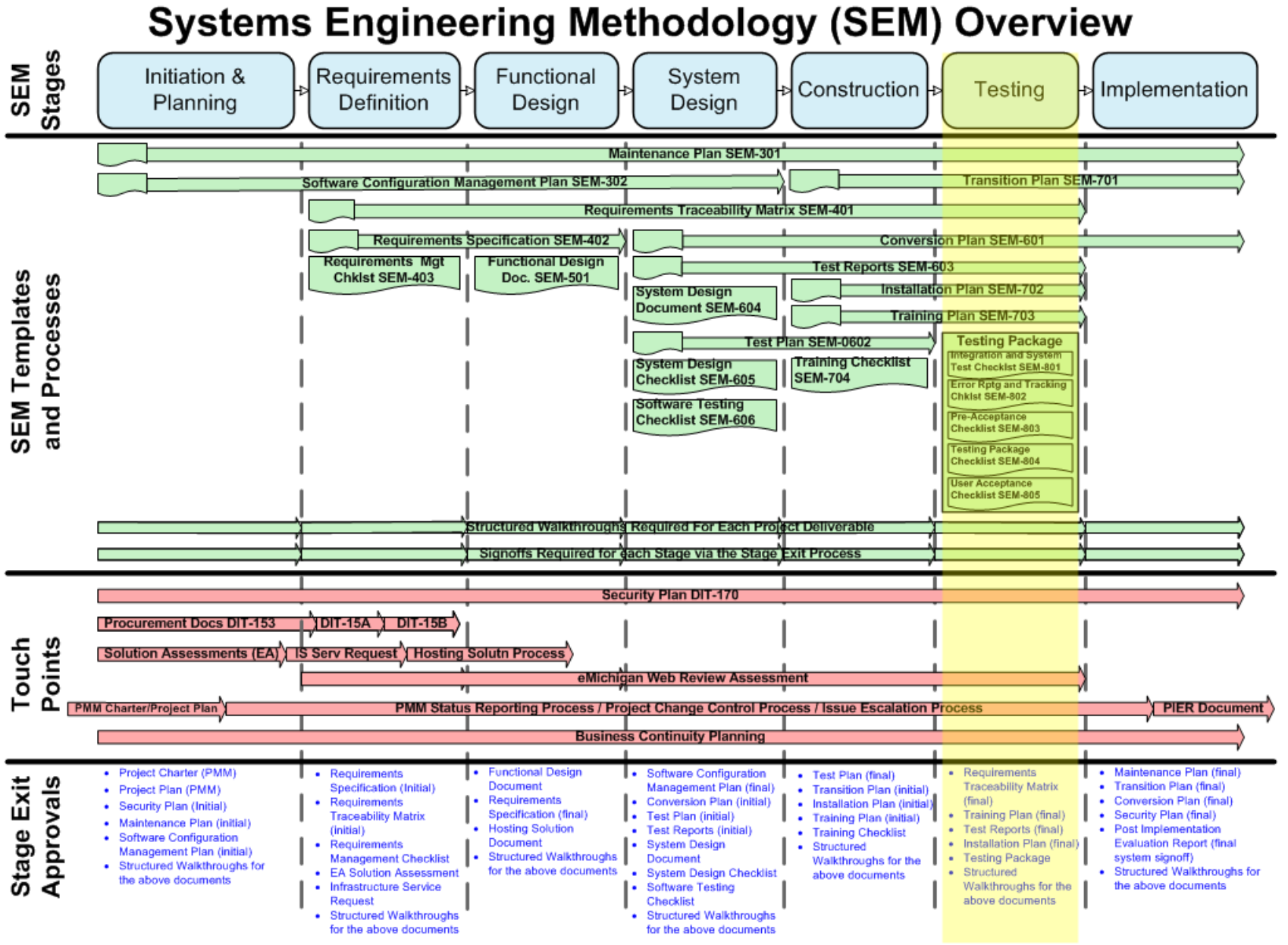
Bibliography:

The following materials were used in the preparation of the Testing Stage chapter.

1. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Developing Software Lifecycle Processes,* IEEE Std 1074-1991, New York, 1992.

2. U.S. Department of Commerce, National Institute of Standards and Technology, *Guide to Software Acceptance,* 500-180, Washington, D.C., 1990.

Exhibit 8.0-1 SEM Overview Diagram – Testing Stage Highlighted



Activity: **8.1 Conduct Integration Testing**

Responsibility: Project Team Developers

Description: Integration testing is the first activity in the Testing Stage and requires special attention to preparation. The Pre-Acceptance Checklist, Integration and System Test Checklist, and Testing Package Checklist each provide the necessary steps for their preparation. These documents are available on the MDIT SUITE website.

During integration, the components constructed by the development staff, off-the-shelf software purchased from vendors, and reusable code or modules obtained from other sources are assembled into one product. Each assembly is tested in a systematic manner in accordance with the Integration Section of the Test Plan. An incremental approach to integration enables verification that as each new component is integrated, it continues to function as designed and both the component and the integrated product satisfy their assigned requirements.

Integration testing is a formal procedure that must be carefully planned and coordinated with the completion dates of the unit-tested modules. Integration testing begins with a structure where called sub-elements are simulated by stubs. A stub is a simplified program or dummy module designed to provide the response (or one of the responses) that would be provided by the real sub-element. A stub allows testing of calling program control and interface correctness. Stubs are replaced by unit-tested modules or builds as integration testing proceeds. This process continues one element at a time until the entire system has been integrated and tested.

Integration testing may be performed using "bottom up" or "top down" techniques. Most integration test plans make use of both bottom-up and top-down techniques. Scheduling constraints and the need for parallel testing will affect the test approach.

The bottom-up approach incorporates one or more modules into a build; tests the build; and then integrates the build into the structure. The build normally comprises a set of modules that perform a major function of the system. Initially, the function may be represented by a stub that is replaced when the build is integrated.

In the top-down approach, individual stubs are replaced so that the top-level control is tested first, followed by stub replacements that move downward in the structure. Using top-down integration, all modules that comprise a major function are integrated, thereby allowing an operational function to be demonstrated prior to completion of the entire system.

Given the incremental nature of the Testing Stage, completion and sign-off of the Integration Section of the Integration and System Testing Checklist is required prior to moving on to System Testing.

Work Products: Each requirement identified in the Requirements Specification must be tested during integration testing. This traceability ensures that the product will satisfy all of the requirements and will not include inappropriate or extraneous functionality. Expand the Requirements Traceability Matrix developed in the Requirements Definition Stage to relate the integration test to the requirements. Place a copy of the expanded matrix in the Project File.

At the completion of each level of integration testing, a test report is written. The report documents test results and lists any discrepancies that must be resolved before the tested components can be used as the foundation for another integration level. Place a copy of all integration test materials in the Project Test File.

A final test report is generated at the completion of integration testing indicating any unresolved difficulties that require management attention. Place a copy of the final Integration Test Report in the Project File.

Sign-off of the Integration section of the Integration and System Checklist signifies completion of the Integration Testing activities.

A formal reporting system by which detected errors and discrepancies are recorded and fully described is recommended. These reports will help to confirm that all known errors are fixed before delivery of the completed product. Error reports also help to trace multiple instances of the same error or anomalous behavior, so that error correction and prevention assignments can be implemented. The Quality Assurance representative assigned to the project can provide assistance in developing and using an error reporting/tracking system. An Error Reporting and Tracking Checklist document is available on the MDIT SUITE website.

Review Process: Conduct a structured walkthrough of the Requirements Traceability Matrix and final Test Report based on type of testing (regression, performance, integration, etc.).

Activity: **8.2 Conduct System Testing**

Responsibility: Project Team or Independent Test Team

Description: During system testing, the completely integrated product is tested to validate that the product meets all requirements. System response timing, memory, performance, security, and the functional accuracy of logic and numerical calculations are verified under both normal and high-load conditions. Query and report capabilities are exercised and validated. All operating documents are verified for completeness and accuracy.

System testing is conducted on the system test bed using the methodology and test cases described in the System Test Requirements section of the Requirements Specification document. The system test environment should be as close as possible to the actual production system environment. Either the project team or an independent test team conducts system testing to assure that the system performs as expected and that each function executes without error. The results of each test are recorded and upon completion included as part of the project test documentation.

Note that regression testing is a critical aspect of system testing. It is performed in order to verify that system modifications have not caused unintended effects and that the software or system component still complies with its specified requirements.

When errors are discovered, they should be reviewed by the test team leader to determine the severity and necessary subsequent action. If appropriate, minor problems can be corrected and regression tested by the project team developers within the time frame allotted for the system test. Any corrections or changes to the product must be controlled under configuration management. Major problems may be cause to suspend or terminate the system test, which should then be rescheduled to begin after all of the problems are resolved.

Users may be encouraged to participate in the system tests to gain their confidence in the product and to receive an early indication of any problems from the user's perspective. Inform users that errors and discrepancies may occur during testing and explain the error correction, configuration management, and retest processes.

Given the incremental nature of the Testing Stage, completion and sign-off of the Integration and System Testing Checklist is required prior to moving on to User Acceptance Testing.

Work Products: Review the draft versions of the operating documents, Training Plan, and Installation Plan. Update the documents as needed. Deliver the final versions of the operating documents, Training Plan, and Installation Plan to the system owner and user for review and approval. Place a copy of the approved documents in the Project File.

Place a copy of all system test materials (e.g., inputs, outputs, results, and error logs) in the Project Test File.

Sign-off of the Integration and System Testing Checklist and the Pre-Acceptance Checklist signifies completion of the System Testing activities.

Generate a test report at the conclusion of the system test process. The report documents the system test results and lists any discrepancies that must be resolved before the software product is ready for acceptance testing. Place a copy of the report in the Project File.

Review Process: Conduct a structured walkthrough of the system test report to ensure that all areas of System Testing are complete and that all issues have been resolved.

Activity: **8.3 Conduct User Acceptance Testing**

Responsibility: Project Team and Acceptance Test Team

Description: Acceptance of a delivered product is the ultimate objective of a development project. Acceptance testing is used to demonstrate the product's compliance with the system owner's requirements and acceptance criteria.

At the system owner's discretion, acceptance testing may be performed by the project team, by the system owner and users with support from the project team, or by an independent verification and validation team. Whenever possible, users should participate in acceptance testing to assure that the product meets the users' needs and expectations. All acceptance test activities should be coordinated with the system owner, user(s), operations staff, and other affected organizations.

Acceptance testing is conducted in the test environment using acceptance test data and test procedures established in the Acceptance Test Requirements section of the Requirements Specification. Testing is designed to determine whether the product meets functional, performance, and operational requirements. If acceptance testing is conducted on an incremental release basis, the testing for each release should focus on the capabilities of the new release while verifying the correct operation of the requirements incorporated in the previous release.

If the project team is not conducting the User Acceptance Test (UAT), training may be required for the personnel performing the testing. The acceptance test participants and their experience with the product and the operating environment should have been identified in the Acceptance Test Requirements within the Requirements Specification.

Acceptance testing usually covers the same requirements as the system test. Acceptance testing may cover additional requirements that are unique to the operational environment. The results of each test should be recorded and included as part of the project test documentation.

UAT is typically the final phase in a software development process in which the software is given to the intended audience to be tested for functionality. UAT is done by making the software available for testing by an in-house testing panel comprised of users who would be using the product in real-world applications. UAT is done in order to get feedback from users to make any final adjustments to the programming before releasing the product to the intended user community.

The level of training will depend on the testers' familiarity with the product and the platform on which the product will run. The advantage of having users acceptance test the product is that they are the experts most familiar with the business information flow and how the product must fit into the workplace.

It is recommended that the operating documents and other test materials be distributed to the test team prior to the actual start of the acceptance test training. This will give the test team time to become familiar with the product and the test process and procedures.

Subject the test environment to strict, formal configuration control to maintain the stability of the test environment and to assure the validity of all tests. Review the acceptance test environment, including the test procedures and their sequence, with the system owner and user before starting any tests.

Testing is complete when all tests have been executed correctly. If one or more tests fail, problems are documented, corrected, and retested. If the failure is significant, the acceptance test process may be halted until the problem is corrected.

Completion and sign-off of User Acceptance Testing is required prior to moving on to the Implementation Stage.

Work Products: Sign-off of the User Acceptance Checklist and the Testing Package Checklist signifies completion of the Testing Stage.

Prepare a formal Acceptance Test Report. Summarize the test procedures executed, any problems detected and corrected, and the projected schedule for correcting any open problem reports. Place a copy of all acceptance test materials in the Project Test File.

Review Process: A Readiness Review is a combined quality assurance and configuration management activity. It focuses on the results of the acceptance test and the readiness of the product to go into production including review of security, sensitive information and privacy control. Descriptions of the components of a Readiness Review are included below.

It includes a functional configuration audit to determine whether the test records demonstrate that the product meets its technical requirements and a physical configuration audit to determine whether the product technical documentation is complete and accurately describes the product.

The Functional Control Audit (FCA) compares the system's elements (programs/modules) to the requirements documented in the current version of the 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.

The Physical Control Audit (PCA) compares the 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.

During the Readiness Review examine acceptance test results with the system owner and user. Document any problems, determine solutions to the problems, and implement action plans. Once any problems associated with the acceptance test are resolved, the product is ready for formal acceptance by the system owner.

A successful Readiness Review establishes the operational baseline for the product. The operational baseline is the final baseline. It consists of the product and the technical documentation that describes the operational product and its characteristics. It contains the current functional baseline, the product baselines for the configuration items comprising the system, and other system-level technical documentation generated during the lifecycle.

If the operational product requires enhancements or changes to correct problems, each new release should be preceded by a Readiness Review, after which the updated system documentation is established as a new operational baseline superseding the previous one.