Federal Student Aid
An OFFICE of the U.S. DEPARTMENT of EDUCATION

PROUD SPONSOR of the AMERICAN MIND™

Enterprise Test Management Standards

Version 5.00 9/6/2016

Version Control

This section summarizes this document revision history. Each entry includes the version number of this document, the date of the change, the page number where the change occurred, and a brief description of the change. The most current change will always appear on the last row of the table.

Version	Date	Description
1.0	09/28/2007	Initial version of Enterprise Testing Standards Handbook.
2.0	09/17/2008	Update to Handbook to include new information from focus groups, industry best practices, and Federal Student Aid standards.
2.1	11/24/2008	List each Testing Techniques in the table of contents under Appendix D. Correct headers in Appendix D and Appendix E. Infrastructure testing – add guidance for Operating System upgrades.
2.2	10/02/2009	Appendix A: Add NIST. Appendix B: Add NIST, Middleware, Penetration Testing, & Risk Based Testing. Appendix E: Add Template Version Number to beginnings of all templates, and to template file names. Add new Template 23 (Disaster Recovery Test Readiness Review documentDocument). In Appendix D: Add guidance on testing requirements in Infrastructure/ Application Testing section. Update Disaster Recovery section. Add items that Enterprise Testing Team and that Business Units are responsible for. Add section/list on DR TRR. Add Risk Based test case determination to section 20. Add statement saying policies on penetration testing and vulnerability scanning will be posted on FSA Portal Website. Replace all instances of EAI with ESB. Section 2.3: Add EDSS SA-3 advisory statement. Add New Section 2.4 (Budget Considerations). Section 3.4.1.1: Add FSA Test Manager responsibility to provide sign-off Signature on PRR Memorandums.

Version	Date	Description
03.00	09/29/2010 11/08/2010	Updated to document in order to address the standardization of the Rational suite of tools (RequisitePro, ClearQuest, ClearCase and Rational Quality Manager) within FSA. Removed some of the templates due to process and tool changes. • Deleted the test execution report, test suites and defect report references throughout the document. • Updated the number of templates in Appendix E • Removed FSA from Figures 1-1 and 1-2 and 3-2.
04.00	09/28/2012	Updated FSA logo; added document number assigned by CM; added a paragraph in Section 1 Introduction and the Defect Management Section to describe other Rational tools being used in FSA; more updates to the Defect Management section and Agile Testing section; updated formats to comply with 508 accessibility guidelines
4.01	12/26/2015	Revised text, format, and document structure to meet PMBOK, IEEE, and Best Practices standards and conventions.
5.00	09/06/2016	Multiple updates were made throughout the document to reflect current FSA enterprise test management standards. A section dedicated to contractor selection was removed. Several vendor specific references were revised so that the document is more vendor neutral. Several significant edits have been made to each major section of the document.

Table of Contents

List of Figures

List of Tables

1 Introduction

Federal Student Aid's Technology Office developed the Enterprise Test Management Standards document as part of an initiative to standardize testing policy and practices within Federal Student Aid. The standards document was created using feedback from Federal Student Aid focus groups, as well as industry and Federal Student Aid best practices. Policies and standards for all test phases are described in this document. The standards document supports Federal Student Aid's efforts to achieve structure, consistency, repeatability, and continuous process improvement in software testing.

The standards document supports Federal Student Aid's efforts to achieve structure, consistency, repeatability, and continuous process improvement in software testing and serves as test specifications for the organization to adhere to when conducting tests of enterprise scale application development platforms.

Federal Student Aid has defined a Target State Vision (TSV) to describe how it should operate and administer Title IV programs. The TSV is based upon delivering student aid in an efficient and cost-effective manner, providing the best access to customers, and maintaining appropriate levels of oversight. The standards document is intended to provide guidance for testing of software applications developed under the TSV.

The information in this standards document covers all phases of application testing. Details of application testing contained in this document include:

- Federal Student Aid-approved standardized terminology and acronyms
- Roles and responsibilities
- Description of required test phases and test types
- Testing organization and processes
- Test planning to include such management requirements as Risk Management and Defect Management
- Required artifacts and templates
- Tools and techniques
- Standards and metrics
- Specialized testing techniques

The requirements contained in this Document apply to Federal Student Aid Application Owners and Contractors who provide software application testing support and management.

The standards document will require revisions over time to reflect lessons learned, new "best practices," changes in industry methods, and the usage of new tools and techniques for software testing.

The Enterprise Test Management Standards document is part of an initiative to standardize testing policy and practices at Federal Student Aid. The information

contained in this document is essential information for all involved in test efforts. The Enterprise Testing Team is responsible for the contents of this document and will provide guidance on the information contained herein.

The Defect Management tools used in FSA are used to support the management of not only testing artifacts including test plans and test cases and but also the identification of testing defects and remediation of these. These tools also allow the test team to work closely with the requirements team and development team.

The Defect Management products currently described in this document at publication date explain the legacy Defect Management tools for requirements and testing. Some FSA Information Technology projects using Agile methodology are already using the Defect Management products that integrate requirements, development, and testing activities. Defect Management requirements may be linked to test cases and work items such as stories, tasks, and defects are managed as well. These processes are described in a limited fashion and will be further described in future updates to this document.

1.1 Purpose

This standards document supports Federal Student Aid's efforts to achieve structure, consistency, repeatability, and continuous process improvement in software testing. It sets forth policies and standards for all aspects and phases of testing, as well as the creation of the ensuing test artifacts.

1.2 Scope

This standards document contains Federal Student Aid's testing requirements.

The focus of this document is on test planning, execution of the phases of testing, and the reports and artifacts generated. These activities occur during the construction and validation, implementation, and support and improvement stages of a project. However, the testing effort must begin much earlier in the project lifecycle, during the definition stage.

1.3 References

- Estimating the Software Testing Process - Youri Kroukov
- Operational Excellence through efficient software testing metrics - Ramesh Pusala
- Guide to the CSTE Common Body of Knowledge - Quality Assurance Institute
- IEEE 829-1998
- Code Coverage Analysis - Steve Cornett
- http://www.webspiders.com/en/testing_lifecycle.asp
- Testing IT: An Off-the-Shelf Software Testing Process - John Watkins
- www.research.umbc.edu\ncseaman\ifsm698\lect0910.ppt
- http://www.umbc.edu/oit/iss/syscore/wiki/Main_Page

- www.eforceglobal.com
- Scott Ambler's website, Agile Modeling.
- Mike Cohn's website, Mountain Goat Software.
- Lisa Crispin and Janet Gregory, Agile Testing: A Practical Guide for Testers and Agile Teams, Addison-Wesley, Crawfordsville, IN, © 2009.
- The Original Software Group Limited, *The Reality of Software Testing in an Agile Environment*, © 2010, http://www.origsoft.com/whitepapers/agile-testing-environment/.
- Roman Pichler, Agile Product Management with Scrum: Creating Products that Customers Love, Addison Wesley, Westford, MA, © 2010.
- Roman Pichler's website, http://www.romanpichler.com/.
- Ken Schwaber and Jeff Sutherland, "The Scrum Guide", © 2010, Scrum Guides.
- The Agile Manifesto, © 2001, http://www.agilemanifesto.org.
- *BEST PRACTICE: Functional System Testing Technique Categories*.
- FSA Detailed Requirements Document Exemplar, Version 1.3.
- FSA Enterprise Requirements Management Standards and Tailoring Guidance, Version 1.00.
- IEEE Standard 829-2008, IEEE Standard for Software and System Test Documentation
- ISO/IEC/IEEE 29119, Software and systems engineering — Software testing
- CMMI® for Development, Version 1.3
- CSTE STBOK (Software Testing Body of Knowledge for CSTE), Version 14.2
- Agile Testing: A Practical Guide for Testers and Agile Teams
- Jeff Sutherland's Scrum Handbook

This document was created in accordance with the following Federal Student Aid policies:

- Department of Education, ACS Directive OCIO: 1-106 Lifecycle Management (LCM) Framework, dated 12/02/2005.
- Department of Education, ACS Directive OCIO: 3-105 Procuring Electronic and Information Technology (E&IT) in Conformance with Section 508 of the Rehabilitation Act of 1973 as Amended, dated May 1, 2006.
- Department of Education, ACS Directive OM: 5-101: Contractor Employee Personnel Security Screenings; dated 03/30/2005.
- Department of Education, Federal Student Aid Acquisition Process Handbook, Version 1.1, January 15, 2008

- Department of Education, Federal Student Aid Style Guide, Version 1, dated July 21, 2006.
- Department of Education, Federal Student Aid System Integration and Testing Process Handbook Volumes 1, 2 and 3, dated January 12, 2001.
- Department of Education, Federal Student Aid Virtual Data Center Configuration Management Database Data Dictionary, Version 1.1, dated November 7, 2007.

1.4 Intended Audience

Cross Reference for Intended Audiences identifies relevant sections of this document for a variety of types of roles, based on responsibilities. Federal Student Aid requires all of these participants to learn and comply with the processes identified with roles in the matrix. Everyone is also encouraged to become familiar with the rest of this document.

Table 1: FSA Intended Audience

FSA Roles	Section 1	Section 2	Section 3	Section 4	Section 5	Section 6	Appendix D
Application Owners	✓	✓	✓				
Project Manager and Program Manager	✓	✓	✓	✓	✓	✓	
Enterprise Test Manager	✓	✓	✓	✓	✓	✓	✓
Test Manager	✓	✓	✓	✓	✓	✓	✓
Development Manager (small apps)	✓		✓	✓	✓	✓	✓
Development Team (small apps)			✓	✓	✓	✓	✓
Requirement Team	✓		✓		✓	✓	
Test Suite Reviewer			✓		✓	✓	
Test Leads (each test phase as assigned)	✓		✓	✓	✓	✓	✓
Integration Testers			✓	✓	✓	✓	✓
UAT Tester			✓	✓	✓	✓	✓
Post Implementation Testers			✓	✓	✓	✓	✓

Table 2: Contractor Intended Audience

Contractor Roles	Section 1	Section 2	Section 3	Section 4	Section 5	Section 6	Appendix D
Project Manager	✓	✓	✓		✓	✓	
Development Manager	✓	✓	✓	✓	✓	✓	✓
Test Manager	✓	✓	✓	✓	✓	✓	✓
Requirements Manager	✓	✓	✓		✓	✓	
Development Team			✓	✓	✓	✓	✓
Requirement Team			✓			✓	
Test Leads (all test phases)	✓		✓	✓	✓	✓	✓
Integration Testers			✓	✓	✓	✓	✓
System Testers			✓	✓	✓	✓	✓
UAT Support Team			✓	✓	✓	✓	✓
Post Implementation Testers			✓	✓	✓	✓	✓

Table 3: Intended Audience Roles

Primary / Secondary	Title/Roles
Primary	Federal Student Aid and/or Contractor Project Managers
Primary	Federal Student Aid and/or Contractor Test Managers/Test Leads
Primary	Federal Student Aid Enterprise Testing Team
Secondary	Testing Teams
Secondary	Federal Student Aid Application Owners
Secondary	Application Development Teams

Non-technical users should also consult **Appendix C: Testing Techniques** for an overview of fundamental testing concepts, as well as the **Glossary** in **Appendix B**.

1.5 Naming Conventions

The terms "*application*," "*application software*," "*project*," and "*system*," used to describe the items to be tested, are defined in the *Virtual Data Center Configuration Management Database Data Dictionary*, Version 1.1, November 7, 2007.

1.6 Document Organization

This document includes six narrative sections that contain the information needed to implement software testing standards and practices that meet Federal Student Aid's testing requirements.

These sections are:

1. Introduction
2. Test Planning – provides for an understanding of the application, requirements, development activities and test planning documentation that will guide the testing effort.
3. Testing Phases – describes application testing during development and after deployment, and includes phase descriptions and the associated standards.
4. Defect Management – describes the process for recognizing, investigating, taking action on, and disposing of defects and production issues.
5. Test Metrics and Reporting – outlines the efficiency and effectiveness of testing activities, test plans, and test processes throughout all phases of testing.

In addition, there are four appendices that provide supplementary information, standard document templates, and documents to aid in implementing the standards and practices described in the narrative sections.

These appendices are:

- **Appendix A: Acronyms and Abbreviations** – contains definitions for all of the acronyms and abbreviations used in this document.
- **Appendix B: Glossary** – contains definitions for the major terms used in this document.
- **Appendix C: Testing Techniques** – contains an introduction and overview of the most common testing techniques.
- **Appendix D: Templates** – contains the required templates that support this document and their instructions where applicable.

Important information is included in Note or Notes followed by the text.

1.7 Terminology

Federal Student Aid recognizes that sometimes there are several terms used in the industry for the same testing concept or activity. Throughout this Document, there is

standardized terminology related to Federal Student Aid testing to avoid confusion and ensure consistency in understanding the concepts and activities described. Federal Student Aid acquisitions will include these terms in language related to testing. Federal Student Aid and contract staff will use these terms in all testing deliverables and documentation.

1.8 Lifecycle Management Methodology (LMM) and Testing

Federal Student Aid has implemented the Lifecycle Management Methodology (LMM) to ensure more effective and responsible management of projects from conception to retirement. The LMM outlines the stages required for each project, the key activities, and the core deliverables.

The LMM advocates a sufficient number of artifacts, placing importance on the quality of the functional solution. There are specified stages that guide project managers through the LMM process by identifying a core set of artifacts to be created to ensure each project's viability. Distinct stages describe the testing artifacts such as the Master Test Plan, Test Suites, and various Test Reports.

Specifically, the degree of rigor assigned to a Test Readiness Review Stage Gate is determined by the project size, scope and complexity. The purpose of Test Readiness Reviews is to provide management with an assessment of the readiness of the development maturity, test environment, test data, test processes, deliverables and other dependencies to ensure the system is ready to pass from build/construct to formal testing (System and User Acceptance) and that known risks have been documented, accepted or mitigated. Test Readiness Reviews are required for all projects for phase of testing.

The Enterprise Test Management Standards contained in this Document complement the LMM by defining specific testing phases that create periodic evaluations. These evaluations are designed to determine how well a project is meeting the requirements defined for the application.

The LMM promotes an iterative development lifecycle where Definition, Development and Testing stages repeat until the developed solution fulfills end user requirements.

FSA's Lifecycle Management Methodology is designed to organize the typically large solution development process into stages that will be easier to manage and understand. The image below provides an overview of the stages of the LMM and the high-level activities that take place in each stage.

Figure 1: Lifecycle Management Methodology and Related Testing Activities, describes the related testing activities and deliverables.

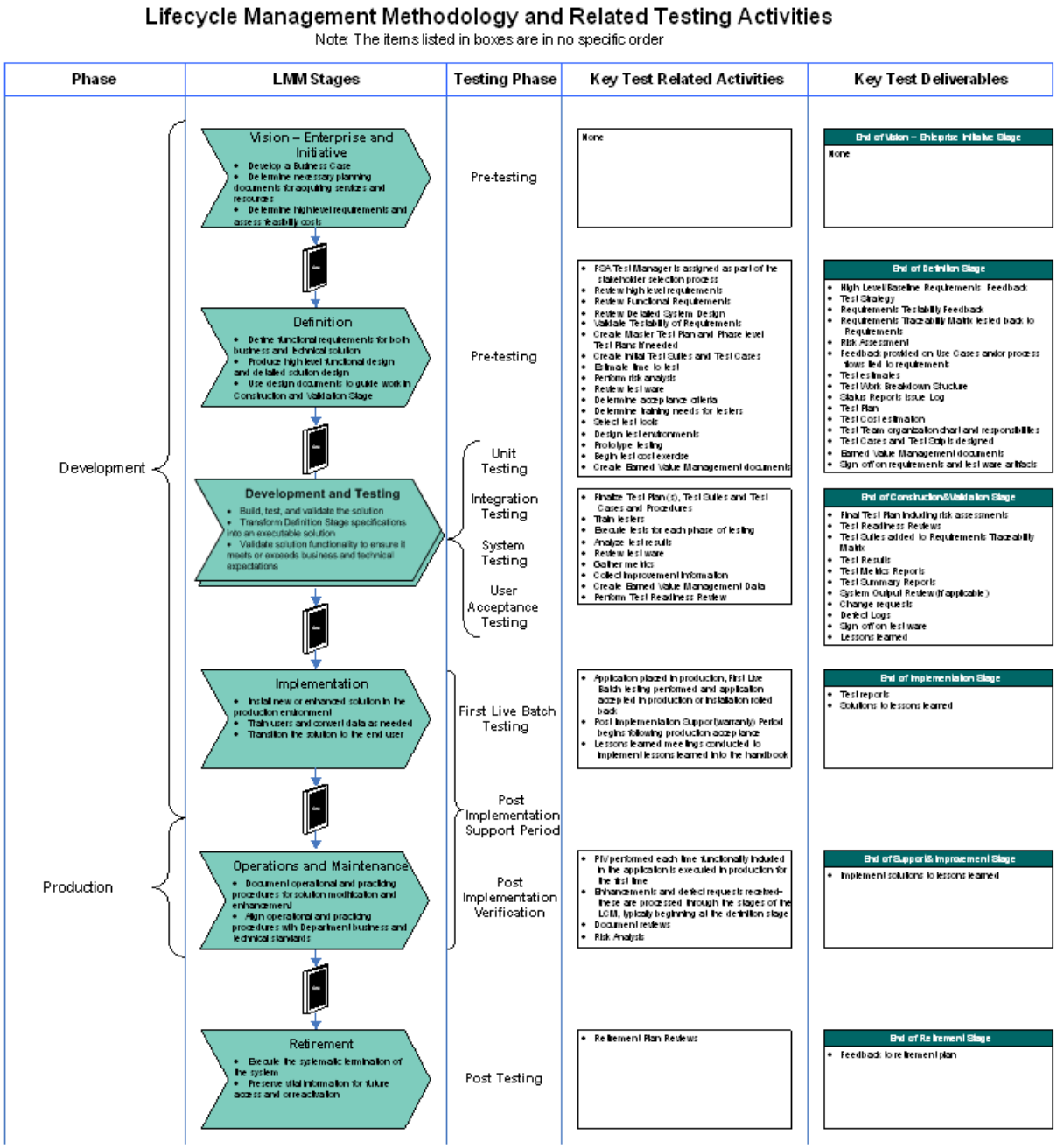


Figure 1: Lifecycle Management Methodology and Related Testing Activities

Another view of the relationship of testing in the Lifecycle Management Methodology is presented in Figure 2.

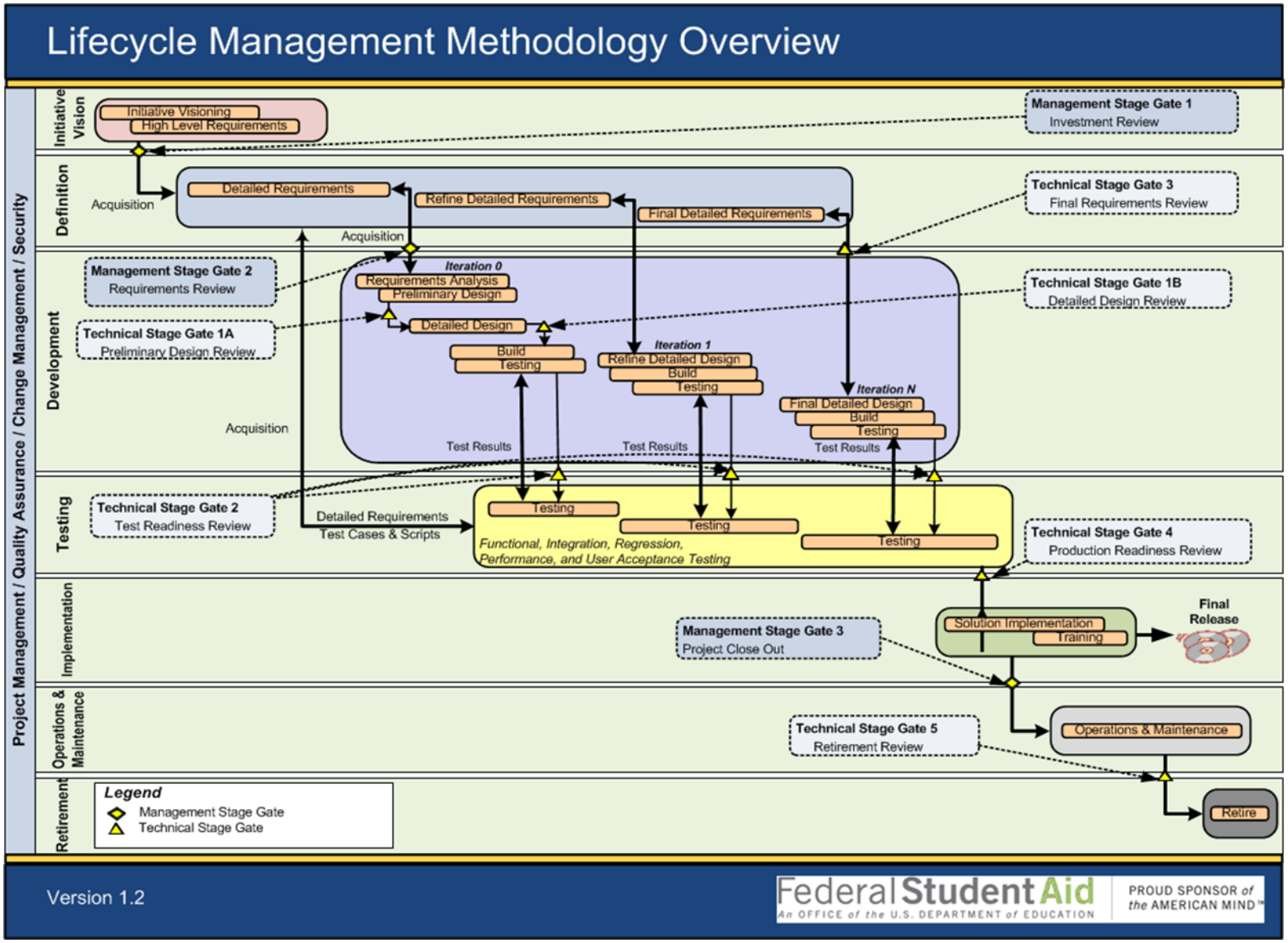
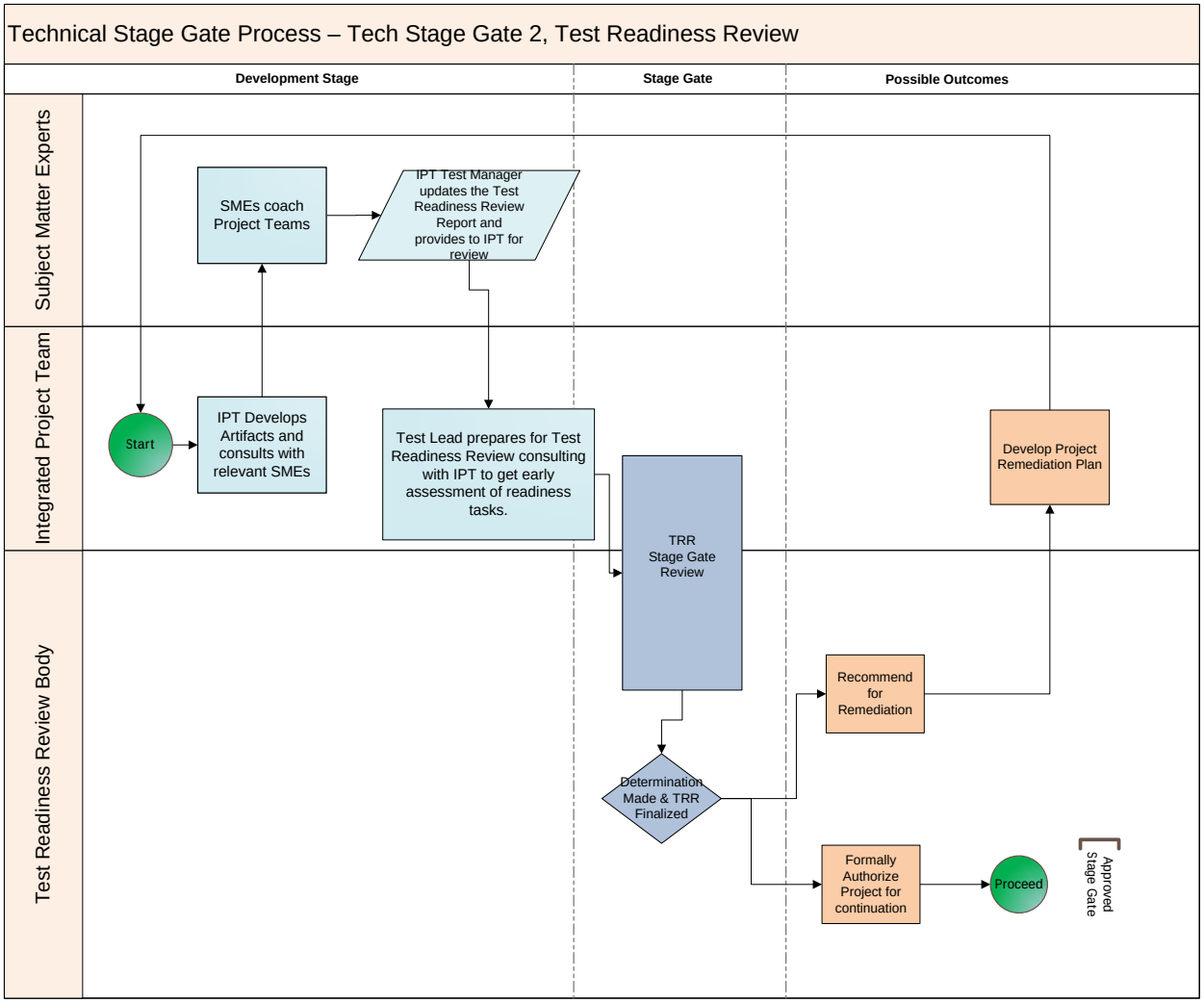


Figure 2: LMM Overview and Related Testing Phases

2 LMM Technical Stage Gate 2 – Test Readiness Review (TRR)

2.1 Purpose

The Test Readiness Review (TRR) is the LMM Technical Stage Gate 2 and is a critical component of testing activities. Test Readiness Reviews are required for all projects for each iteration of testing and for system and user acceptance testing and formal integration testing when required. . The degree of rigor assigned to a Test Readiness Review Stage Gate is determined by the project size, scope and complexity. The purpose of Test Readiness Reviews is to provide management with an assessment of the readiness of the development maturity, test environment, test data, test processes, deliverables and other dependencies to ensure the system is ready to pass from build/construct to formal testing and that known risks have been documented, accepted or mitigated. Test Readiness Reviews are required for all development projects and COTS implementations based on tailoring plan guidance or guidance from the FSA SME LMM Stage Gate 2 subject matter expert.



The TRR document is reviewed by senior members of the team to determine if testing is ready to be conducted. Ultimately the completeness and accuracy of the TRR will determine whether or not the system under review is ready to proceed into formal testing. Approval of Stage Gate 2 is required before system testing, user acceptance testing and any other formal testing may begin.

Task	Description
Method:	The Test Readiness Review Body (Test Lead and IPT including the technical lead) reviews the Test Readiness Review document to assess and verify whether or not development requirements, artifacts, inputs, and project requirements have been met and activities completed and all technical requirements have been met prior to the start of testing.

Task	Description
Process Inputs:[1] Information or Documents Required	Updated Tailoring Plan and outputs from Technical Stage Gates 1A and B. Acquisition Strategy, Initiative Vision, Project Initiation Artifacts, Privacy Artifacts, Data Retention Schedule, Requirements Management Plan, High Level Requirements Document, Project Management Plan Artifacts, Project Monitoring Artifacts and Pre- and Post-Award Artifacts, ISSO Appt. Letter, External Data Exchange Artifacts, Master Test Plan, UI Specification Document, Data Migration Plan, Implementation/Transition Management Plan, Continuity of Services Artifacts, Configuration Management Plan, Preliminary Design Document, Requirements Traceability Matrix, Detailed Design Document, Test Suites, Training Plan, Security Risk Assessment Artifacts, Solution Source Code and Deployable Packages, and Operations and Maintenance Plan.
Who Attends:	• Testing Team (leads and critical test team members) • IPT (PMs, business analyst lead, and, development, test, and security teams) • Assigned FSA Test Lead, FSA Project Manager, FSA Technical Lead, Business Owner Representatives • Contractor technical team leads • CO and/or COTR (optional)

The TRR document provides information concerning test methods and procedures, the scope of the testing, security, and the resources required to properly coordinate and support planned testing. The TRR document also provides information concerning development maturity, schedule, environment constraints and risks.

The TRR template (found in Appendix D) is used to convey the information required for readiness review to take place. All TRRs must be provided to the LMM SME, even if the project is canceled prior to implementation. The following provides information that must be gathered for inclusion in the TRR.

2.2 Roles and Responsibilities

Stage Gate Preparation: The test team, development team and project manager are the key contributors to the completion of the TRR. More defined roles include:

[1] *Process inputs are aligned with LMM Artifacts Summary and Activities. While a set number of artifacts and stage gates have been identified, there may be variances based on the type of project (e.g. Tier 1, 2, 3 projects). The LMM Tailoring Team and SMEs will work with IPTs on tailoring based on the needs and best interests of the project.*

- Stage Gate 2 Subject Matter Expertise required to review TRR outcome and available for consultation; and
- Lead IPT members and Personnel who are independent from the project being assessed (based on SME recommendation). Independence should be determined as part of the criteria during the project planning.
    - **Test Manager / Test Lead** - Responsible for compiling TRR information, documenting it within the TRR template, and coordinating the TRR meeting to assess testing preparedness.
    - **Test Team** – Provide information and plans concerning test phase participants, test phase logistics, test phase scope, documentation, entrance criteria, exit criteria, defect management, analyzing change requests, creation of test cases, test scripts, test suites, requirements traceability matrix, and test reporting.
    - **Development Team** - Provide information concerning designing the application and technical architecture, performs Unit Test activities, and supports subsequent testing efforts by analyzing change requests and updating code as required.
    - **Project Manager** – Responsible for the successful testing and implementation of the initiative solution.
    - **Technicall Lead** – Responsible for providing technical guidance to the team.
    - **Business Owner or Business Owner Representative** – Sponsor the development of the project. Provide information concerning defining requirements, participating in risk assessments, prioritizing and clarifying change requests and providing resources to support user acceptance testing activities.

2.3 Entrance Criteria

- Updated requirements traceability matrix.
- All test cases and test suites have been created and uploaded into applicable testing tool.
- The TRR functions as the stage gate signifying that the requirements for beginning the next phase of testing conditions have been met.
- Prior to the TRR meeting the test team must be informed of the urgent and high defects found in the previous test phase.

2.4 Planning and Preparation

In compiling TRR information it is important to ensure that the information documented within the template is not merely the perspective of the test manager / test lead, but is the recording of explicit and verifiable information that has been provided by the prospective IPT member. The TRR document must be tailored for each individual project.

Technical Review Stage Gate 2 - Test Readiness Review: Technical Review Stage Gate 2 Test Readiness Review Body may include FSA Testing Subject Matter Experts, the project FSA Test Lead, FSA Technical Lead, FSA security experts, contractors and other stakeholders. Stakeholders from each area responsible for TRR document items must attend TRR meetings. In preparing for this Stage Gate, a Pre-Test Readiness Review is recommended to be completed. At this time it is recommended that the test lead provide the Stage Gate owner a draft of the TRR for review. During the formal TRR, the IPT determines whether or not the artifacts/inputs, and project requirements have been met and activities are completed adequately. If all agree on the completion, the key leads of the project sign off on the TRR giving the test team the green light to start testing. A determination based on project and product risks will be made in some cases calling for the Test Readiness Review SME and Technology Office involvement. . The IPT Test Manager will provide the Test Readiness Review Report to the Stage Gate owner for review and archival.

Stage Gate Decision & Outcomes: If the project does not meet requirements, risks requiring remediation will be cited and the IPT is required to remediate risks and repeat the Test Readiness Stage Gate. If required, refer its conclusions and recommendations to the Executive Board (ERB), either based on a pre-review determination that an Executive review is required, or based on significant concerns identified during the review, which warrant the added level of review. Approval to proceed to the Test Stage is indicated by a determination that the project meets requirements.

The test manager / test lead is responsible for the scheduling the Test Readiness Review meeting and performing the role of moderator during the actual review. In addition, the following should be performed in preparation for the meeting:

1) Schedule the facilities for the review. Ensure the facility will accommodate the number of personnel to attend, proper equipment is available, etc.
2) Develop an agenda. The agenda will include a list of review participants and a schedule that will allow sufficient time to evaluate and discuss all review products, and identify and document discrepancies.
3) Notify all participants.
   a) Inform review participants of time and location of the review.
   b) Notify presenter of specific material to present and allotted time frame.
   c) Advise meeting recorder and review participants of any other special assignment.
4) Distribute the Test Readiness Review template to all participants prior to the review meeting (The test lead/manager should work with the integrated project team to complete as much of the TRR prior to the TRR meeting).
5) A pre-TRR meeting is encouraged before the formal TRR meeting for ciritcal and Tier 1 projects.

2.5 TRR Meeting Execution

The following tasks are executed during the TRR meeting:

1) Confirms that all participants are present at the meeting and are prepared. (Test lead/Manager or designee)
2) Reviews the purpose / objective of the Test Phase for which the TRR meeting has been convened. (Test lead/Manager or designee)
3) Introduces each review item for discussion..
4) Ensures appropriate attendee confirms each review item and addresses any missing or incomplete requirements, providing plans for completion. (Test Lead/Manager or designee and Project Manager)
5) Records any appropriate information edits in the TRR template. (Test Lead/Manager or designee)
6) Ensures each risk is documented with contingency or mitigation plan. (Test Lead/Manager or designee and Project Manager)
7) Notes all action items within the TRR template. (Test Lead/Manager or designee)
8) Summarizes action items obtain approving signatures (if applicable) and close meeting. (Test Lead/Manager or designee)
9) Determines decision for test readiness (verbal approvals and then obtain written signatures). (IPT)

10) Prepares the final TRR and obtains signatures. (Test Lead/Manager or designee)

2.6 Exit Criteria

The following items are produced from the TRR meeting:

1) An updated TRR document
2) A decision and accompanying criteria / direction regarding the commencement of testing
3) The signatures of all TRR template approving authorities

2.7 Post Review Follow-Up

Subsequent to the TRR meeting the Test Manager / Test Lead verifies that all deficiencies have been addressed, and all action items tracked to closure. TRR document submitted to FSA LMM Stage Gate 2 SME.

3 Test Planning

3.1 Overview and Master Test Plan

Federal Student Aid test planning includes understanding the application and requirements, knowing the schedule of development activities, and properly and completely creating the test-planning document(s) that will guide the testing effort. The Test Manager will use the requirements in this section and the test plan templates in **Appendix –D: Templates** to create the test plans. Following these instructions ensures compliance with Federal Student Aid policies and promotes consistency and repeatability of test planning activities across all Federal Student Aid projects. For scheduling purposes, test planning is estimated to be approximately one third of the total testing effort. The Test Manager should engage the FSA Defect Management Support Group early in the process in order to ensure appropriate access to the Defect Management suite of tools.

The Test Manager will create a Master Test Plan (MTP) for most projects. Exceptions will be determined by the Federal Student Aid Application Owner and Project Manager based on system criticality, information sensitivity, system complexity and cost. Project size and complexity are discussed in detail in **Section 2.4.5, Test Estimation** of this document. It is advisable that exceptions be discussed with the Enterprise Test Group. This plan is created before testing begins as a high-level description of the planned testing effort and may be refined throughout the project as additional information becomes available to improve the clarity and accuracy of the planned test effort. For large or complex projects, individual Phase Level Test Plans will supplement the MTP (for example, a Unit Testing Plan and a UAT Plan will be created in addition to the MTP). The MTP is finalized after approval and must not be updated when the project calls for Phase Level Test Plans.

When a MTP is required, the phase level test plans are supplemental to the MTP and these plans must go through the approval process. The MTP may be conditionally approved and later approved based on the approval of all phase level test plans.

The Test Manager will use the MTP Template to create the MTP at a high level before the project begins. The Test Manager will add additional detail as it becomes available during the project. When Phase Level Test Plans are required, the Test Manager will use the Phase Level Test Plan Template to create the Phase Level Test Plans.

Proxie's for the MTP and the Phase Level Test plans may be created in tools such as Rational Quality Manger (RQM). The purpose of the RQM test plan proxies is to allow other RQM-based testing artifacts, such as suites and test cases, to be linked back to the test plans. The test plans created in the tool are not a substitution for the MTP and Phase Level Test Plans for which the templates are in **Appendix D: Templates.**

Notes:

Master Test Plans are conditionally approved when phase level test plans are required.

All phase level test plans are appendices of the Master Test Plan.

The Master Test Plan is approved only after all phase level test plans are approved.

At the end of each test phase all defects must either be closed, deferred, or in a state which is acceptable to the Federal Student Aid Test Manager/Test Lead.

3.2 Relevance of Test Planning to this Document's Audience

The Test Planning section is relevant to all primary and secondary parties that make up the audience for this Document (see **Section 1.4**). The following key staff members need to fully understand and follow the requirements in the Test Planning section of this Document due to their associated responsibilities for this important aspect of testing:

- Federal Student Aid Project Manager
- Federal Student Aid Test Manager/Test Lead
- Federal Student Aid Enterprise Test Manager
- Federal Student Aid Application Owner
- Federal Student Aid Application Development Team
- Federal Student Aid Application Test Team

The following matrix provides staff roles and specific application testing responsibilities. Individuals participating in test efforts will use this section as guidance for creating test plans, evaluating contractor compliance, and planning of all test related activities.

Table 4: Test Planning Roles & Responsibilities

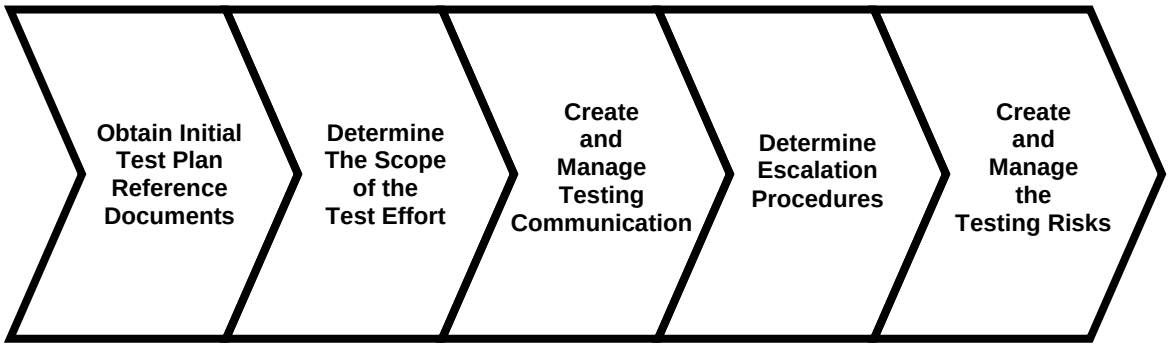
Title	Responsibilities
Project Manager	• Provide input to the MTP by supplying the baselined project plan document and high-level design documents to the Federal Student Aid Test Manager.

Title	Responsibilities
	• Synchronizie the project plan and project schedule with the MTP. • Approve the development schedule. • Participating in risk assessment and mitigation activities.
Test Manager	• Create, maintain, and implement the MTP and Phase Level Test Plans (if separate from the MTP). • Coordinate testing activities between the Test Team and the Project Manager. • Perform ad hoc testing is appropriate in some cases because unique errors may be uncovered. When ad hoc testing is performed, the testing process must be documented step-by-step to facilitate reproducing the test during defect analysis. The Test Manager must manage this process. • Participate in risk assessment and mitigation activities.
Federal Student Aid Enterprise Testing Team	• Maintain this Document to guide test planning activities (according to current Federal Student Aid policy). • Evaluat contractor compliance with Federal Student Aid test planning requirements. • Consult with application test managers and provides testing expertise to development projects.
Federal Student Aid Application Owners	• Estimate resources needed for testing by using both the information in this section and related planning information provided in individual project Test Plans. • Participate in testing risk assessment using information in this section and evaluating the Test Plan risk analysis provided by the Test Manager. • Signing off on test artifacts.
Application Development Teams	• Create and implement the Test Plan for Unit Testing. • Analyze change requests.
Application Test Team	• Assist in creating and implementing the following Phase Test Plans: o Integration Test Plan o System Test Plan o User Acceptance Test Plan o User Acceptance Test Support Plan o Post Implementation Verification Plan o Prioritizing test efforts and perform testing efficiently.

3.3 Test Planning Activities

The Test Manager may perform the following activities identified in **Figure 3: Test Planning Activities** to develop all Test Plans.

Figure 3: Test Planning Activities



The following subsections provide details for each of the activities identified above:

3.3.1 Obtain Initial Test Plan Reference Documents

The Project Manager will provide the following documents to the Test Manager and Testing Team as the basis for starting the test planning process.

- **Baseline Project Management Plan**: This document contains appropriate delivery dates, which will be used to schedule testing activities.
- Baseline System Requirement Specifications: This document will be used to:
  - Plan the scope of testing activities
  - Plan to test both positive and negative test scenarios
  - Determine which components to test
  - Develop testing strategy
  - Facilitate risk identification
  - Determine staffing and training needs
- **High Level and Detailed Design Documents:** Information in these documents will be used to:
  - Determine test environment needs
  - Prepare Test Readiness Reviews (TRR)
  - Identify staffing and training needs
  - Create the test strategy and risk identification using the Baseline System Requirement Specifications
- **Interface Control Documents (ICD):** These documents will be used to plan for Intersystem Testing.
- **Data Management Plan:** During test planning the Data Management Plan must be reviewed in order to determine the types of testing needed, and the types of conversion/balancing reports that will be tested.
- **Statement of Work Document:** This document will be used to determine the stated needs of the application owner.

3.3.2 Determine the Scope of the Test Effort

Federal Student Aid will use the application system criticality, information sensitivity, system complexity, and cost to determine the test plans that must be created for a project. Based on this information, Federal Student Aid will determine whether one comprehensive MTP is acceptable or if separate, detailed Phase Level Test Plans must supplement a MTP.

3.3.3 Create and Manage Testing Communication

Testing communication ensures that all key managers, the test team, and stakeholders are kept informed about testing activities, schedules, and issues. The MTP will contain a Testing Communications Plan. Since testing is critical to and affects many other areas of the application development effort as well as Federal Student Aid infrastructure, communication is vital to the success of testing projects. The Test Manager has overall responsibility for developing and managing all testing communication.

3.3.4 Determine Escalation Procedures

Test Plan(s) must include procedures and responsibilities for addressing situations when service levels are not met or testing is not successful. This process identifies specific areas where escalation to a higher level of authority is appropriate to resolve test issues. An example of an issue resolution process is using colors to determine the level of criticality. Use three levels of criticality, identified as green, yellow, and red. If an issue cannot be resolved at its current level within the time specified in the Test Plan, the issue is escalated to the next level, as indicated below:

- **Green**: This is the initial default status and used when testers log issues. Testers are responsible for issues rated green
- **Yellow**: At this escalation, the Test Lead becomes involved and alternative solutions are considered
- **Red**: When escalated to red, the Test Manager becomes involved in the resolution

3.3.5 Create and Manage the Testing Risks

The principal objective of risk management is to reduce risks to an acceptable level. Risk Management, as part of the Test Plans, identifies project risks, which impact the testing effort and their associated mitigating strategies. In some cases, Risk Management for testing activities may be incorporated into the overall project risk plan. Risk management includes the following activities:

- Risk Identification
- Probability Determination
- Impact Assessment
- Mitigation Measurement Development
- Contingency Plan Development

The Project Risk Plan may contain guidance on how to define the likelihood and impact of risk. If there is no guidance in the Risk Management Plan, the MTP must state how risks will be identified and managed.

The Test Manager will assess the target system functions for criticality and risk. Criticality analysis will be based on the potential consequences associated with an error in or failure of the function. Risk assessment will also be based on the likelihood of an

error in or failure of the function. The results of this analysis will be used to identify catastrophic, critical, and high-risk functions, and to focus testing resources on the most significant aspects of the system design. See Appendix C: section 22 for more information on Risk Based Testing.

The definition of the acceptable level of risk and identification of those software components that are considered significant may include, for example, high-risk software using new technology or concepts. The risk evaluation activity for purposes of focusing the software testing should be accomplished in conjunction with the formal Risk Management approach of the project. The Test Manager should work in close coordination with the Project Manager.

3.4 Detailed Test Plan Planning Activities

As additional information is identified or defined during application project planning, refinements can be made to the MTP and initial planning can begin for each phase of testing. Each Test Plan will be prepared using the appropriate Test Plan template provided in **Appendix D: Templates**. Additional sections may be required based on the type of application being tested.

- Identify Roles and Responsibilities
- Determine Test Strategies
- Determine Test Phases and Test Types (Project based)
- Determine Test Coverage (Project based)
- Prepare Test Estimation
- Identify Components To Be Tested
- Identify Components Not To Be Tested
- Prepare Test Readiness Review (Requires wet signatures on sanctioned template, Appendix D)
- Determine Test Execution Cycle
- Determine Pass/Fail Criteria
- Determine Entry/Exit Criteria
- Determine Suspension Criteria and Resumption Requirements
- Establish / Document Test Deliverables
- Determine Environmental Needs
- Determine Staffing and Training Needs
- Obtain Schedule
- Obtain Metrics and Reporting Planning
- Identify or Create Tailoring the Test Plan or Phase Level Test Plan

3.4.1 Roles and Responsibilities

Each Test Plan will include a section showing the roles and responsibilities of the various contractor and Federal Student Aid staff who are involved in testing activities.

In some cases more than one person may be assigned to or participate in a given activity. In these cases, the individuals work together and the most senior person has overall responsibility.

The following table describes the roles and responsibilities of Federal Student Aid staff in the testing process.

Table 5: Federal Student Aid Roles and Responsibilities

Title	Responsibilities
Application Owner	• Provide resources to support user acceptance testing activities. • Prioritize change requests. • Participate in risk assessments.
Project Manager	• Develop appropriate risk mitigation strategies to ensure that testing is successful. • Approve or rejecting recommendations to perform System Testing and UAT in parallel. • Prioritize change requests. • Communicate the formal recommendation of acceptance or rejection of test deliverables to the Federal Student Aid Contracting Officer or Contracting Officer's Representative. • Provide signature/approval to move to the next phase of testing based on the outcome of the Test Readiness Review. • Determine the feasibility of performing parallel System Testing and UAT (performing testing in both phases simultaneously) due to scheduling issues.
Enterprise Testing Team	• Define organization wide testing standards • Provide leadership and strategic direction to Test Managers and test teams. • Develop the Enterprise Testing data. • Collect Defect Data and Performing Trend Analysis. • Recommend improvements related to Application Testing to the Federal Student Aid Chief Information Officer and Federal Student Aid Application Owners. • Build Federal Student Aid's test information repository and Enterprise Application Testing Management Report. • Maintain the Enterprise Testing Standards Document. • Evaluate contractor's compliance with Federal Student Aid testing requirements. • Lead Test Readiness Reviews for projects as requested by project managers or as directed by senior management based on project criticality, Tier level and need for independence

Title	Responsibilities
Test Manager	• Determine the feasibility of performing parallel System Testing and UAT (performing testing in both phases simultaneously) due to scheduling issues. • Note: Parallel testing may cause risk to the project and the parallel testing decision must include a risk mitigation strategy. The final decision must include input from the Federal Student Aid Project Manager. If a contract is involved, input from the Contractor Project Manager and Contractor Test Manager must be included. • Develop appropriate risk mitigation strategies to ensure that testing is successful. • Develop the recommendation and justification to perform System Testing and UAT in parallel when no contractor is involved. • Develop an appropriate mitigation strategy to ensure that testing is successful. If testing problems occur, employing the above-mentioned technique. • Observe the test effort performed by the contractor. • Note: the Federal Student Aid Test Manager and Project Manager will determine when the observation will take place. No formal announcement has to be given to the contractor as to when the observation will take place. • Communicate the formal recommendation of acceptance or rejection of test deliverables to the Federal Student Aid Project Manager. • Prioritize change requests. • Manage the UAT process, monitoring the activity of test contractor, and participating in the TRR process; and providing a recommendation to move to the next phase of testing based on the TRR. • Provide required sign-off signature on Production Readiness Review (PRR) memorandums.
Development Team	• Design and perform Unit Test activities • Support subsequent testing efforts by analyzing change requests and updating code as required. • Review test artifacts. • Review and signoff of the MTP and any Phase Level Test Plans. • Review test reports to determine developments progress and impacts.

Title	Responsibilities
Test Lead	• Coordinate between the Federal Student Aid Test Manager and the Federal Student Aid Test Team. • Create, track, and maintain Phase Level Test Plans in order to assure completion of testing within time, cost, and quality constraints. • Identify risks in testing and creating mitigation strategies, and taking actions in conjunction with the Test Manager and the Project Manager. • Participate in analyzing and reviewing requirements for clarity, understanding, and testability. • Assign tasks and reviewing deliverables. • Manage escalations from the Federal Student Aid Test Manager and Federal Student Aid Test Team. • Conducting Test Readiness Reviews (TRR) • Work with the Federal Student Aid Requirements Liaison to resolve issues. • Create testing artifacts including: o Phase Level Test Plan o Phase Test Results o Phase Test Defects Reports o Phase Test Metrics o Test Summary Report • Communicate the formal recommendation of acceptance or rejection of test deliverables to the Federal Student Aid Project Manager. • Manage UAT process, monitoring activity of test contractor, and participating in TRR process.

Title	Responsibilities
Testers	• Design and/or provide input to, and/or maintaining Phase Level Test Suites, Test Cases and Test Scripts. • Perform Peer Reviews. • Review the Phase Level Test Readiness Reviews. • Review test scripts created by contractors. • Execute phase level tests, and validating test results. • Enter defects into a central defect repository and verifying their resolution. • Report escalations to the Federal Student Aid Test Lead or Federal Student Aid Test Manager. • Participate in reviewing and analyzing initiative requirements for clarity, understanding, and testability. • Provide input or creating testing artifacts including: o Phase Level Test Suites, Test Cases and Test Scripts o Phase Level Test Results o Input for the Phase Level Test Defect Report
Post Implementation Testers	• Execute Post Implementation Test Suites. • Validate Post Implementation Test Results. • Enter issues into a central defect repository and verifying their resolution. • Report escalations to the Federal Student Aid Post Implementation Verification Team Leader. • Create testing artifacts including: • Document Post Implementation Test Results • Create Post Implementation Defect Report
Performance Test Team	• Performance testing normally occurs independently from application development. The Performance Test Team is responsible for creating the Performance Test Plan and Reports.
Security Test Team	• Security testing may be required to meet Certification & Accreditation (C&A) for the system. If C&A is not required, the System Security Officer is required to sign off on the test plans and data being used to ensure security is being addressed.
Business Analyst	• Provide clarification of the requriements for the application under development. • Create test artifacts as determined by the Federal Student Aid Test Manager on a project-by-project basis.

Title	Responsibilities
Contracting Officer	• Communicate the formal acceptance or rejection of test deliverables to the contractor.

The following table describes the roles and responsibilities of contractor staff in the testing process.

Table 6: Contractor Roles and Responsibilities

Title	Responsibilities
Project Manager	• Provide input into the feasibility of performing parallel System Testing and UAT (performing testing in both phases simultaneously) due to project constraints . Note: Parallel testing may cause risk to the project and the parallel testing decision must include a risk mitigation strategy. The final decision must include input from the Federal Student Aid Test Manager, Federal Student Aid Project Manager, and Contractor Test Manager. • Synchronize the project plan and project schedule with consideration of the testing requriements and risks • Develop an appropriate risk mitigation strategy to ensure that testing is successful. • Provide input to the MTP by supplying the baseline project management plan document and high-level design documents to the Test Manager. • Ensure quality reviews of the MTP and Phase Level Test Plans.

Title	Responsibilities
Test Manager	• Support UAT, which may include creating defect reports, etc. • Providing input into the feasibility of performing parallel System Testing and UAT (performing testing in both phases simultaneously) due to scheduling issues. • Note: Parallel testing may cause risk to the project and the parallel testing decision must include a risk mitigation strategy. The final decision must include input from the Federal Student Aid Test Manager, Contractor Project Manager, and Federal Student Aid Project Manager. • Create, maintain and implement the MTP and Phase Level Test Plans (if separate from the MTP). • Coordinate testing activities between the Test Team and the Project Manager. • Providing input into Test Readiness Reviews • Develop the recommendation for using the above-mentioned technique. • Participate in risk assessment and mitigation activities. • Develop an appropriate mitigation strategy to ensure that testing is successful, and if testing problems occur employing the above-mentioned technique. • Manage the testing contractor • Provide input in to the TRR
Development Team	• Perform Unit Test Activities. • Support subsequent testing efforts by analyzing change requests, and updating code, as required. • Provide input into the impact analysis • Create testing artifacts, including: o Unit Test Plans o Unit Test Suites and Test Scripts o Unit Test Results o Resolve Defects

Title	Responsibilities
Test Lead	• Design and maintain Phase Level Test Suites, Test Cases and Test Scripts. • Create, track, and maintain the phase level test plan in order to assure completion of testing within time, cost, and quality constraints. • Identify risks in testing and creating mitigation strategies with the Contractor Test Manager and the Contractor Project Manager. • Participate in analyzing and reviewing requirements for clarity, understanding, and testability. • Assign tasks and reviewing deliverables. • Manage escalations from the Contractor Test Manager and Test Team. • Prepare for and conduct TRRs. • Collaborate with the Federal Student Aid Requirements Liaison to resolve issues. • Create testing artifacts, including: o Phase Level Test Plan o Phase Test Results o Phase Test Defect Reports o Phase Test Metrics o Phase Test Summary Report • Manage the contractor test team and provide input in to the TRR.
Testers	• Design and maintain Phase Level Test Suites, Test Cases and Test Scripts. • Perform Peer Reviews. • Review the Phase Level Test Readiness Document. • Conduct phase level tests, and validating test results. • Enter defects into a central defect repository and verifying the resolution. • Report escalations to the Contractor Test Lead or Contractor Test Manager. • Participate in reviewing and analyzing initiative requirements for clarity, understanding, and testability. • Create testing artifacts including: o Phase Level Test Suites, Test Cases and Test Scripts o Phase Level Test Results o Input for the Phase Level Test Defect Report

Title	Responsibilities

Post Implementation Verification Testers	• Execute Post Implementation Test Suites.
	• Validate Post Implementation Test Results.
	• Enter defects into a central defect repository and verifying the resolution.
	• Report escalations to the contract Post Implementation Test Team Leader.
	• Develop testing artifacts including:
	  o Post Implementation Test Results
	  o Post Implementation Issues Report

Performance Test Team	• The Performance Test Team is responsible for creating the Performance Test Plan and Reports.
Security Test Team	• Security testing may be required to meet Certification & Accreditation (C&A) for the system. If the system is not required to perform C&A, the System Security Officer is required to sign off on production data that may be used during testing and should sign off on test plans to ensure security is being addressed.

3.4.2 Test Strategies

Test strategies involve the approach and methods that will be used to plan, conduct, and report testing activities. Organizational goals and objectives and/or specific application project goals and objectives may influence the selection of test strategies. The test strategy section of the plan will specifically address the following organizational test strategies and project test strategies:

Table 7: Test Strategies

Organization Test Strategy	Project Test Strategy
• Standardization of all software testing activities across projects • Building a strong testing organization • Evaluating the current staffing model • Setting up the service level agreements • Timely review of existing processes • Test automation effort in terms of percentage of tests automated • Handling of meetings and other organizational processes • Participate in Configuration Management process • Participate in Change Management process	• Test coverage • Test tools • Training requirements • Types of metrics used • Level of Regression Testing

3.4.3 Test Phases and Test Types

There are five testing phases that occur during application development and deployment. An important part of Test Planning is to identify test types that will be performed during each test phase. Table 1**Table 8** lists standard test types and corresponding testing phases in which the testing is either required or recommended:

Table 8: Standard Test Types

Standard Test Types	Unit	Integration	System	UAT	Post Implementation
Component Testing	▲	-	-	-	-
Integration Testing	●	-	-	-	-
Component Interface Testing	-	▲	-	-	-
Subsystem Testing	-	●	▲	▲	●
Functional Testing	-	-	▲	▲	▲
Intersystem Testing	-	-	▲	▲	●
Regression Testing	-	-	▲	▲	-
Performance Testing	-	-	●	-	-
Usability Testing	-	-	●	▲	●

Legend: ▲: Required, ●: Recommended

In addition to the standard test types shown in **Table 8,** depending on the nature of the project, there are specialized tests that may be required. The Test Team must consider the need for each of these specialized test types when preparing test plans. A discussion of specialized testing can be found in **Appendix C: Testing Techniques**.

Table 9: Specialized Test Types

Specialized Test Types	Unit	Integration	System	UAT	Post Implementation
508 Compliance Testing	●	●	▲	▲	-
Commercial off the Shelf (COTS) Testing	●	●	▲	▲	-
Documentation Testing	●	-	●	●	-
Evergreen Testing	-	-	▲	▲	●
Graphical User Interface (GUI) Testing	-	-	▲	●	●
Client Server Architecture Testing	●	●	●	●	-
Service Oriented Architecture Testing	▲	▲	▲	▲	▲
Web Based Architecture Testing	▲	▲	▲	▲	-

Legend: ▲: Required, ●: Recommended

In all testing phases and types, the approach should be to “break” the system (negative testing) in addition to demonstrating that the functionality works.

3.4.4 Test Coverage

Test coverage is a quality measure used in software testing. It describes the degree to which the source code of a program has been tested. To measure how well the program is exercised by a test suite, one or more *coverage criteria* are used. There are a number of coverage criteria, the main ones being:

- **Function coverage –** has each function in the program been executed?
- **Statement coverage –** has each line of the source code been executed?
- **Condition coverage** (also known as Branch coverage) – has each evaluation point (such as a true/false decision) been executed?
- **Path coverage –** has every possible route through a given part of the code been executed?
- **Entry/Exit coverage –** has every possible call and return of the function been executed?

The Test Manager, in conjunction with the Project Manager, may decide to capture test coverage in order to:

- Identify the portions of the application that tests have not exercised
- Accumulate coverage data over multiple runs and multiple builds
- Merge data from different programs sharing common source code
- Work with the unit testing tool to make sure that errors are found throughout the application
- Generate test coverage reports. These reports include, at a minimum, code location information, statement coverage, and condition coverage

Selection of a Test Coverage Tool to determine the test coverage provided is a test planning activity. The test Manager, in conjunction with the Project Manager and test team is responsible for selection of a test coverage tool. Before delivering a build to the Testing Team, the Development Team will configure the test coverage tool for the build to be tested. During execution, the Project Manager is responsible for ensuring the creation of periodic test coverage reports using the test coverage tool and providing them to the test team.

3.4.5 Test Resource & Schedule Estimation

Test estimation provides inputs for resource and schedule planning. The Test Manager will estimate the test effort using multiple variables, constants, and calculations. The following example describes an acceptable estimation methodology.

This method uses five variables as follows:

Table 10: Test Estimation Values

Variable	Description
Complexity Constant	The complexity constant provides a means of defining the additional effort required to perform moderately complex and complex tests. Using a factor of one for relatively simple tasks, a multiplier is used to increase the value of the complexity constant for more complex tasks. In the example below, a moderately complex task is estimated to require 20% more effort than a simple task (1+(1*20%)) =1.2 and a complex task is estimated to require 50% more effort than a simple task (1+(1*50%)) =1.5. Initially, these multipliers are pure estimates. As actual data is accumulated and analyzed, and as tools and processes evolve, these values should be reevaluated to determine whether the values should be adjusted to provide more accurate estimates.
Number of Requirements	The number of requirements represents the number of tasks that need to be completed. For this method to produce reasonable estimates, the definition of a requirement must be consistent from phase to phase within a project, as well as consistent from project to project.

Variable	Description
Basic Hours	Basic Hours is an estimate based on experience that is used to calculate the time required to perform a task.
Regression Ratio	The regression ratio is equal to the number of defects divided by the number of test cases. This value is initially estimated. As testing results are reported, the estimate is adjusted based on experience.
Regression Runs	Regression runs is the number of times a test was performed.

For example:

Step 1 – Determine Complexity Constant Values based on whether the complexity of the requirement to be tested is Low, Medium, or High.

Table 11: Complexity Constant Values

Complexity Constant Value	Description	Complexity Constant
Low	Relatively Simple	1
Medium	Moderately Complex	1.2
High	Complex	1.5

Step 2 – Apply the Complexity Constant Value to Calculate the Subtotal Hours.
Subtotal Hours = Number of Requirements * Complexity Constant * Basic Hours

Table 12: Calculate Estimated Subtotal to Complete Testing

Number of Requirements	Complexity Constant	1Basic Hours	2Sub-Total
A	B	C	A*B*C=D
30	1.5	2	90
10	1.2	2	24
30	1	2	60

Step 3 – Calculate the estimated time to complete testing.
Total Hours = Subtotal Hours * Regression Ratio * Regression Runs

Table 13: Calculate Estimated Time to Complete Testing

[2]Sub-Total	[3]Regression Ratio	[4]Regression Runs	[5]Estimated Hours
D	E	F	D*E*F=G
90	0.5	3	135 Hrs.
24	0.5	3	36 Hrs.
60	0.5	3	90 Hrs.
Total Estimated Hours			261 Hrs.

3.4.6 Components to Be Tested

Test Plans must include the list of components (functions or requirements) that will be tested or demonstrated by the test. The component list in the plan will include a high-level description of each component without providing technical details.

Traceability to requirements and design is required in the test plan to identify the test design specification associated with each feature and each combination of features. Federal Student Aid follows the **Institute of Electrical and Electronics Engineers (IEEE) Standard (Std.) 829-2008,** which instructs Test Managers to "specify the minimum degree of comprehensiveness desired. Identify the techniques that will be used to judge the comprehensiveness of the testing effort (e.g., determining which statements have been executed at least once). Specify any additional completion criteria (e.g., error frequency). The techniques to be used to trace requirements should be specified."

To ensure that all requirements are tested a draft Requirements Traceability Matrix (RTM) must be completed and delivered to Federal Student Aid prior to the start of testing. Test Suites should include requirements and provide full traceability from the test suite and test case to the requirements being tested at the step level. Developers and testers should be working from the same requirements documents. In addition, this documentation should be base-lined and under configuration management control. The integration between RQM and RequisitePro supports the creation and management of relationships between requirements and testing artifacts, including test plans and test cases. Standard RQM reports can be generated which show the traceability relationships between requirements and test plans and test cases. These traceability reports can be used to produce the Requirements Traceability Matrix required for delivery.

3.4.7 Components Not To Be Tested

This section of the Test Plan describes the list of components (functions or requirements) that will not be tested. The component list in the plan will include a high-level description of each component without providing technical details. Federal Student Aid follows IEEE Std. 829-2008, which includes the requirement to identify all features and significant combinations of features that will not be tested and the reasons.

3.4.8 Technical Stage Gate 2, Test Readiness Review

The Test Readiness Review (TRR) process is the first step in formal Integration Testing. The TRR functions as the second technical phase gate, signifying that the requirements for beginning the next phase of testing conditions have been met. The TRR must include communication of defects found in previous test phases to the test team involved in the next phase of testing.

The Test Lead / Manager is responsible for facilitating the TRR. The TRR is ultimately approved by the integrated project team leads.

Use the FSA sanctioned template to record TRR information (see Appendix D).

All TRRs must be saved and archived, even if the project is canceled prior to implementation.

Note: *Projects that follow Agile or some other iterative development approach may not require a TRR for every sprint. (see Appendix D.21 Agile Testing).*

Testing

When performing testing, the steps in the TRR process are as follows:

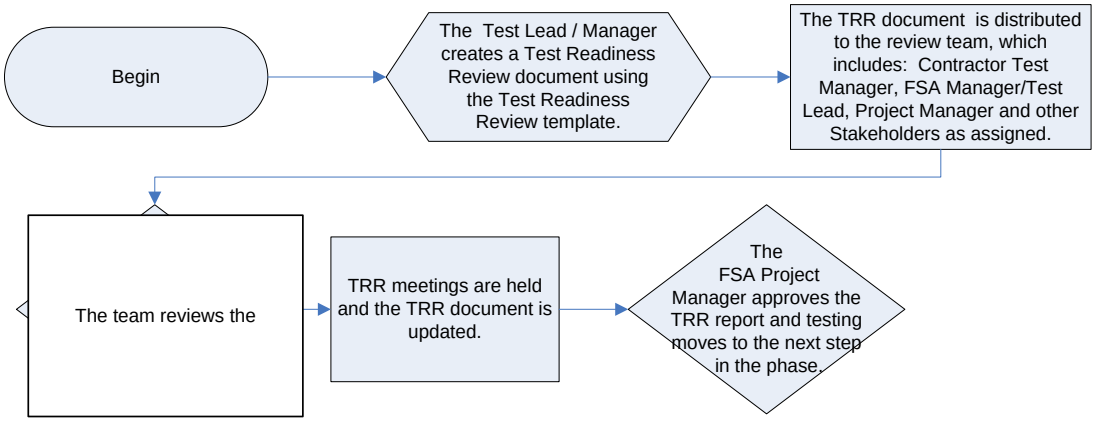


Figure 4: TRR Process Testing

3.4.9 Test Execution Cycle

Each planned test execution is a Test Execution Cycle. The number of Test Execution Cycles may change depending on the defects encountered in the system during the test execution.

The test execution cycle involves the following steps:

- Creation of a test suite which contains the test scenarios, test cases, test procedures, and test scripts
- Execution of all test cases in the test suite
- Reporting test results
- Defect management
- Implementation of a Test Suite may begin as soon as a test case is defined.

Note: An approved software build is usually required before executing a Test Suite to capture measurable test results.

3.4.10 Pass/Fail Criteria

Successful testing is based on pass/fail criteria. Federal Student Aid requires that results be clearly documented in the Test Suites. If the actual result is the same as the expected result the test passes; otherwise it fails. This section of the Test Plan contains the specific criteria that will be used for Pass/Fail assessment. In addition, dry runs of testing should be held by the test team prior to formal reviews with Federal Student Aid

to validate the expected results. The plan will include the following three steps related to Pass/Fail criteria:

- Establish precondition for the test.
- Execute the test.
- Verify the test result.

Note: *Expected results must be clearly documented in the Test Suites and the results should be verified during testing.*

3.4.11 Entrance/Exit Criteria

This section of the Test Plan contains the criteria that determine when testing within any given test phase begins (entrance criteria) and ends (exit criteria). Entrance and exit criteria must be specified for all test efforts.

3.4.12 Suspension Criteria and Resumption Requirements

In some cases, testing must be stopped based on certain conditions or situations. For example, if an application is under test and the login feature does not work, the tester will suspend testing activity and resume once the application has been fixed. Some conditions may cause all or a portion of the test activity to be suspended

3.4.13 Test Deliverables

This section of the Test Plan contains a list of artifacts, such as status reports and charts, required by the contract as formal test deliverables.

3.4.14 Environmental Needs

This section of the Test Plan includes the hardware details, test data details, simulators, and security related concerns, such as access, user accounts, ISSO's, and PII data descriptions. All other support required to perform testing in a designated environment should be mentioned in this section as well.

3.4.15 Staffing and Training Needs

This section of the Test Plan includes the number of resources and skills required for the testing project and the identification of areas where training is required. This section also identifies training requirements and how those training requirements will be fulfilled.

3.4.16 Schedule

This section of the Test Plan contains the testing schedule in the form of a work breakdown structure.

3.4.17 Metrics and Reporting Planning

The metrics that will be used to determine the success or failure of testing activities must be included in the Test Plan. Additional information related to test metrics and reporting can be found in **Section 5 Test Metrics and Reporting** and **Subsection 5.4 Metrics Identification and Selection** may be especially useful. Also refer to the Defect Management Test Management User Guide.

3.4.18 Tailoring the MTP or Phase Level Test Plan

When it becomes necessary to tailor a Test Plan that deviates from the direction in this Document, the Test Manager will provide the following information and submit it to the Federal Student Aid Test Manager for approval:

- Item number (for tracking each requested change)
- Applicable Document Section/Subsection number and title
- Requirement as stated in this document
- Tailored proposal
- Rationale for tailoring

3.5 Test Plan Reviews, Revisions, and Acceptance

Test Plan(s) are living documents that require frequent review and revision. Changes affecting the Test Plan may also affect major documents (such as test scenarios and scripts) associated with testing activities. All revisions of any lifecycle document must adhere to the required versioning control or change management to ensure that only the latest versions of documents are implemented in any lifecycle phase.

3.5.1 Test Plan Reviews

The Test Manager is responsible for reviewing and updating the MTP and Phase Level Test Plans (when required) for all other phases of testing prior to and during each test phase to ensure that the most up-to-date project-related information is included and considered from a testing perspective. The Federal Student Aid Project Manager must work with the Contractor to ensure that the planned Test Plan Reviews are line items in the overall project schedule. The review process includes:

- Distributing the updated Test Plan to the Federal Student Aid Test Manager for review and comment.
- The Federal Student Aid Project Manager will work with the Federal Student Aid Test Manager to assign a team to perform the review.
- The Federal Student Aid Test Manager will recommend acceptance of the test plan.
- Conducting a meeting with the assigned review team to discuss the input and identify changes.
- Incorporating changes and distributing the updated plan to the review team.
- Conducting additional reviews and revisions as necessary, until the review team agrees that the revised Test Plan meets Federal Student Aid’s needs.
- The “Formal Review” section of the RQM-based test plans should be used to document the review process including review comments, rejection, or acceptance of each test plan delivered to the Test Manager.

3.5.2 Test Plan Revisions

There will be numerous times during the test project where revisions to Test Plans and to other documents (i.e., test suites, test scenarios, test cases, test scripts, and test procedures) may be required. In most cases, an approved Change Request will be required in order to make the change. Refer to the FSA Enterprise Configuration Management User Guide for guidance on processing a change request. Reasons for revisions may include:

- Receiving new information
- Changes in the environment
- Development-testing actions
- Changes in schedules
- Addition of stakeholders

The Project Manager is responsible for ensuring that revisions are made promptly and that all relevant information associated with the changes is captured.

- The Project Manager will provide revised Test Plans and other test project related documents to the Federal Student Aid Project Manager and Federal Student Aid Test Manager within the period required by the contract, or as directed by the Federal Student Aid Project Manager or Federal Student Aid Test Manager.
- The Federal Student Aid Project Manager or Federal Student Aid Test Manager will review each revision for content and accuracy.
- All changes to baselined Test Plans and other test-related project documents will require notification to the test lead/manager and project manager and approval of updates will be required.

Note: *Change Requests may not be required for updating Test Plans in all cases. Examples of cases where Change Requests may not be required include Simple, Tier-3 projects, and Agile projects (see Appendix D.21 Agile Testing). Each project's Change Management Plan indicates whether modifications to Test Plans require Change Requests.*

3.5.3 Test Plan Approvals

The Federal Student Aid Enterprise Testing Contract Officer's Technical Representative provides a recommendation on the acceptance or rejection of the test plan to the Contract Officer.

4 Testing Phases

4.1 Overview

There are five testing phases required by Federal Student Aid; Unit Testing, Integration Testing, System Testing, User Acceptance Testing, and Post Implementation Verification. This section describes the required testing phases and their standards for application testing during development and after development.

The processes, requirements, and artifacts included in **Section 4** are the minimum required to meet Federal Student Aid Testing Standards. Additional procedures, requirements, and artifacts may be required to ensure compliance with the intent of this standards document.

Figure 5: Overview of Testing Phases, on the following page displays the overall testing process across each of the five test phases required by the Federal Student Aid. This figure also indicates the processes Federal Student Aid will use for evaluating the readiness of an application for production and when it occurs relative to the individual testing phases. These evaluation processes include the Production Readiness Review (PRR) and Post Implementation Verification.

For Agile and other iterative development projects, the test phases may be repeated. In these situations, phases such as Unit Testing and Integration Testing may be repeated multiple times before System Testing and UAT occur (see Appendix D.21 Agile Testing).

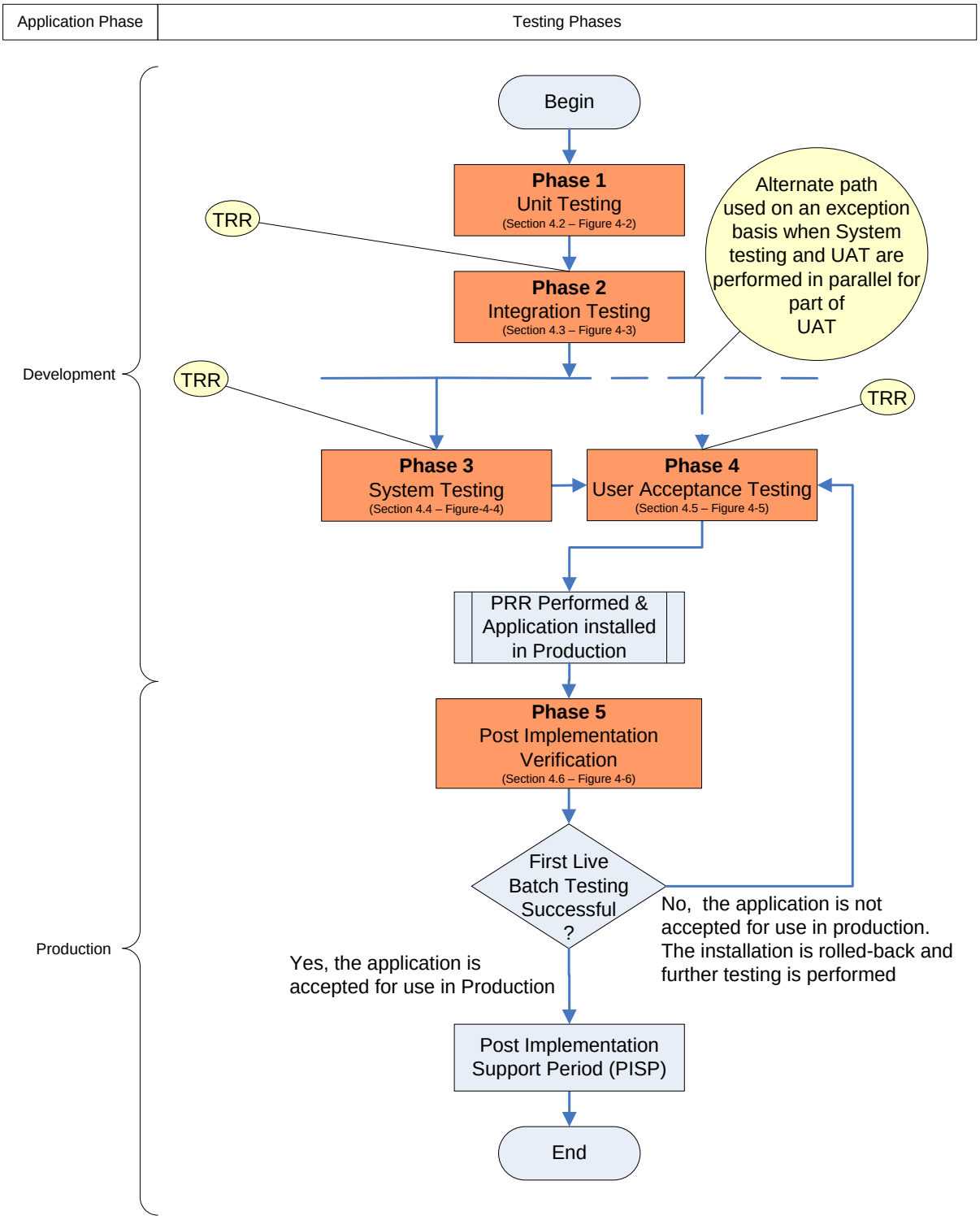


Figure 5: Overview of Testing Phases

4.1.1 Application Development Testing

During application development, four testing phases occur:

- Phase 1: Unit Testing
- Phase 2: Integration Testing
- Phase 3: System Testing
- Phase 4: User Acceptance Testing

Testing in these phases is required for new applications as well as maintenance. In addition, all four of the test phases apply to Commercial-off-the-Shelf (COTS) software, COTS software with modifications, and custom developed applications. The four test phases occur sequentially, with the possible exception of System Testing and User Acceptance Testing (UAT). These two phases may occur with some overlap with supporting documentation and approval from the Federal Student Aid Project Manager and Test Manager.

The tests performed and the related artifacts produced during these testing phases enable Federal Student Aid to evaluate new and upgraded software to determine whether the software is ready for use in a production environment.

Following UAT, key Test Team members will participate in the PRR to verify that the application is ready for the production environment.

Notes:

Testers will comply with the Privacy Act of 1974 as amended, 5 U.S.C § 552a, whenever test data is obtained from a data set containing personal information.

The Project and Test Managers will evaluate the applicability of each phase to each project. If a phase is not applicable, the test manager will document why it does not apply in the MTP. The Federal Student Aid Project and Test Managers will approve the approach.

4.1.2 Post Implementation Verification

Phase 5, Post Implementation Verification is the final test phase and occurs once the application is moved to production. Post Implementation Support Period Testing may occur during this phase.

Post Implementation Verification

4.1.3 Organization

Sections 3.2 through **3.6** are organized by phases; Phase 1 - Phase 5. Each phase contains subsections with the following information:

- Description of the testing that occurs
- Table of roles, responsibilities, and artifacts

- Process diagram
- Entrance Criteria that must be completed before the test phase starts
- Test preparation guidance
- Test execution information
- Test reporting requirements
- Exit Criteria that must be met before the phase is completed and further testing can begin

4.2 Phase 1: Unit Testing

During Unit Testing, developers design and perform tests to ensure that the code written performs according to the application's design. At the successful conclusion of Unit Testing, the Test Team recommends advancing the code to the next phase of development – Integration Testing. There are two types of Unit Testing:

- **Component Testing**: Component Testing is the testing of individual components of the code. A component is the smallest product of code that accomplishes a specific action.

  Component Testing is always required as part of Unit Testing

- **Unit Integration Testing**: Unit Integration Testing occurs when developers combine components so that several individual components are tested together.

  Unit Integration Testing is required if integrated components are created during development.

Federal Student Aid requires that Unit Testing be performed using selected path testing within the code module, as well as code coverage tools. The Federal Student Aid Project Manager will agree on the percent of the affected branch conditions and lines of code developers must test for each project. The development team is responsible for developing test drivers and stubs, as needed, to perform these tests. **Appendix D** of this document discusses useful tools, including code coverage tools.

An example for unit testing is described below:

Unit Testing Scenario

The "last name" field on the Free Application for the Federal Student Aid (FAFSA) application module may contain a maximum of 30 characters.

Test

The development tester validates that the "last name" field on the FAFSA application only allows 30 characters. The developer was able to enter 31 characters as a last name on the FAFSA application. The developer looks at the internal code to determine why more than 30 characters were allowed in the field and makes a correction to the code. The developer then retests the scenario to ensure that the "last name" field only accepts 30 characters.

Note:
The Federal Student Aid Project Manager and Federal Student Aid Test Manager must have insight into all Unit-Testing Activities.

Close interaction between the Project Manager and the Development Team is critical for Unit Testing to be successful and for the code to be advanced to the next testing phase. **Table 14: Unit Testing Roles, Responsibilities, and Artifacts,** shows the Project Manager and Development Team roles and responsibilities as required by Federal Student Aid and the artifacts that must be created for successful Unit Testing.

Table 14: Unit Testing Roles, Responsibilities, and Artifacts

Role	Responsibility
Project Manager	Plans for testing and testing related activities.
	Ensures that all Entrance Criteria are satisfied prior to beginning Test Preparation. Minimum Entrance Criteria include: • Baseline MTP • Baseline Detailed Design Document • Approved Requirements Traceability Matrix • Approved Unit Test Plan • Unit Test Environment • Coded software components that will be tested • Completed a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase
	Oversees testing activities.
	Reviews all testing artifacts.

Role	Responsibility
	Ensures that all Exit Criteria are satisfied prior to submitting Unit Testing Artifacts to the Federal Student Aid Project Manager. Minimum Exit Criteria include ensuring that: • Planned Unit Test Cases pass • Urgent and high priority defects identified during Unit Testing are resolved • Artifacts, reports, and metrics are updated by the developer to reflect the current state of the components tested • The Project Manager has reviewed and approved all artifacts • The Federal Student Aid Deliverable Review Process is completed
Federal Student Aid Test Manager	When a contractor is used, the Federal Student Aid Test Manager has the option of reviewing the Unit Test.
Federal Student Aid Architect	The Federal Student Aid Architect may review unit test results and may request the development team to perform additional testing.
Development Team	Includes the Federal Student Aid Project Manager in all project correspondence, meeting invitations, other communications, and reports.
	Prepares for Unit Testing by: • Updating the Unit Test Plan (as needed) • Creating and updating the Unit Test Framework (where required - e.g., stubs and drivers) • Creating and updating Unit Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform Component Testing and (if applicable) Integrated Component Testing • Creating and updating Unit Test Data that: o Is reusable when appropriate o Is generated utilizing an automated tool whenever possible o Includes identifying and planning for additional resources if test data is to be provided from a source outside the development group

Role	Responsibility
	Executes Unit Testing by: • Executing Unit Test Cases using the Unit Test framework • Evaluating Unit Test results • Performing Defect Management to verify, log, track and retest defects • Updating code to resolve defects
	Reports Unit Testing by: • Updating Unit Testing Test Suites • Updating Unit Testing Plans • Creating Unit Testing Reports • Creating Unit Testing Metrics • Creating the Unit Testing Defect Report • Submitting all artifacts to the Project Manager for review and approval

Figure 6 shows the Unit Testing process.

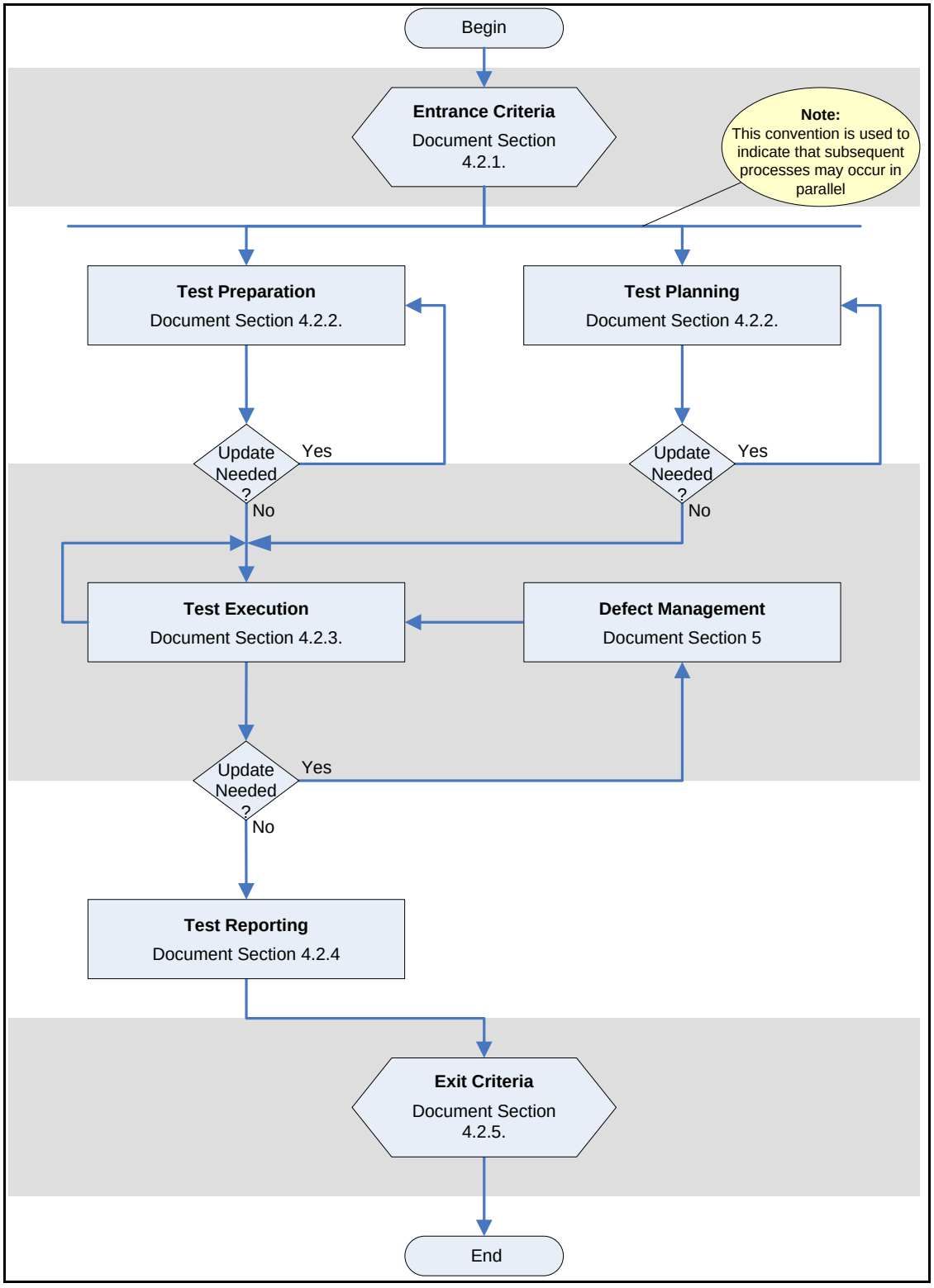


Figure 6: Process Overview of Unit Testing

4.2.1 Entrance Criteria

Before Unit Testing can begin, Entrance Criteria must be met. Federal Student Aid may define additional entrance criteria for a project above the Federal Student Aid minimum requirement. A contractor may also recommend extra criteria for Federal Student Aid approval; however, the Federal Student Aid Project Manager must approve any exceptions to the minimum criteria. Entrance criteria conditions include at a minimum:

- Baseline MTP
- Baseline Detailed Design Document
- Approved Requirements Traceability Matrix
- Approved Unit Test Plan
- Unit Test Environment
- Coded software components that will be tested
- Completion of a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase

4.2.2 Test Preparation

Along with the Entrance Criteria, the Development Team must prepare for testing by:

- Updating the Unit Test Plan (as needed)
- Creating and updating the Unit Test Frameworks (where required [e.g. stubs and drivers])
- Creating and updating Unit Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform Component Testing and, if applicable, Unit Integration testing
- Identifying and planning for additional resources if test data is to be provided from a source outside the development group
- Creating and updating Unit Test Data that:
    - Is reusable when appropriate
    - Is generated utilizing an automated tool whenever possible

4.2.3 Test Execution

Once the test preparations are complete, the Development Team performs Unit Testing by:

- Executing Unit Test Cases using the Unit Test frameworks
- Evaluating Unit Test results
- Performing Defect Management to verify, log, track and retest defects
- Updating code to resolve defects

Note: *Useful testing techniques during Unit Testing include, but are not limited to, White Box Testing, Code Review and Walkthrough, API Testing, Automated Testing, Regression Testing, and Security Testing. The* ***Glossary in Appendix B*** *provides descriptions of each technique.*

4.2.4 Test Reporting

The Development Team prepares the following series of reports and submits them to the Project Manager for review and approval. Additional reports may be required for a specific project.

- Updated Unit Test Suites
- Updated Unit Test Plans
- Unit Test Reports
- Unit Test Metrics
- Unit Test Defect Report

4.2.5 Exit Criteria

Exit Criteria define the quality standards that qualify the code for promotion to the next level or development stage. The following conditions are the minimum Exit Criteria for Unit Testing cases. (Federal Student Aid and/or the testing contractor may define additional Exit Criteria for a project – Exceptions must be approved by the Federal Student Aid Project Manager.)

- All Unit Test Cases pass
- All urgent and high priority defects identified during Unit Testing are resolved
- Deferred defects documented and included in summary counts
- All artifacts (including test suites and automated test scripts), reports, and metrics are updated by the development team to reflect the current state of the components tested
- The Project Manager has reviewed and approved all artifacts
- The Federal Student Aid Deliverable Review Process was applied to each artifact
- In all cases, the Federal Student Aid Test Manager and the System Architect may review the test results and may require additional testing

4.3 Phase 2: Integration Testing

Once Unit Testing is complete, the integration team integrates software units and code modules to create higher-level system components creating a complete system (application).

Integration testing is always required by the development team but formal integration testing may be required for complex new development. The Federal Student Aid project manager must determine when formal Integration testing is required based on the

criticality, information sensitivity, complexity, and cost of a system. If formal integration testing is required, a request must be made to complete the necessary Defect Management administrative updates to accommodate the request.

Integration Testing Test Types include:

- **Component Interface Testing:** Tests the interfaces between code modules and software units.

Note: *Integration Testing continues until developers have integrated all of the code to create the final application. Federal Student Aid requires integration testing for applications*

- **Subsystem Testing**: Required if Subsystems are created during development

As integrated components are created, Testers can begin Component Interface Testing, testing the interfaces between code modules. Depending upon how the design requirement is defined, Integration Testing continues until developers have integrated all of the code to create and test the final application.

Note:
In large applications, these components can be integrated to form subsystems that are integrated with other subsystems to create the completed application.

Integration Testing is an iterative process occurring with components integrated again and again until the application is completed. Once the first Integration Testing cycle is complete, the Test Manager will modify the Integration Test Plan and resource estimates as needed for subsequent Integration Testing Cycles. To facilitate defect resolution, testers should have an open communication path with developers prior to and during Integration Testing.

An example of an integration testing scenario is described below:

Integration Testing Scenario
The FAFSA website has a feature to allow the user to perform a search of schools by state.

Test

The integration tester enters the state code “VA” in the "Federal School Code” search field. The tester validates that the FAFSA website and the Federal School Code search feature work together to return only Virginia schools as a result on the FAFSA web page.

The Integration Test Team, including the Test Manager and Integration Testers, plans the testing activities for the interfaces to be tested. The Integration Test Plan includes the assigned tasks, schedule, and resources. Planning must include review of the Unit Component Test Defect Report and the Unit Integration Test Defect Report to identify problems and areas of concern that occurred during Unit Testing and that may need additional scrutiny during Integration Testing.

Table 15 shows the Project Manager, Test Manager, and Integration Test Team roles and responsibilities as required by Federal Student Aid, and the artifacts that must be created for successful Integration Testing.

Test Suites that have been approved by Federal Student Aid may be changed during test execution, but must be reported to the Federal Student Aid test manager indicating the reason for the change. The Federal Student Aid project team will review the updated test suites to ensure the test meets the requirements. When analysis of defects leads to requirement clarification or modification, new Test Suites must be created and approved by Federal Student Aid.

Table 15: Integration Testing Roles, Responsibilities, and Artifacts

Role	Responsibility	Artifact(s)
Project Manager	Provide information for testing and testing related activities	• Updated Project Management Plan • Updated Work Breakdown Structure
Federal Student Aid Test Manager	Completion of the Federal Student Aid Deliverable Review Process	• Federal Student Aid Test Manager sign-off after reviewing and approving artifacts
	If a contractor is used, the Federal Student Aid Test Manager is responsible for the following: • Reviewing and approving test artifacts • Monitoring testing activities	• Contractually related recommendation to accept or reject test artifacts.

Role	Responsibility	Artifact(s)
Test Manager	Ensure all Entrance Criteria are satisfied prior to beginning Test Preparation. Minimum Entrance Criteria include having: • Baseline MTP • Detailed Design Document • Approved Requirements Traceability Matrix • Approved Integration Test Plan • Integration Test Environment • Coded Software components that will be tested • Integrated code modules completed by Developers and ready to be tested • TRR documentation and process • Tests cases assigned to testers • Test Suites, Test Cases, Test Scripts, and Test Procedures created • Completed a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase • An Application login and password	• Baselined MTP • Completed TRR • Approved Integration Test Plan • Integration Test Environment exists
	Oversee testing activities	• Emails • Meeting Minutes • Weekly Status Reports
	Review all testing artifacts	• Reviewed Integration Test Cases • Reviewed Integration Test Results

Role	Responsibility	Artifact(s)
	Ensure that all Exit Criteria are satisfied prior to submitting Integration Testing Artifacts to the Federal Student Aid Project Manager. Minimum Exit Criteria include ensuring that: • All planned Integration Test Cases pass • All urgent and high priority defects identified during Integration Testing are resolved • All artifacts, reports, and metrics are updated by the Test Team to reflect the current state of the components tested • The Test Manager has reviewed and approved all artifacts	• Completed Integration Test Cases • Defect Report • Updated artifacts, reports, and metrics • Test Manager sign-off after reviewing and approving artifacts

Role	Responsibility	Artifact(s)
Integration Test Team	Prepare Integration Testing by: • Updating the Integration Test Plan (as needed) • Determining and obtaining the Test Environment • Determining and obtaining the Integration Test Framework required to execute Integration Test Cases • Creating and updating Integration Test Suites, including determining test scenarios, test scripts, and test procedures to perform Component Integration Testing, and, if applicable, Subsystem Testing • Obtaining logins and passwords to interface(s) • Generating Integration Test data that: • Is reusable when appropriate • Is generated utilizing an automated tool whenever possible • Is generated from data that has been processed on the system and that can be re-used for testing	• Updated Integration Test Plan (if needed) • Integration Test Framework (as needed, e.g., stubs and drivers) • Integration Test Suite (including test scenarios, test cases, test scripts, and test procedures) • Integration Test Data
	Execute Integration Testing by: • Executing Integration Test Cases using the Integration Test Framework • Evaluating Integration Test Results • Performing Defect Management to verify, log and retest defects	• Executed Integration Test Cases • Evaluated Integration Test results • Complete documentation of defects in the defect management tool • Updated code that resolves defects • Documentation to show that defects were tracked and managed to the resolution

Role	Responsibility	Artifact(s)
	Report Integration Testing by: • Updating Integration Testing Test Suites • Updating Integration Testing Plans • Creating Integration Testing Reports • Creating Integration Testing Metrics • Creating the Integration Testing Defect Report • Submitting all artifacts to the Project Manager for review and approval	• Updated Integration Test Suites • Updated Integration Test Plan • Integration Test Summary Report • Integration Test Metrics • Integration Test Defect Report

The contractors responsible for Integration Testing will ensure that tests are performed on all components that make up the integrated components of the application. At each step of the integration process, the test must verify the functionality and reliability of the application according to approved design specifications.

Figure 7 is a process flow diagram that introduces the high-level activities within Integration Testing. Additional detail for each activity is located in this section.

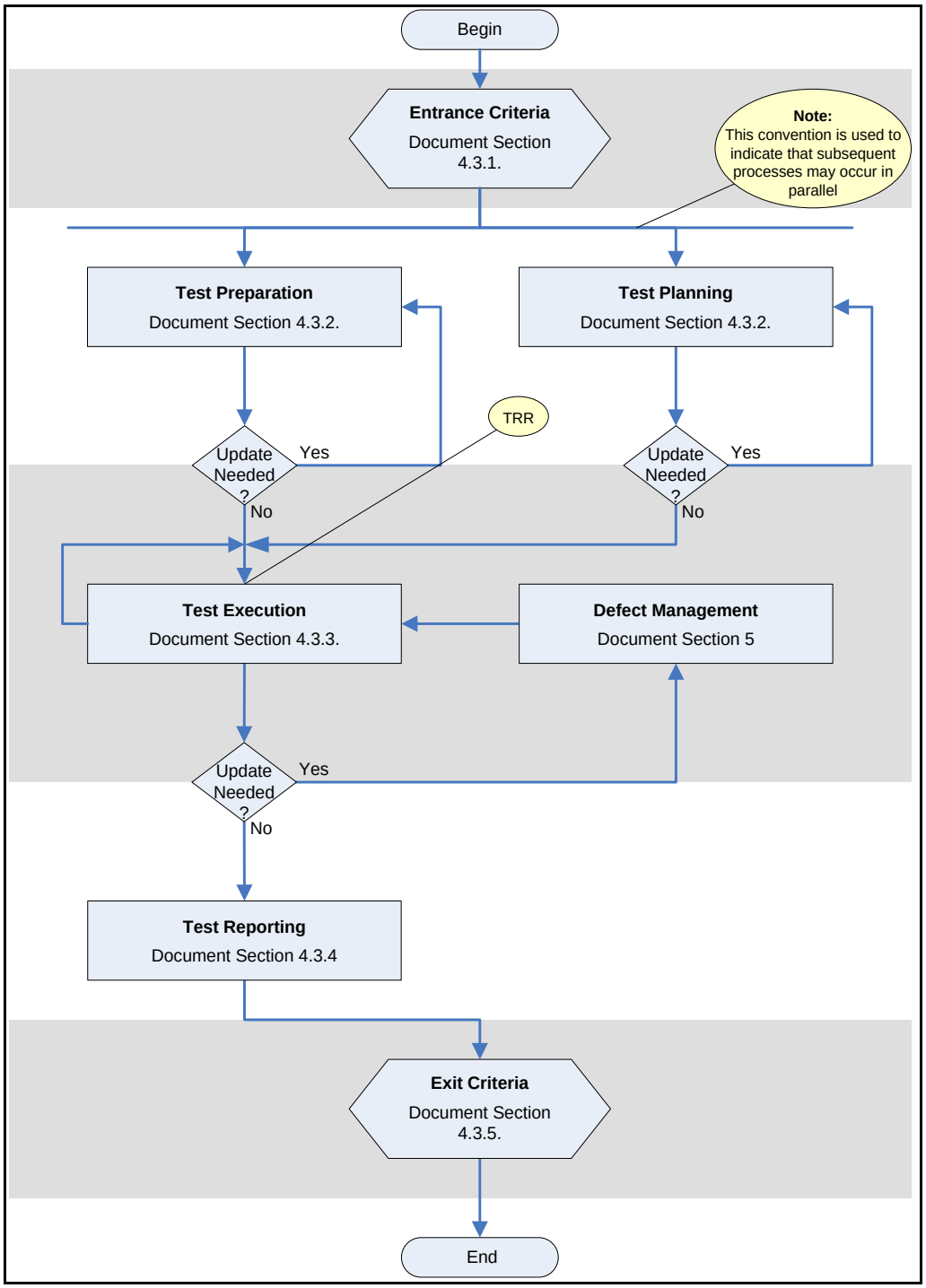


Figure 7: Process Overview of Integration Testing

Note: *The Integration Testing roles and process are tailored for Agile projects to fit the Agile team and iterative development lifecycle approach. See Appendix D.21 Agile Testing for more information.*

4.3.1 Entrance Criteria

The following conditions are the minimum Entrance Criteria that must exist before Component and Subsystem Integration Testing can begin. (Federal Student Aid and/or the Testing Contractor may define additional Entrance Criteria for a project – Exceptions must be approved by the Federal Student Aid Project Manager.)

- Baseline MTP
- Baseline Detailed Design Document
- Approved Requirements Traceability Matrix
- Unit Testing of components is complete and all urgent and high defects found have been addressed and resolved
- Integrated code modules have been created and are ready to be tested
- Integration Test Environment has been created
- Integration Test Plan has been created
- Test Readiness Review has been completed
- Tests have been assigned to Testers
- All Test Suites, Test Cases, Test Scripts, and Test Procedures have been created
- All change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase, have been reviewed
- An Application login and password have been assigned
- Other criteria may be required based on the needs of the project

4.3.2 Test Preparation

The Integration Testing Team is responsible for performing the following tasks before and, as needed, during Component Integration and Subsystem Testing:

- Updating the Integration Test Plan (as needed)
- When applicable, the Integration Test Plan must include communication guidelines between each subsystem. The Test Manager or Project Manager of the application under test is responsible for working with the Test Manager and Project Manager of the other subsystem to coordinate this testing phase.
- Determining and obtaining the Test Environment
- Determining and obtaining the Integration Test Framework required to execute Integration Test Suites
- Creating and updating Integration Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform Component Integration Testing and, if applicable, Subsystem Testing

- Obtaining logins and passwords to interfaces
- Generating Integration Test data that:
- Is reusable when appropriate
- Is generated utilizing an automated tool whenever possible
- Is generated from data that has been processed on the system and that can be re-used for testing

4.3.3 Test Execution

Integration Test execution tasks performed by the Integration Test Team include:

- Executing Integration Test Cases using the Integration Test Framework
- Evaluating Integration Test Results
- Performing Defect Management to verify, log, and retest defects

***Note**: Testing techniques useful during Integration Testing include, but are not limited to, White Box Testing, Code Review and Walkthrough, Regression Testing, Communication Testing, and Security Testing. The Glossary in Appendix B provides descriptions of each technique.*

4.3.4 Test Reporting

Integration Test reporting includes, but is not limited to, the following:

- Updated Integration Testing Test Suites
- Updated Integration Testing Plans
- Integration Testing Reports
- Integration Testing Metrics
- Integration Testing Defect Report

4.3.5 Exit Criteria

Exit Criteria define quality standards that qualify the code for promotion to the next level or development stage. The following conditions are the minimum Exit Criteria for Integration Testing cases. (Federal Student Aid and/or the Testing Contractor may define additional Exit Criteria for a project – Exceptions must be approved by the Federal Student Aid Project Manager.)

- All Integration Test Cases have been executed
- All urgent and high priority defects identified during Integration Testing are resolved
- Deferred defects documented and included in summary counts
- All the artifacts (including test cases and automated test scripts), reports, and metrics are updated by the testing team to reflect the current state of the interface(s)

- The Test Manager has reviewed and approved all artifacts
- The Federal Student Aid Deliverable Review Process is completed
- Test coverage reported to Federal Student Aid
- When a contractor is used, the Federal Student Aid Test Team reviews the contractor's artifacts. The Federal Student Aid Test Manager recommends acceptance or rejection to the Federal Student Aid Project Manager and Contracting Officer and may require the contractor to perform additional testing.
- Other criteria may be required based on the needs of the project

4.4 Phase 3: System Testing

Following Integration Testing, Testers perform System Testing. System Testing evaluates the integrated system (application) as a whole. The Testing Team performs tests to ensure that each function of the system works as expected and that any errors are documented, analyzed, and resolved appropriately. The structure of the contracts will determine how the testing procedures in this section are applied to each contractor.

System Testing Types include:

- Functional Testing (Required)
- Regression Testing (Required)
- Intersystem Testing (Required if applicable)
- Performance Testing (Required if applicable)
- Non Functional Testing (Required if applicable)

Specialized Testing Types applicable during System Testing include:

- 508 Compliance Testing (Required if applicable)
- Security Testing (Required if applicable)

Regression Testing is performed during System Testing to ensure that changes made to the application do not adversely affect existing functionality. (*Note: Regression Testing can also be performed to ensure that changes to other systems have not had an adverse effect on the system under test.*) Performance Testing is dependent on the satisfactory completion of all other planned System Test types. The Entrance Criteria, Test Preparation, Test Execution, Test Reporting, and Exit Criteria for Performance Testing are contained in **Section 3.6.1**.

The System Testing Team, including the Test Manager and System Testers, plans System Testing activities. The Test Manager will plan the types of tests that the Test Team will perform. To the extent that these test types are not interdependent for a particular application, tests may be performed sequentially, in any order, or if staffing permits, in parallel. The System Test Plan includes the assigned tasks, schedule, and

resources. Planning must include review of the Unit Testing Defect Reports and Integration Testing Defect Reports to identify problems and areas of concern that occurred in previous testing phases and that may need additional scrutiny during System Testing. When System Testing includes Intersystem Testing, the Project and Test Managers are responsible for coordinating testing activities with the major players of subsystems that may be affected by the testing activities.

End to end test scenarios must use the same data from the beginning to the end of the test. Federal Student Aid requires verification points to be contained within the same test suite. For instance, if an end to end test requires data entry, database updates and creation of reports, all three tests must be performed using the same input starting with data entry, reviewing of the database updates, and analyzing the reports to ensure that the data is correct.

Below is an example of a system testing scenario:

System Testing Scenario

The FAFSA website has a feature to allow the user to perform a search of schools by state.

Test

The system tester enters the state code “VX” in the "Federal School Code" search field. The tester validates that the FAFSA website returns an appropriate error message since “VX” is an invalid state code.

Table 4-3, System Testing Roles, Responsibilities, and Artifacts, summarizes Federal Student Aid’s minimum requirements for the roles, responsibilities, and artifacts related to System Testing.

Test Suites that have been approved may be changed during test execution after notification is given to the project team as to the reason for the change. When analysis of defects leads to requirement clarification or modification, then new Test Suites must be created and approved by Federal Student Aid project team.

Table 16: System Testing Roles, Responsibilities, and Artifacts

Role	Responsibility	Artifact(s)
Project Manager	Provide information for testing and testing related activities	• Updated Project Management Plan • Updated Work Breakdown Structure

Role	Responsibility	Artifact(s)
Federal Student Aid Project Manager (Technology Office Project Manager)	If a contract is involved, work with the Federal Student Aid Test Manager, and Contractor Project Manager, and Contractor Test Manager, to complete the following: • Evaluate any need to perform parallel System Testing and UAT (performing testing in both phases concurrently, although not recommended) • Develop an appropriate mitigation strategy to ensure that testing is successful • Keep track of test efforts to ensure that the schedule is being met • Communicate recommendation of acceptance or rejection of the test artifacts to the contracting officer (only when the PM is the COTR for the contract) If Intersystem Testing is performed, coordinate with representatives of other applications that interface with the system under development to establish schedules for Intersystem Testing.	• Approved parallel testing plan and related mitigation strategy
Contract Project Manager	Work with the Federal Student Aid Project Manager, Federal Student Aid Test Manager, and Contractor Test Manager to evaluate the need to perform parallel System Testing and UAT (performing testing in both phases concurrently) for part of each testing phase and to develop an appropriate mitigation strategy to ensure that testing is successful.	• Approved parallel testing plan and related mitigation strategy

Role	Responsibility	Artifact(s)
Test Manager (FSA or Contractor)	Ensure that all Entrance Criteria are satisfied prior to beginning Test Preparation. Minimum Entrance Criteria include having: • Baseline MTP • Baseline Detailed Design Document • Approved Requirements Traceability Matrix • Approved System Test Plan • System Test Environment • Integration Testing completed • System created by the developers that performed the integration ready to be tested • Application login and password assigned • Interface Control Document for Intersystem Testing • Intersystem Testing activities coordinated • The Test Readiness Review completed • Test cases assigned to testers • All Test Suites, Test Cases, Test Scripts, and Test procedures created • Completed a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase • Defect tracking tool is available and operational Works with the Federal Student Aid Project Manager, Contractor Project Manager, and Federal Student Aid Test Manager to evaluate the need to perform parallel (although not recommended) System Testing and UAT Develops an appropriate mitigation strategy to ensure successful testing Follows up on outcomes of the TRR	• Approved Requirements Traceability Matrix • System Test Plan created and approved • System Test Environment exists • Defect Reports • Completed to approve TRR • Test Summary Report

Role	Responsibility	Artifact(s)
	Oversee testing activities	• Email • Meeting Minutes • Status Reports
	Review all testing artifacts	• Reviewed System Test Cases • Reviewed System Test Results
	Ensure that all Exit Criteria are satisfied prior to submitting System Testing Artifacts to the Federal Student Aid. Minimum Exit Criteria include ensuring that • All planned System Test Cases pass • All urgent and high priority defects identified during System Testing are resolved • All artifacts, reports, and metrics are updated by the Test Team to reflect the current state of the tested system • The Project Manager has reviewed and approved all artifacts • Federal Student Aid Deliverable Review Process completed	• Completed System Test Cases • Defect Report Updated artifacts, reports, and metrics Test Manager sign-off after reviewing and approving artifacts • Federal Student Aid Test Manager sign-off after reviewing and approving artifacts

Role	Responsibility	Artifact(s)
Federal Student Aid Test Manager	• If a contract is involved, work with the Federal Student Aid Test Manager, and Contractor Project Manager, and Contractor Test Manager, to complete the following: • Evaluate the need to perform parallel System Testing and UAT (performing testing if both phases concurrently, although not recommended) • Develop an appropriate mitigation strategy to ensure that testing is successful • Review test artifacts, provide feedback • Communicate recommendation of acceptance or rejection of test artifacts	• Sign-off on test artifacts • Comments matrix for reviewed test artifacts
Development Team	• When testing with other applications is required for Intersystem Testing, the development team must provide the Federal Student Aid test team the Interface Control Documents at least one month prior to the Intersystem Testing effort unless the contract states otherwise. • Perform defect analysis	• Interface Control Document • Update to defect reports and impact analysis documentation

Role	Responsibility	Artifact(s)
System Test Team	Prepare System Testing by • Updating the System Test Plan (as needed) • Determining and obtaining the System Test Environment • Determining and obtaining the System Test Framework required to execute System Test Cases • Creating and updating System Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform each of the types of tests they will conduct during System Testing • Obtaining logins and passwords to interface(s) • Generating System Test data that: • Is reusable when appropriate • Is generated utilizing an automated tool whenever possible • Is generated from data that has been processed on the system and that can be re-used for testing	• Updated System Test Plan (if needed) • System Test Framework • System Test Suite (including test scenarios, test cases, test scripts, and test procedures) • System Test Data
	Execute System Testing by • Executing System Test Cases using the System Test Framework • Evaluating System Test Results • Performing Defect Management to verify, log and retest defects	• Executed System Test Suites • Evaluated System Test results • Complete documentation of defects in the defect management tool • Updated code that resolves defects • Documentation to show that defects were tracked and managed to the resolution

Role	Responsibility	Artifact(s)
	Report System Testing by • Creating the System Testing Defect Report • Submitting all artifacts to the Project Manager for review and approval	• Updated System Test Suites • Updated System Test Plan • System Test Summary Report • System Test Metrics • System Test Defect Report

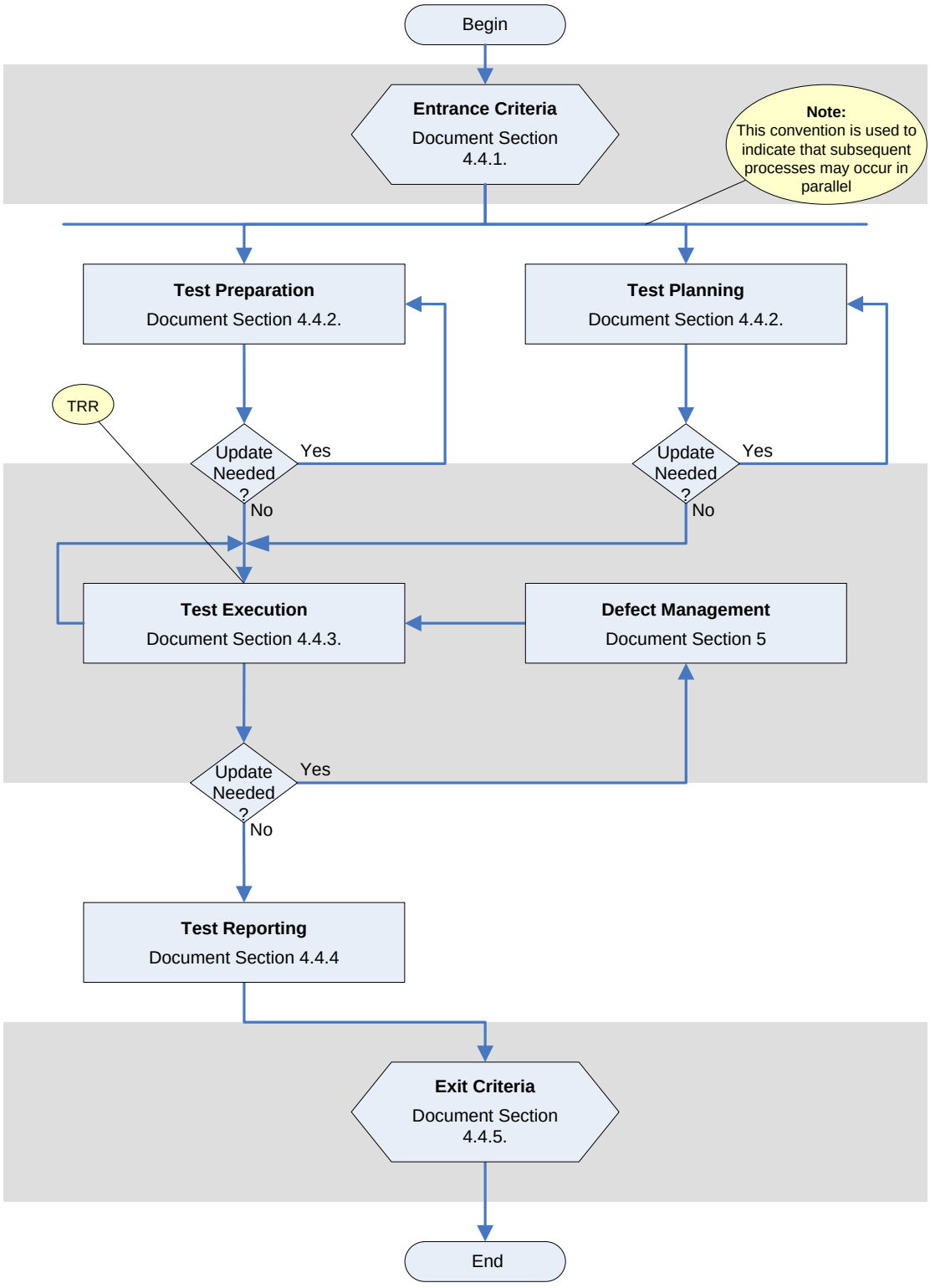


Figure 8: Process Overview of System Testing

Note: *The System Testing roles and process are tailored for Agile projects to fit the Agile team and iterative development lifecycle approach. See Appendix D.21 Agile Testing for more information.*

4.4.1 Entrance Criteria

The following conditions are the minimum Entrance Criteria that must be met before System Testing can begin:

- Baseline MTP
- Baseline Detailed Design Document
- Approved Requirements Traceability Matrix
- If applicable, Integration Testing is complete and all urgent and high errors found have been addressed and resolved
- The code deployed in environment by the developers that performed the integration is ready to be tested
- The application login and password have been assigned
- System Test Environment has been created
- System Test Plan has been created
- Test Readiness Review has been completed
- Tests have been assigned to Testers
- All Test Suites, Test Cases, Test Scripts, and Test Procedures have been created
- All change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase, have been reviewed
- Other criteria may be required based on the need of the project

Note: *These conditions must include consideration for each of the test types to be performed during System Testing.*

4.4.2 Test Preparation

The System Testing Team is responsible for performing the following tasks before and, as needed, during System Testing.

- Updating the System Test Plan (as needed)
- Work with the development team and infrastructure team to ensure that the test environment will be ready
- Determining and obtaining the System Test Framework required to execute System Test Suites

- Determining the guidelines between each application for Intersystem Testing. The Test Manager or Project Manager of the application under test is responsible for working with the Test Manager and Project Manager of the other application to coordinate this testing phase
- Creating and updating System Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform each of the types of tests
- Conduct peer review of System Test Suites to identify defects and improvements
- Obtaining logins and passwords to interface(s)
- Generating System Test data that:
  - Is reusable when appropriate
  - Is generated utilizing an automated tool whenever possible
  - Is generated from data that has been processed on the system and that can be re-used for testing

4.4.3 Test Execution

System Test execution tasks performed by the System Test Team include:

- Executing System Test Suites
- Evaluating System Test Results
- Performing Defect Management to verify, log and retest defects

***Notes**: Testing techniques useful during System Testing include, but are not limited to, Intersystem Testing, Black Box Testing, Walkthrough, Regression Testing, Communication Testing, Security Testing, and Peer Review of Test Suites. The **Glossary in Appendix B** provides descriptions of each technique.*

If contractor testers are to perform Intersystem Testing during System Testing, files are created for transmission to business partners, and files are desk checked to ensure conformance to the recipient's formatting requirements. No data exchange will occur between the contractor and any party external to Federal Student Aid without the written authorization of the Federal Student Aid Project Manager.

If Intersystem Testing involves new applications or current applications containing new data interchange interfaces with stakeholders outside Federal Student Aid, exchange of data outside Federal Student Aid must not occur until Federal Student Aid has reviewed the application during UAT. The exception to this policy is that a limited exchange of data may take place under Federal Student Aid supervision following receipt of written approval from the Federal Student Aid Project Manager. Federal Student Aid will exchange test files with partners during UAT to complete Intersystem Testing.

4.4.4 Test Reporting

System Test reporting includes creating the following artifacts:

- **Summary of Phases by Iteration**: Summarizes the number of test suites that have been assigned and completed by iteration for a specific phase of testing.
- **Test Execution Report:** Summarizes test execution status of test cases associated with test suites for a given testing phase.
- **Test Status Report for Phase:** Provides a high-level status of testing for a specific phase
- Requirements Traceability Matrix
- Defects by Component
- Defect Detailed
- Defect Summary
- Defect Summary by Severity
- CR and Defect Summary
- Peer Review Report for Phase: Ensures all defects were captured and requirements coverage
- Project self-assessment/audit to ensure all planned tasks were completed.

4.4.5 Exit Criteria

Exit Criteria define quality standards that qualify the code for promotion to the next level or development stage. The following conditions are the minimum Exit Criteria for System Testing cases. (Federal Student Aid and/or the testing contractor may define additional Exit Criteria for a project – Exceptions must be approved by the Federal Student Aid Project Manager.)

- All System Test Cases have been executed.
- All urgent and high priority defects identified during System Testing are resolved. The Federal Student Aid Project Manager and Test Manager has discretion to add medium severities to this list.
- Deferred defects documented and included in summary counts.
- All the artifacts (including test cases and automated test scripts), reports, and metrics are updated by the testing team to reflect the current state of the interface(s).
- Test Manager has reviewed and approved all artifacts.
- Federal Student Aid Deliverable Review Process completed.
- Test coverage reported to Federal Student Aid.
- When a contractor is used, the Federal Student Aid Test Manager reviews the contractor's test results and accepts them, or requires additional testing.
- Other criteria may be required based on the needs of the project.

4.4.6 Performance Testing

The overall goal for any performance test project is to ensure that the application under testing operates as efficiently as possible. Performance testing is about balancing the needs of the customer with the resources available to the application owner.

The Federal Student Aid Project Manager and Test Manager must coordinate Performance Testing activities with the Enterprise Performance Testing Team in the early stages of a project. The Enterprise Performance Testing Team maintains documentation that details the process.

Typically, the Performance Test Procedures are comprised of four phases. The phases are planning, development, execution, and documentation.

1. **Planning**
    - Discovery – the performance test team reviews requirements (business purposes, processes, and uses) and constraints of the system being tested, identifies stakeholders and their needs, determines the schedule, and determines the general usage patterns.
    - Requirements – the performance test team documents the efficiency of the application based on the best balance of the application owner's needs and the customers' needs. This allows the performance test team to set clearly defined goals for the test.
    - Scheduling – the performance test manager determines the schedule, specific tasks, resources needed and the length of time for each task required for a successful performance test. In addition to the performance test team, availability of the development and support staff must be considered.
2. **Development**
    - Scripts – development of scripts depends on the business processes, which are then translated into steps for a virtual user to emulate a customer using the application during a performance test. The scripts selected must be those scenarios executed most frequently by the majority of users. All scripts are verified by peer reviews and mock load tests (10 users executing the script 10 times).
    - Scenarios – development of scenarios is based on information from project documentation and subject matter experts. If the tool used for performance testing is Load Runner, user groups and load patterns are created.
3. **Execution**
    - Testing – once the environment setup is complete, performance testing begins. An iterative approach is used during the test cycle. The scenarios and results are logged, development and support staff are notified of issues, changes may be made to the code or architecture, stakeholders are notified when test efforts are complete, and the daily *Test Summary Execution Report* (log) is published.

Note: *Daily meetings are held to: describe the results of the daily report, anomalies found, review of all issues, assignment of issues, assessment of impact and risks to the performance test schedule, which may impact the overall delivery schedule, identify and confirm the primary focus of the performance test, and agree on the strategy for the next performance test run.*

- Tuning and Analysis – this process requires thorough analysis of data, teamwork (performance test team, project management, development team, and support staff) and constant communication. Views will be created for analyzing data, which may be web server health, application server health, application server session health, etc. The data should be created in a manner in which a novice will be able to review it and notice that something is amiss. All unexplained anomalies in the data must be identified, classified as problematic or symptomatic, and addressed. This is an iterative process. During this process, key metrics are assessed for performance, root cause of unexpected or undesirable behaviors of the system are analyzed, and all that has transpired during this process is documented.

  Note: *Corrective action recommendations are presented to the stakeholders; Federal Student Aid project management approves the changes to be implemented.*

- Reporting – the report provides information for peer review that can assist in validating the tasks performed as well as information about the problems encountered during the test and how they were resolved. Lastly, it provides guidance for improving the accuracy of future performance testing for the application under test. The report includes both failed and successful test runs. Keep in mind that additional information may be required based on the type of system undergoing performance testing.

  Note: *Documenting results of the performance test is required. At a minimum, the report must contain the objective of the report that states why the performance test was needed, the 5 W's: (1) who: the members of the performance and development teams involved with the test; (2) when: what time each task was executed; (3) where: what was the state of the environment and the test suite where the test was run; (4) why: why was the task performed, what changes brought about the tasks and how the test was verified; and (5) what: what did the task accomplish, what were the resulting discoveries, changes and follow-up tasks and the conclusion stating whether or not the objectives of the test were met.*

4. Documentation

- Readiness – this section of the final test summary report describes the expected behavior of the application under test at peak loads and identifies the point at which the application starts experiencing performance problems that may be due to availability of the system or transaction failures. This section gives management the information needed to deploy the application, which includes worst case scenarios, so they are prepared to address those issues if they occur.

Bottom line, this section gives the go/no-go recommendation from the performance team based on the goals and the results of the test.

- Capacity – this section of the final test summary report describes the capacity planning done during the test and the run data on which it was based. Historical information on usage, projected usage, recorded data and monitoring information from peak runs from each run is included. Capacity based on run data and projected usage patterns and historical data are also included in this section.

  Note: *The application development and support teams will review and accept this section.*

This section is not mandatory and will be dictated by the type of project being tested.

- Configuration – this section of the final test summary report describes the steps to replicate the tuning done in the performance environment. Specific findings or anomalies are detailed and the corrective actions for each are included. All configuration details are also included. These corrective actions are not completed by the performance team but by the support team of the performance environment or the application development team.
- Findings – this section is the executive summary of the final test summary report, written with management in mind, and provides full disclosure of the facts. It describes the testing achieved, issues, and how the issues were resolved. The section will provide a chronology of the test and significant results, the worst and median of all the 90% response times, provide details on poor performing transactions, significant changes to the application or environment due to the findings, projected capacity, and list all the teams involved in the test effort.

  Note: *The Federal Student Aid Performance Test Manager and the Performance Test Team approve the document prior to release to the Federal Student Aid Project Manager.*

  The primary testing requirement is to have an established, test ready environment that mirrors the environment of the system under test. Mirroring production provides the most accurate testing results, but due to complexity, budget restrictions, and testing goals, this may not always be possible. Test plans must include the environment expectations, and the environment must be validated before testing begins.

  Planning for performance testing should start during the development of the system. However, since the system being developed is in flux, the planning will not be complete until after the initial test of the system is complete. A change in requirements may lead to a change in the business processes to be performance tested.

  The performance test team must coordinate testing with the project manager and the development team. Performance testing should not begin until all features of the system have been functional tested, and there are no open issues or defects

that impact the functionality of the system. Ideally, the component being tested should be finalized and have passed functional and user acceptance testing.

Note: *Parallel testing is allowed but must be agreed to by Federal Student Aid project management, and the following milestones must be completed: Performance Test Plan completed and approved, System Test of the system completed (if there is more than one cycle of System Testing, then the first cycle of system testing must be completed), and scripting of Performance Test Scenarios are completed.*

Performance testing requires the active and responsible participation of members of the development and application support staff. In the final days of the performance test, developers are expected to be available full time to quickly address issues found during the performance test.

Note: *The application owner, development lead, and other stakeholders must approve the performance test plan.*

- Typically, the performance test team will hold a meeting to discuss the plan and may require designated stakeholders to be physically present at the meeting.

  Notes: *The performance test schedule must be completed and approved by stakeholders prior to the start of the performance test.*

  Federal Student Aid project management must give the performance test team the green light to proceed with the test.

  Performance Testing is an exception to the process described as System Testing. The high-level differences are described below.

4.4.6.1 Entrance Criteria

The following conditions must exist before Performance Testing can occur:

- Baseline software requirement specification, functional requirement document, detailed system design documents,
- Master Test Plan and Phase Level Test Plans (if applicable) have been created
- The application to be tested has been tested for system functionality
- A performance testing tool has been installed
- A Test Environment has been created for Performance Testing
- All the performance test scripts complete successfully
- The Performance Test Plan has been created

4.4.6.2 Test Preparation

The application team provides test data for the performance test. The type of data needed may include:

- Test data for boundary value analysis and equivalence partitioning

- Test data which considers volumes required and anticipated usage patterns, as well as data dependency

4.4.6.3 Performance Test Execution

The Performance Test Team is responsible for performing the following Performance Test related tasks:

- Reviewing Performance Test readiness
- Creating Performance Test Suites, scenarios, Test cases, scripts and procedures
- Determining the monitoring metrics to be measured, maximum allowable values, anticipated values, and minimum acceptable values. *Note: Input will be required from the project subject matter experts.*
- Executing the Performance Test Scripts
- Validating Performance Test results
- Tracking and managing defects
- Creating Performance Test Reports consisting of Performance Test metrics

4.4.6.4 Performance Test Reporting

Performance Testing must provide the following outputs:

- Performance Test Summary Report
- Performance Test Defect Report
- List of configuration changes needed on the application and on the environment

4.4.6.5 Exit Criteria

The Exit Criteria for Performance Test Cases are:

- Application is stable under a specified load
- Necessary configuration changes have been applied

4.5 Phase 4: User Acceptance Testing (UAT)

Federal Student Aid Application Owners perform UAT. User Acceptance Testing is a phase near the end of the SDLC in which the customer has an opportunity to execute the application in an environment that parallels the operational environment. The primary purpose of UAT is to provide proof the application is ready to be delivered. In the Federal Student Aid Organization, the Federal Student Aid Application Owners perform UAT.

The test contractor may be responsible for developing the UAT Test Plan, Test Suites, and related UAT activities and documentation. The test contractor or development team may be responsible for providing data to be used for UAT.

If the test contractor is not contractually responsible for these tasks and for assisting Federal Student Aid in performing UAT, the Federal Student Aid Test Manager is responsible for performing these tasks. The testing types that occur during UAT are:

- Usability Testing (Required)
- Intersystem Testing (Required, if applicable)
- Regression Testing (Required)
- Functional Testing (Required)
- Specialized Testing Types if applicable:
- 508 Compliance Testing
- Security Testing

The UAT Team, including the Test Manager and UAT Testers, plan UAT activities. The UAT Test Plan includes the assigned tasks, schedule, and resources. Planning must include review of Integration Testing Defect Reports and System Testing Defect Reports to identify problems and areas of concern that occurred in previous testing phases and that may need additional scrutiny during UAT. When UAT includes Intersystem Testing, the Federal Student Aid Test Manager is responsible for coordinating testing activities with the major players of subsystems that may be affected by the testing activities.

For applications that include functionality performed through batch jobs, the contractor is responsible for running each job and providing Federal Student Aid with the data used to create the output and with the hard copy reports and/or files produced. Batch functionality typically includes processing data received from other systems, creating data for transmissions to other systems, and updating data without any manual modification of data. Federal Student Aid will be responsible for validating the output from those jobs.

Below is an example of a user acceptance testing scenario:

UAT Scenario

The Federal Student Aid Information Center's Interactive Voice Response Unit (IVRU) allows callers to speak the first two letters of their last name to access loan status information.

Test

The Application Owner tester calls into the Federal Student Aid Information Center and receives a prompt to speak the first two letters of the last name being used for the test. The system does not recognize the letters spoken by the tester on the first two attempts. The system then recognizes the letters spoken on the third attempt. The Application Owner would then determine if the IVRU system met the acceptable quality level, business expectations, and/or service level agreements for the project.

Note: *When an application provides functionality such as data entry, data update, and data retrieval on-line or through a reporting tool to Application Owners within Federal Student Aid, Federal Student Aid is responsible for performing hands-on UAT.*

Table 4-4, UAT Testing Roles, Responsibilities, and Artifacts, shows the Project Manager, Test Manager, and UAT Test Team roles and responsibilities as required by Federal Student Aid, and the artifacts that must be created for successful UAT.
Test suites that have been approved by Federal Student Aid may not be changed during test execution. When analysis of defects leads to requirement clarification or modification, then new Test Suites must be created and approved by Federal Student Aid.

Table 17: UAT Testing Roles, Responsibilities, and Artifacts

Role	Responsibility	Artifact(s)
Federal Student Aid Project Manager	Collaborate with the Federal Student Aid Test Manager, Contractor Project Manager, and Contractor Test Manager if a contract is involved, to evaluate the need to perform parallel System Testing and UAT (performing testing in both phases concurrently) for part of each testing phase and to develop an appropriate mitigation strategy to ensure that testing is successful. The Federal Student Aid Project Manager will approve or reject a recommendation to perform testing using this technique, which should only be used on an exceptional basis. If Intersystem Testing is performed, coordinates with representatives of other applications that interface with the system under development to establish schedules for Intersystem Testing.	• Approved parallel testing plan and related mitigation strategy • Approved test artifacts
Contract Project Manager	Provide information for testing and testing related activities.	• Updated Project Management Plan • Updated Work Breakdown Structure

Role	Responsibility	Artifact(s)
Federal Student Aid Test Manager	Collaboarate with the Federal Student Aid Project Manager, Contractor Project Manager, and Contractor Test Manager if a contract is involved, to evaluate the need to perform parallel System Testing and UAT (performing testing in both phases concurrently) for part of each testing phase and to develop an appropriate mitigation strategy to ensure that testing is successful.	• Approved parallel testing plan and related mitigation strategy
	Review Deliverables from test effort. Provides input in to TRR approval process. Coordinate with assistive technology team for section 508 testing.	• Comments Matrix for testing artifacts • Sign-off after reviewing and approving artifacts
Test Manager	Collaborate with the Federal Student Aid Project Manager, Contractor Project Manager, and Federal Student Aid Test Manager to evaluate the need to perform parallel System Testing and UAT (performing testing in both phases concurrently). If such a technique is to be recommended, the Test manager will develop the recommendation, and develop an appropriate mitigation strategy to ensure that testing is successful even if testing problems occur employing this technique.	• Approved testing artifacts • Comments matrix for testing artifacts

Role	Responsibility	Artifact(s)
	Ensure that all Entrance Criteria are satisfied prior to beginning Test Preparation. Minimum Entrance Criteria include having: • Baseline MTP • Baseline Detailed Design Document • Approved Requirements Traceability Matrix • Approved UAT Test Plan • UAT Test Environment • Application software installed • Application login and password assigned • Interface Control Document for Intersystem Testing • Intersystem Testing activities coordinated • UAT Plan • TRR documentation and process • System Testing completed with all urgent and high errors having been resolved • Test cases assigned to testers • All Test Suites, Test Cases, Test Scripts, and Test Procedures created • Completed a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase Generated UAT Test data that: • Is reusable when appropriate • Is generated utilizing an automated tool whenever possible • Is generated from data that has been processed on the UAT and that can be re-used for testing Review test status from testers which may include sign-off sheets Follows up on outcomes of the TRR	• UAT Test Plan created and approved • UAT Test Environment exists • Integrated software components • User ID and password • UAT Support Test Plan (Requirement for Contractor) • Completed TRR • System Test Defect Report • Tests cases have been assigned to Testers • All Test Suites, Test Cases, Test Scripts and Test Procedures have been created • All change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase have been reviewed • UAT Test Data

Role	Responsibility	Artifact(s)
	Oversee testing activities.	• Email • Meeting Minutes • Status Reports
	Review all testing artifacts.	• Reviewed UAT Test Cases • Reviewed UAT Test Results
	Ensure that all Exit Criteria are satisfied prior to submitting UAT Artifacts to the Federal Student Aid Project Manager. Minimum Exit Criteria include ensuring that: • All planned UAT Test Suites pass • All urgent and high priority defects identified during UAT are resolved • All artifacts, reports, and metrics are updated by the Test Team to reflect the current state of the UAT • The Project Manager has reviewed and approved all artifacts • The Federal Student Aid Deliverable Review Process is completed • When a contractor is used, the Federal Student Aid Test Manager reviews the contractor's test results and accepts them, or requires additional testing	• Completed UAT Test Cases • Defect Log • Updated artifacts, reports, and metrics • Test Manager sign-off after reviewing and approving artifacts • Federal Student Aid Test Manager sign-off after reviewing and approving artifacts • Test Coverage reported
Development Team	When testing with other applications is required for Intersystem Testing, the development team must provide the Federal Student Aid test team the Interface Control Documents at least one month prior to the Intersystem Testing effort unless the contract states otherwise.	• Interface Control Documents • Data Management Information

Role	Responsibility	Artifact(s)
UAT Test Team	UAT Testing preparation • Updating the UAT Test Plan (as needed) • Determining and obtaining the UAT Test Environment required to execute UAT Test Suites • Creating and updating UAT Test Suites, including determining test scenarios, test scripts, and test procedures to perform each of the types of tests they will conduct during UAT • Obtaining logins and passwords	• Updated UAT Test Plan • UAT Test Framework • UAT Test Suite (including test scenarios, test scripts, and test procedures)
	UAT execution by: • Executing UAT Test Cases using the UAT Test Framework • Evaluating UAT Test Results • Performing Defect Management to verify, log and retest defects	• Executed UAT Test Cases • Evaluated UAT Test results • Defects entered into the defect management tool • Updated code that resolves defects • Defects tracked and managed to the resolution
	UAT reporting by: • Updating UAT Test Cases • Updating UAT Plans • Creating UAT Reports • Creating UAT Metrics • Creating the UAT Defect Report	• Updated UAT Test Suites • Updated UAT Test Plan • UAT Test Summary Report • UAT Test Metrics • UAT Test Defect Report

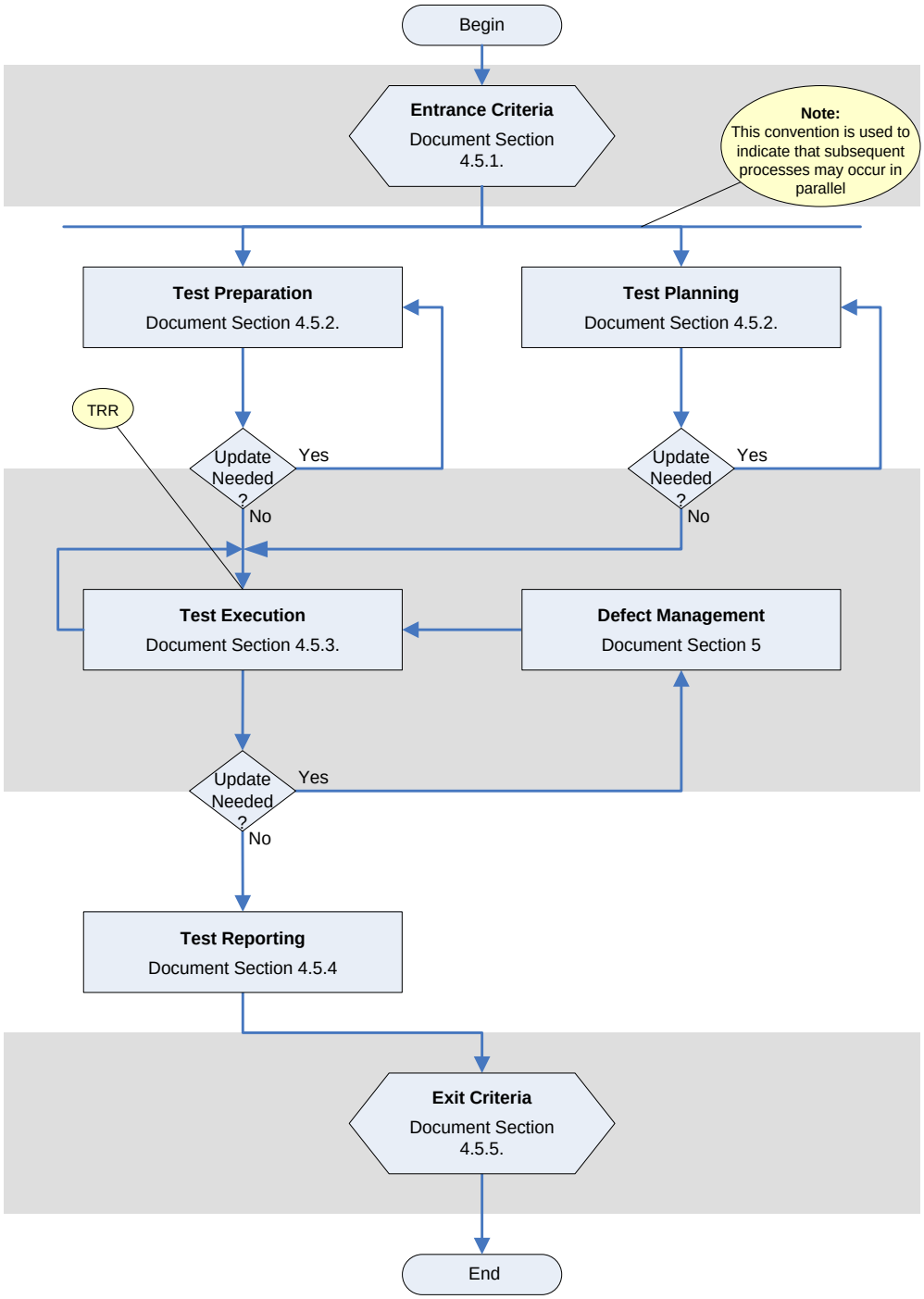


Figure 9: Process Overview of User Acceptance Testing

Note: The UAT roles and process are tailored for Agile projects to fit the Agile team and iterative development lifecycle approach. See Appendix D.21 Agile Testing for more information.

4.5.1 Entrance Criteria

The following conditions are the minimums that must be met before UAT can begin. Federal Student Aid may define additional Entrance Criteria for a project. A contractor may recommend additional criteria for Federal Student Aid approval. The Federal Student Aid Project Manager must approve exceptions.

- Baseline MTP
- Baseline Detailed Design Document
- Approved Requirements Traceability Matrix
- Approved UAT Plan
- UAT environment available for testing
- Application software installed
- Application login and password have been assigned
- UAT Support Plan has been created
- Stage Gate 2 TRR approved
- Overview on defects found in previous phase
- Meeting to discuss defects found and defect resolution
- System Testing is complete and all urgent and high errors found have been addressed and resolved
- Tests have been assigned to Testers, and all Test Suites, Test Cases, Test Scripts and Test Procedures have been created

Note: These conditions must include consideration for each of the test types to be performed during UAT.

4.5.2 Test Preparation

Before UAT can begin, the UAT Team must make the following test preparations

- Updating the UAT Test Plan (as needed)
- Determining and obtaining the UAT Test Environment required to execute UAT Test Suites
- Creating and updating UAT Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform each of the types of tests. The contractor develops UAT Test Suites unless stated otherwise in the contract.
- Obtaining logins and passwords
- Generating UAT Test data that:

- Is reusable when appropriate
- Is generated utilizing an automated tool whenever possible
- Is generated from data that has been processed on the UAT and that can be re-used for testing

- The development team may be responsible for providing data to the FSA test team in accordance with the project schedule.

4.5.3 Test Execution

Once the test preparations are completed, the UAT Test Team performs the actual User Acceptance, as follows:

- Executing UAT Test Suites using the UAT Test Framework
- Evaluating UAT Test Results
- Performing Defect Management to verify, log, and retest defects

Note: *Testing techniques that are useful during UAT include, but are not limited to, Intersystem Testing, black box testing, regression testing, 508 testing, communication testing, and* security *testing. The* ***Glossary in Appendix B*** *provides descriptions of each technique.*

If contractor testers are to perform Intersystem Testing during UAT, files are created for transmission *to business partners, and files are desk checked to ensure conformance to the recipient's formatting requirements. No data exchange will occur between the contractor and any party external to Federal Student Aid without the written authorization of the Federal Student Aid Project Manager*

4.5.4 Test Reporting

The UAT Testing Team prepares the following reports and submits them to the Test Manager or Project Manager for review and approval. Additional reports may be required for a specific project.

- UAT Test Suites
- UAT Plans
- UAT Reports
- UAT Metrics
- UAT Defect Report
- UAT Sign-Off Sheet (template in **Appendix D: Templates)** when required by the Test Manager

4.5.5 Exit Criteria

UAT will not be considered complete until all of the Exit Criteria are completed and approved. Federal Student Aid may define additional Exit Criteria for a project. A contractor may recommend additional criteria for Federal Student Aid approval. The

Federal Student Aid Test Manager must approve exceptions. The following are the minimum Exit Criteria:

- All urgent and high priority defects (and all other defects as defined by the Federal Student Aid Project Manager) identified during UAT are resolved
- Mitigation strategy for resolving outstanding open medium and low defects
- All the artifacts (including test cases and automated test scripts), reports, and metrics are updated by the testing team to reflect the current state of the UAT
- Deferred defects documented and included in summary counts
- The Test Manager has reviewed and approved all artifacts
- The Federal Student Aid Deliverable Review Process is complete
- Test Summary Report delivered to Federal Student Aid
- UAT Sign-Off Sheets complete

4.6 Phase 5: Post Implementation Verification

The Post Implementation Support Testing will occur during the Post Implementation Support Period (PISP). PISP is the time period after implementation of the application into the production environment.

The test team may be required during the PISP. This will ensure that the development or maintenance activities receive the same level of testing as is given prior to acceptance of the application by Federal Student Aid. The Project Change Control Board will prioritize the issues identified during the Post Implementation Support Period. CRs will be created for each issue and approved CRs will be assigned to the development team for implementation. The CRs will be tested by the test team (contractor and/or Federal Student Aid testers) prior to deployment to production. Metrics are captured during this period.

An example of Post Verification testing is to verify that the monthly SAIG report contains accurate data. The install may occur on the 10th of the month, but the monthly report is scheduled to run on the 28th of the month. The Post Implementation Verification testing will occur on the 28th of that month.

The objective of **Post Implementation Verification Testing** is to validate the execution of critical functions of the application when the application is installed into production. If the test is found to be unacceptable, the application owner has the option of requiring the software be rolled back. At that point, a check must be performed in production to ensure that the correct version of the software was installed. Should a change be required in order to correct the issue, a formal Change Request must be submitted, reviewed and approved prior to correcting the issue and redeploying the software to production. At no time should a Defect record be created in order to document a post-production issue. Once the CR has been created, verification of the issue should be performed in the test environment to help in the analysis of the issue that was found during Post Implementation Verification.

An example of Post Implementation Verification testing is to verify that records for Pell Grant awards appear on the web. Schools will be prevented from sending data until the Post Implementation Verification test is proved successful. One school will be allowed to send data to Federal Student Aid that will in turn trigger the application to update the Pell Grant awards data. If the web does not display the expected results, the application owner may require the software to be rolled back to its previous version.

Table 4-5, Post Implementation Verification Testing Roles, Responsibilities, and Artifacts, summarizes Federal Student Aid's minimum requirements for the roles, responsibilities, and artifacts related to Post Implementation Verification Testing.

Test Suites that have been approved by Federal Student Aid may not be changed during test execution. When analysis of issues leads to requirement clarification or modification, then new Test Suites must be created and approved by Federal Student Aid.

Table 18: Post Implementation Testing Roles, Responsibilities, and Artifacts

Role	Responsibility	Artifact(s)
Federal Student Aid Project Manager	Provide information for testing and testing related activities.	• Updated Project Management Plan • Updated Work Breakdown Structure
	Transition testing documentation provided to Operations and Maintenance contractor or new test team (test plan, test cases, lessons learned).	None

Role	Responsibility	Artifact(s)
Test Manager	Ensure that all Entrance Criteria are satisfied prior to beginning Test Preparation. Minimum Entrance Criteria include having • The application deployed to a production environment • Post Implementation Verification (test) Plan created • Baseline MTP • Approved Requirements Traceability Matrix • Application login and password assigned • Test cases assigned to testers • Test Suites, Test Cases, Test Scripts, and Test Procedures created Completed a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase	• Final MTP • Approved Requirements Traceability Matrix • Post Implementation Verification Testing Test Plan created and approved • Post Implementation Verification Testing Test Environment exists • Integrated software components
	Oversee testing activities.	• Emails • Meeting Minutes • Status Reports
	Review all testing artifacts.	• Reviewed Post Implementation Verification Testing Test Cases • Reviewed Post Implementation Verification Testing Test Results

Role	Responsibility	Artifact(s)
	Ensure that all Exit Criteria are satisfied prior to submitting Post Implementation Verification Testing Artifacts to the Federal Student Aid Project Manager. Minimum Exit Criteria include having: • The functions tested during Post Implementation Verification Testing work correctly • All Post Implementation Verification Testing cases passed • All urgent and emergency priority Change Requests created as a result of issues found in the Post Implementation Verification Testing phase reported and resolved • The Federal Student Aid Test Manager working in consultation with the contractors, Federal Student Aid Application Owners, and project teams to determine whether the Post Implementation Verification Testing test results are satisfactory. • All artifacts, reports, and metrics updated by the Test Team to reflect the current state of the tested system • The Federal Student Aid Deliverable Review Process completed • When a contractor is used, the Federal Student Aid Test Manager reviews the contractor's test results and accepts them, or requires additional testing	• Completed Post Implementation Verification Testing Test Cases • Issue Log • Updated artifacts, reports, and metrics • Test Manager sign-off after reviewing and approving artifacts • Federal Student Aid Test Manager sign-off after reviewing and approving artifacts

Role	Responsibility	Artifact(s)
Post Implementation Verification Testing / Post Implementation Test Team	Post Implementation Verification Testing preparation Create Post Implementation Verification Testing Test Suites and determining the Post Implementation Verification Testing Functional Test scenarios consisting of the Post Implementation Verification Testing Functional Test Suites, test Scripts, and test Procedures. Create Post Implementation Verification Testing test reports that include test metrics and test results Using production data to perform Post Implementation Verification Testing Functional Tests. Update the Post Implementation Verification Plan (as needed) Create and updating Post Implementation Verification Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform each of the types of tests they will conduct during Post Implementation Verification	• Updated Post Implementation Verification Testing Test Plan (if needed) • Post Implementation Verification Testing Test Framework • Post Implementation Verification Testing Test Suite (including test scenarios, test cases, test scripts, and test procedures • Post Implementation Verification Testing Test Data
	Post Implementation Verification Testing execution by: • Executing Post Implementation Verification Test Suites using the Production system • Evaluating Post Implementation Verification Test Results • Performing Issue Management to verify, log, and retest issues.	• Executed Post Implementation Verification Testing Test Suites • Evaluated Post Implementation Verification Testing Test results •

Role	Responsibility	Artifact(s)
	Post Implementation Verification Testing / Post Implementation Testing reporting by • Creating Post Implementation Verification Testing Summary Reports • Creating Post Implementation Verification Testing metrics and documentation of issues found in production • Submitting all artifacts to the for review and approval •	• Updated Post Implementation Verification Testing Test Suites • Updated Post Implementation Verification Testing Test Plan • Post Implementation Verification Testing Test Summary Report • Post Implementation Verification Testing Test Metrics

Note: *The contractor is responsible for Post Implementation Verification Testing unless the contract indicates otherwise. The responsible party will ensure that the application's functionality, performance, and reliability meet Federal Student Aid's requirements.*

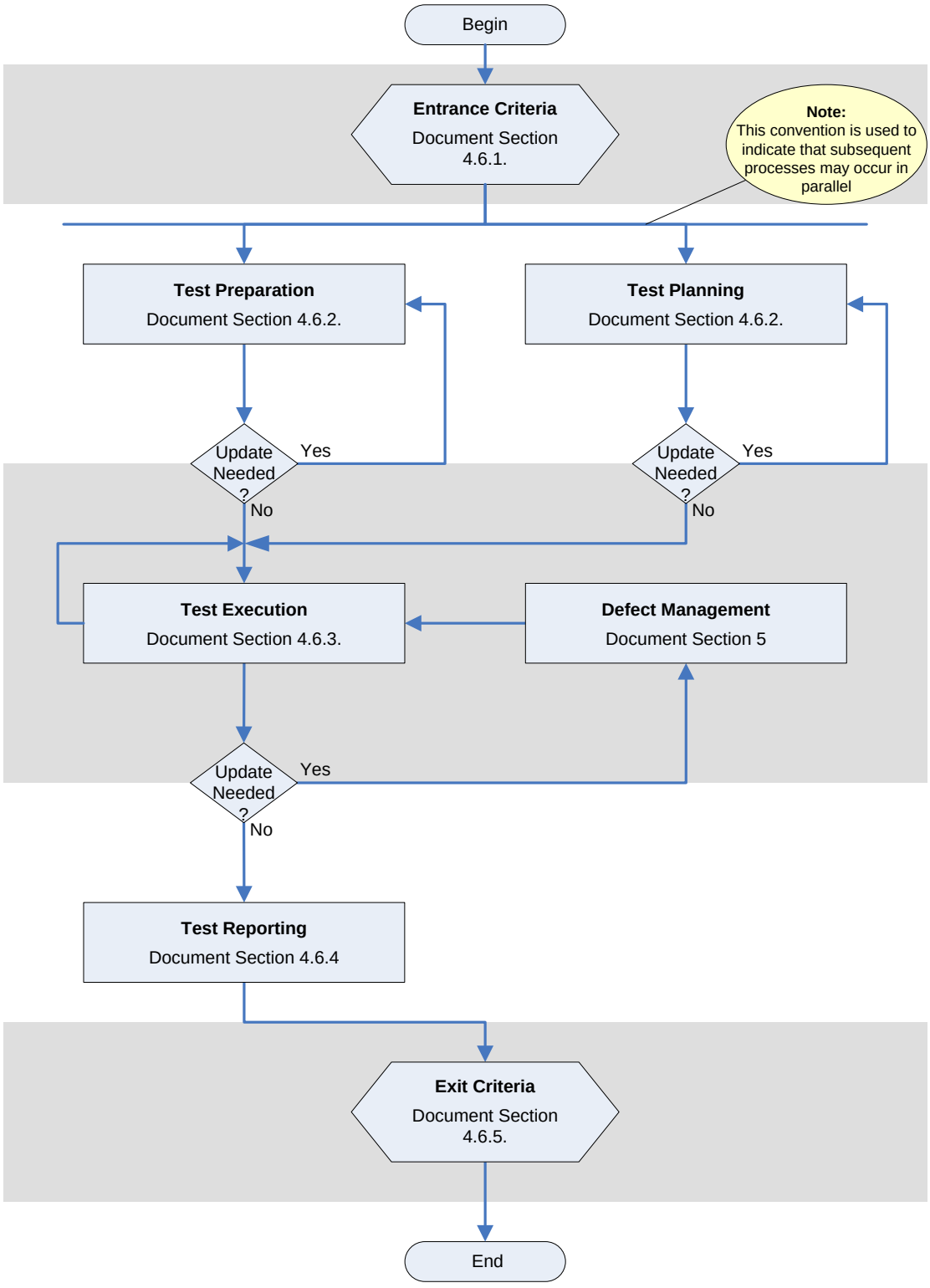


Figure 10: Process Overview of Post Implementation Verification

4.6.1 Entrance Criteria

The following conditions must be met before Post Implementation Verification Testing can occur:

- The application must be deployed to a production environment
- A Post Implementation (test) Plan must be created
- Baseline MTP must be created
- Approved Requirements Traceability Matrix must be created
- Application login and password must be assigned
- Test Suites, Test Scripts, and Test Procedures must be created
- Tests must be assigned to testers

4.6.2 Post Implementation Verification Testing Preparation

The Post Implementation Verification team is responsible for performing the following tasks before and during Post Implementation Verification Testing:

- Creating Post Implementation Verification Test Suites and determining the Post Implementation Verification Testing Functional Test scenarios
- Creating Post Implementation Verification Test reports that include test metrics and test results
- Updating the Post Implementation Verification Plan (as needed)
- Obtaining logins and passwords to interface(s)

4.6.3 Test Execution

The Test Team performs the following execution activities:

- Executing Post Implementation Verification Test Suites in the Production environment
- Evaluating Post Implementation Verification Test Results
- Identifies issues, submit CRs to address the issues, and eventually retest.

4.6.4 Test Reporting

Post Implementation Verification must provide the following outputs:

- Post Implementation Verification Test Summary Reports
- Post Implementation Verification Test Metrics
- Post Implementation Verification Test Issue Reports, if applicable
- Submitting all artifacts to the Project Manager for review and approval

4.6.5 Exit Criteria

Exit Criteria define the standards for work product quality that enables it to go to the next level or phase of testing. The following Exit Criteria are required for Post Implementation Verification cases:

- The critical functions tested during Post Implementation Verification Testing work correctly
- All Post Implementation Verification Testing cases passed
- All emergency and urgent CRs or others as defined by the Federal Student Aid Project Manager encountered in the Post Implementation Verification Testing phase are reported and resolved
- The Federal Student Aid Test Manager, working in consultation with the contractors, the Federal Student Aid Application Owner, and project teams to determine whether the Post Implementation Verification Testing test results are satisfactory
- All artifacts, reports, and metrics updated by the Test Team reflect the current state of the tested system
- The Project Manager completing the review and approving all artifacts
- The Federal Student Aid Deliverable Review Process completed
- When a contractor is used, the Federal Student Aid Test Manager reviews the contractor's test results and accepts them, or requires additional testing

4.7 Post Implementation Support Testing

The Post Implementation Support Testing will occur during the Post Implementation Support Period (PISP). PISP is the time period after implementation of the application into the production environment.

The application development contract must specify the term of the PISP and the testing contract must require that testing resources are available during the support period. This will ensure that the development or maintenance activities receive the same level of testing as is given prior to acceptance of the application by Federal Student Aid. The combined contractor and Federal Student Aid Change Control Board (CCB) will prioritize the issues identified during the Post Implementation Support Period.

The identified issues will be categorized and included in planning and scheduling for resolution. An issue may be identified as critical and is needed to keep the implemented system running. Other issues may also be identified as an enhancement or one that was missed in the development process that will need to be fixed and issued in another release.

The critical items will then be modified by the team responsible for problem resolution and testing resources are made available.

CRs will be created for each issue and approved CRs will be assigned to the development team for implementation. The CRs will be tested by the contractor and Federal Student Aid testers prior to deployment to production.

Metrics are captured during this test period.

4.8 Testing Emergency Software Releases

Urgent and high errors found in production require the immediate attention of the development team and the Federal Student Aid Application Owner. Testing support must be available when an urgent and high error is found in the application. The Test Team will need to review the requirements and design test suites to address the corrective application modification. Testing must include regression test scenarios to ensure that key functionality has not been compromised. Testing must be accurate to confirm that the modification meets the business need.

5 Defect Management

5.1 Overview

Defect Management is the process of recognizing, investigating, taking action on, and disposing of defects. Defect Management follows the testing phase activities. Defect management is a part of all test phases. Although prevention is mainly the responsibility of the development team, the testing team is also responsible to ensure that test scenarios, scripts and the test environment are correct to avoid wasting time on invalid defects.

Defects must be identified and resolved through a specific Defect Management Process, which includes:

- **Defect Prevention**: A risk based process involving the identification of urgent and high defect related risks, estimation of the impact of each of the risks, and using risk mitigation to minimize the impact of each risk.
- **Defect Discovery**: Includes identifying defects, documenting them, and receiving acknowledgement from the development team that the Testing Team has identified a valid defect.
- **Defect Tracking**: The Testing Team enters each defect into a Defect Management Tool as soon as the defect is discovered, and tracks each defect to closure. Defect Management tools used in FSA are descrived in Section 4.5.
- **Defect Correction or Resolution:** Following discovery, developers and users prioritize, schedule, resolve a defect, document the resolution, and provide notification of the resolution so the tester can validate that the issue has been resolved based on assigned severity and priority. .
- **Process Improvement**: Analysis of the process in which a defect, found in testing, originated to identify ways to improve the process to prevent future occurrences of similar defects.

During the project and at the end of the project, the Test Manager reports defects and related metrics to the contract Project Manager, the Federal Student Aid Project Manager, and the Federal Student Aid Enterprise Test Manager. The defect management reporting process must include metrics that provide maximum benefit to the Federal Student Aid's process improvement efforts. The overall goal is to use lessons learned either to minimize defects in the future, or to identify defects early in the development process, and to maintain consistency in order to reduce adverse impacts and costs of correction.

Notes: *A deliverable is baselined each time a predefined milestone is reached. Errors identified before a deliverable is baselined will not be considered a defect.*

At the end of each test phase all defects must either be closed, deferred or in a state which is acceptable to the Federal Student Aid Test Manager/Test Lead.

5.2 Relevance of Defect Management to this Document's Audience

The following key staff members need to fully understand and follow the requirements in the Defects Management section of this Document. Subsections following this list describe the responsibilities for each of the following positions:

- Federal Student Aid Project Manager
- Federal Student Aid Test Manager/Test Lead
- Federal Student Aid Enterprise Test Manager
- Federal Student Aid Application Owner
- Federal Student Aid Application Development Team
- Federal Student Aid Application Test Team

Table 19: Defect Management Roles & Responsibilities

Title	Responsibilities
Project Manager	• Analyzing defect metrics and reports and providing input to the defect prevention and resolution process. • Receiving information throughout the defect lifecycle including defect evaluation information, defect status information, and defect resolution information. • Being involved in assigning defect priorities.
Test Managers	• Analyzing management reports and metrics, and providing input to the defect prevention, defect discovery, and process improvement processes.
Federal Student Aid Enterprise Test Management	• Analyzing defect data for trends. • Defect resolution and process improvement processes through various management reports. • Validating compliance of the testing contractor with Federal Student Aid Defect Management Requirements.
Federal Student Aid Application Owners	• Reporting defects discovered during UAT. • Tracking defects to resolution. • Providing input into the defect prevention, defect discovery and process improvement processes using information obtained from Defect Management reports.

Title	Responsibilities
Application Development Team	• Resolving defects reported during each phase of testing. • Defect analysis. • Providing input into the defect prevention, defect discovery and process improvement processes using information obtained from Defect Management reports. • Recording or updating defects in a defect tracking tool.
Application Test Team	• Reporting defects discovered during testing. • Analyzing defects by providing the testing artifacts to support the defects such as screen shots, providing a description, assigning the defect severity and defect type for each. • Analyzing the resolved defects to prepare for regression testing. • Tracking defects to resolution.

5.3 Defect Classification

A valid defect is one that is not a duplicate of a previously reported defect and one that can be reproduced. The following defect classifications will be used during the test effort:

- Type
- Severity
- Priority
- Status

5.3.1 Defect Types

A defect type indicates the source and/or location of the defect. The test team may assign multiple defect types to a single defect. Defect types include the following characteristics:

- Gap in requirements definition
- Design problem
- Data element problem (missing, incorrect edit, etc.)
- Calculation problem (missing, incorrect result, etc.)
- User Interface (incomplete, incorrect presentation, etc.)
- Intersystem Interface (file layout, etc.)
- Report (missing data, incorrect format, etc.)

- New Requirement (an enhancement or may have missed something in the development process that will need to be fixed and issued in the next release)
- Tester Error/Invalid Defect
- Timing Error – a defect due to incorrect or missing consideration to timeouts
- Documentation – missing or incorrect steps in test cases, ambiguous or inadequate test cases
- Navigation – links not working or links working incorrectly
- System Defect
- Performance Defect
- Error Handling
- Other: The tester will be allowed to specify the defect type if none of the predefined types are appropriate.

Note: *The required Federal Student Aid defect types must be entered in the predefined fields in the applicable defect management tool prior to any testing activity.*

5.3.2 Defect Severity

Defect severity indicates the level of business impact of the defect. The tester defines defect severity using one of four levels of severity based on the IEEE scheme.

Every application/project should cite specific examples for each severity level that are pertinent to the functions to be implemented and tested.

At the end of each testing phase, there cannot be any open defects with an **Urgent** or **High** severity level.

Table 20: Defect Severity Levels

Severity	Description	Example
Urgent	Prevents the accomplishment of an operational or mission essential capability.	• Jeopardizes data quality (e.g., corrupt data entered into the system, data is corrupted by the system, or data is calculated incorrectly) • Software crashes (deprive user of system or cause loss of data) • Loss of data (e.g., incorrect file formats sent to other systems or decoded incorrectly) • Jeopardizes safety (normally not software related, but may be, e.g. if personal safety is involved) • Inability to navigate to a required page due to an error in establishing a specified URL • Jeopardizes security (e.g., potential violation of Privacy Act)
High	Adversely affects the accomplishment of an operational or mission essential capability and no work around solution is known.	• Noncompliance with a required standard (e.g., 508 Compliance if required) • Software issues which are not critical but impact the use of the system (e.g., printing errors, inability to print a form) • Data input problems (which do not corrupt data), etc. • Missing help files, no other system documentation • Missing windows • Missing functionality that is crucial • A search function that provides incorrect results • Drop down selections do not work • Permission levels are compromised with functions that are not applicable
Medium	Adversely affects the accomplishment of an operational or mission essential capability, but a work around solution is known and productivity is negatively impacted.	• A required help file is missing, but there is a hardcopy document containing the same data available to the user. • A form or window does not print, but there is another way to print it (e.g., the print button does not work, but there is a pop-up window which has a print function and it works)

Severity	Description	Example
Low	Results in user inconvenience or annoyance but does not affect a required operational or mission essential capability.	• Typos in help files • Minor color issues such as window or button colors (which do not impact 508 compliance if required) • Typos on screen for internal Federal Student Aid use only (user facing errors should be rated high or medium) • Typos in documentation for internal Federal Student Aid use only (user facing errors should be rated high or medium) • Operator annoyances (e.g., tabbing between fields is not in the most desirable order or sub-window groupings are not logical)

Note: *The required Federal Student Aid defect severity levels must be entered in the predefined fields in the applicable defect management tool prior to any testing activity.*

5.3.3 Defect Priority

Defect priority indicates the order in which the development team will address defects. This rating provides an estimate of the relative standing of the issue in relation to all other identified defects. The Defect Priority categories are:

- **High**: All Urgent and most High severity defects
- **Medium**: High severity defects not assigned to a high priority
- **Low**: All Medium and Low severity defects

Note: *The Federal Student Aid Test Manager and Federal Student Aid Project Manager have the authority to override the priority assigned by the contractor. The Federal Student Aid Test Manager (with help with the application owner/user) assigns defect priority during the UAT phase.*

Note: *The required Federal Student Aid defect priority categories must be entered in the predefined fields in the applicable defect management tool prior to any testing activity.*

5.3.4 Defect Status

Defect status indicates the current disposition of a defect and is updated throughout the test cycle. The current supported FSA defect statuses are described below:

Table 21: Defect Status Levels

Defect Status	Description
New	A defect is submitted by the tester or developer
Start Working	A defect is assigned to a developer
Resolved	A defect has been corrected by the developer
Verified	A resolved defect has been tested by the tester with successful results
Reopen	A resolved defect has been tested by the tester with unsuccessful results or a previously verified defect reappeared in a course of testing

5.4 Defect Lifecycle

Every defect has a lifecycle, beginning with submission of the defect into the defect management system and ending with defect closure or cancellation.

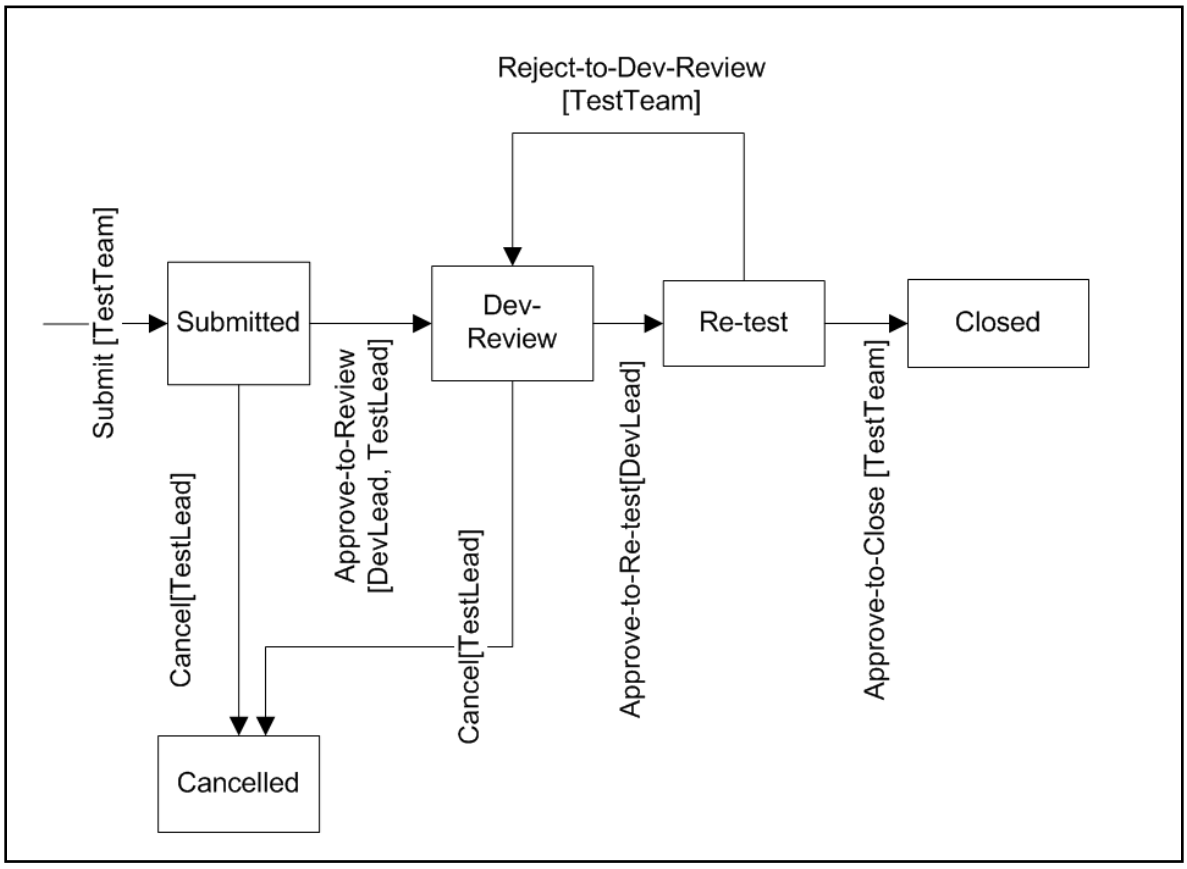
In the use of the legacy Defect Management tools, the defect lifecycle is closely tied to the lifecycle of a CR. Additional details on how a defect relates to a CR can be found in the Federal Student Aid Enterprise Test Management User Guide.

The table below shows the series of steps taken by each FSA team member when a defect is identified and corrected when using the legacy Defect Management tool.

Table 22: Defect Step Action Table

Person	Action
Tester	Creates a defect record in the applicable defect tool in response to a failed test case.
Test Lead	Also may create a defect record in RTC in response to a failed test case
Dev Lead	Reviews the Defect; contacts the Test Lead with reasons why the issue identified is NOT a defect so that it can be resolved; if a defect, sets the status to ‘in progress’ and starts working on it
Test Lead	Based on Dev Lead review, either agrees to cancel the defect or wait for defect resolution
Developer	Corrects the Defect, documents changes, and sets the defect status to ‘resolved’
CM Team	TBD
Tester	Retests and closes the defect after verifying that it has been corrected and sets the status to ‘verified’ or sets the status to ‘re-opened’ when the defect is not remediated successfully

The images below show the lifecycle of a defect and which type of users are responsible for transitioning the defect from one state to another. The first one is for the defect statuses used in the legacy Defect Management Tools. The second one is for the defect statuses used for Agile projects.



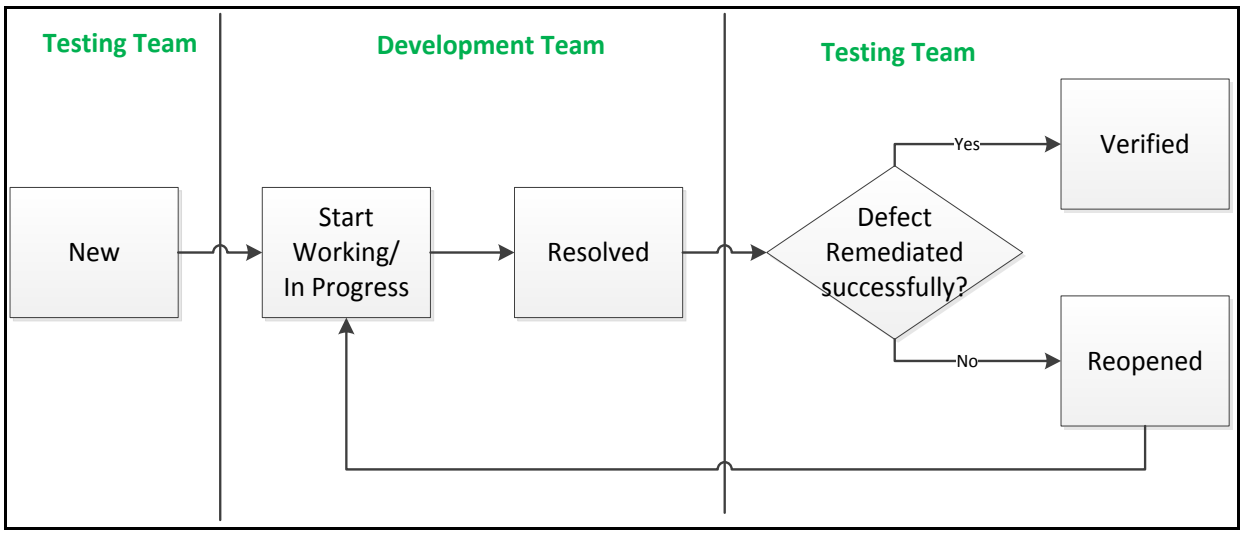


Figure 11: Defect State Models (Workflow)

5.4.1 Opening Defect

Once a defect is discovered, the testing team reviews the defect to ensure that it is valid. Once a valid defect has been determined, the tester that discovered the defect reports the defect in a defect management tool that captures data for all of the fields.

5.4.2 Evaluating Defect

Once a tester has entered a defect in the defect management system, the defect is evaluated for:

- Validity
- Priority
- System Impact

Validity includes determining that the reported error:

- Is a non-conformance of the application under test to its requirements
- Is not a duplicate of a previously reported error

If a defect is determined to be invalid (i.e., if the defect cannot be reproduced), the tester changes the defect to 'resolved' with additional comments.

The application owner advises the tester and developer on assigning priority. The developer will then change the status to 'start working/in progress'.

5.4.3 Analyzing Defect

In legacy Defect Management, each valid defect will be associated with an existing or new Change Request. The CR will be analyzed, categorized, and prioritized. During the analysis of the CR, an impact analysis will be performed by the requirements team, the security team, and the development team; who will also determine the complexity of correcting the defect.

The severity of the defect is assigned based on the characteristics of the defect.

5.4.4 Correcting Defect

In legacy Defect Management, the defect will be corrected as part of implementing a CR or Child CR. When the CR/Child CR reaches the system testing state, the defect will be transitioned to the "Re-Test" as a signal to the tester that it is time to retest the defect.

The defect status is set to 'resolved' by the developer and the testing team may initiate retesting the remediated function(s).

5.4.5 Retesting to Validate Defect and Closing Defect

If a tester logged a defect, the defect has to be retested. After the defect is retested, the state of the defect will be updated in the applicable defect management tool.

- In the legacy Defect Management tool, if the error is resolved, the tester changes the defect state to 'closed'.
- If the error still exists, the tester indicates the defect was not fixed and the defect is rejected back to the "dev-review" state indicating that the defect has not been

corrected. At this point, the related CR is rejected back to the “development” state and the developer is re-engaged in order to correct the problem.

In RTC, if the error is resolved, the defect status is changed to ‘verified’. If the error still exists, the tester describes that the defect was not fixed and the defect is rejected back and the status is set to ‘re-opened’.

The resolution of a defect can be cancelled at the discretion of the Test Manager and the Federal Student Aid Project Manager. The defect may also remain in a deferred state until reopened by the Federal Student Aid Project Manager.

5.5 Defect Management Tools

The defect management tools used in FSA are implemented with user authentication to ensure that the changes are made only by authorized users. Members of the respective FSA projects work with the FSA Defect Management System Security Officer to ensure that only authorized users have access to the system with their respective roles. These tools have effective audit trail mechanism in order to trace the origin and details of all activities and also have advanced search capabilities.

A defect is input onto the project repository by the project testing team. The tool allows the input of the defect description, the attachment of pertinent screenshots to describe the defect and other documentation, allows the input of comments and other details such as the release the defect was found in, the priority assigned by the user or developer, and the severity assigned. The development team also updates other pertinent fields related to remediation. The defects may also be listed with the queried fields and downloaded onto Excel for further reporting metrics.

In addition to the tracking features, it may be integrated with Version Management System, Requirement Management System and Test Management System in order to extend traceability across the full software testing lifecycle. Refer to the Federal Student Aid Test Enterprise Test Management User Guide for additional details on the Defect Management capabilities and how it is integrated.

6 Test Metrics and Reporting

6.1 Overview

Test metrics indicate the efficiency and the effectiveness of testing activities, test plans, and test processes throughout all phases of testing. Metrics also identify trends, productivity, and utilization of testing resources. Effective and standardized reporting of metrics facilitates communications with stakeholders and helps with decision-making processes.

Templates have been provided in Appendix D in the forms of instructions, documents and forms. These templates should be used if the defect management tool does not provide a report to reflect the data required by Federal Student Aid.

6.2 Relevance of Test Metrics and Reporting to this Document's Audience

The following key staff members need to fully understand and follow the requirements in the Test Metrics and Reporting section of this document. Subsections following this list describe the responsibilities of each position:

- Federal Student Aid Project Manager
- Federal Student Aid Test Manager/Test Lead
- Federal Student Aid Enterprise Test Manager
- Federal Student Aid Application Owner
- Federal Student Aid Application Development Team
- Federal Student Aid Application Test Team

Table 23: Test Metrics Roles & Responsibilities

Title	Responsibilities
Project Manager	• Evaluating existing metrics • Providing input into the metric reporting process • Determining how to capture and create metric information related to project status and process improvement with the FSA Test Manager
Test Manager	• Evaluating existing metrics and determining the appropriate metrics for a testing project, during the metric identification process • Providing input into the metric reporting process • Determining how to capture and create metric information related to project status and process improvement with the FSA Project Manager

Title	Responsibilities
Federal Student Aid Enterprise Test Management	• Validating compliance of the testing team with Federal Student Aid Enterprise Test Management Standards • Collecting, analyzing, and creating enterprise level metrics • Providing inputs to the defect prevention, defect discovery, defect resolution and process improvement for various management reports
Federal Student Aid Application Owner	• Providing input to the metric reporting process
Application Development Team	• Creating appropriate metrics for Unit and Unit Integration Testing
Application Test Team	• Developing the metric reporting process using the Defects Report • Creating appropriate test execution and defect related metrics for testing

6.3 Test Metrics Definition

Test metrics are quantitative measures of the degree to which a system, component, or process possesses a given attribute.

There are two basic types of metrics, each with a specific type of data source:

- **Base metrics**: These are metrics for which data can be captured directly. Examples are number of test suites or number of test cases, number of test cases executed, number of test cases failed or number of defects identified, number of test cases blocked or number of test cases deferred.
- **Derived metrics**: These are values derived from base metrics. For example, Test Efficiency is the ratio of number of test suites executed and the number of defects identified.

6.4 Metrics Identification and Selection

The Test Manager and Project Managers are responsible for metric identification and selection. Metrics identification and selection occur during test planning, with approved metrics included in the Phase Level Test Plans. Most metrics will be standardized based on industry best practices and lessons learned. Each metric will include what will be measured as well as the standard for acceptance and failure or rejection.

Federal Student Aid requires that all stakeholders meet before the testing process begins to identify metric recommendations that testers will use during the testing process. Discussion will also take place on how the data will be collected for metrics and how the metrics will be reported. The recommendations will first be approved by Federal Student Aid Test Manager and Project Manager, and then included in the Phase Level Test Plans. If there are no Phase Level Test Plans, metrics that will be used for the test will be defined in the master test plan.

- **Required Metrics**: Federal Student Aid requires that testing use the following sets of metrics (at a minimum):
  - Test Schedule and Effort (See **Section 5.6.1**)
  - Test Defects (see **Section 5.6.2**)
  - Test Defects by Type (see **Section 5.6.2**)
  - Defects Report and Defect Severity and Current State (see **Section 5.6.3)**
  - Test Execution (see **Section 5.6.4)**
- **Optional Metrics: Federal Student Aid** may require the following testing metrics:
  - Defect Execution Reviews (see **Section 5.6.4**)
  - Code Coverage (see **Section 5.6.5**)

Note: *The Federal Student Aid Test Manager and Project Manager must approve exclusion of any required metrics. Applications that are system critical or information sensitive must adhere to the required metrics.*

- Stakeholders may review the following types of information to assist them with making recommendations.
  - **Existing Metrics**: Metrics collected during other projects or those that may be recommended by a manufacturer of a COTS application.
  - **Industry Standards**: Additional metrics and defect reporting may be required based on the project type or the Federal Student Aid project manager's needs. The contractor may provide additional metrics based on the project type or best practices.
  - **Lessons Learned**: Information on what did and did not work on previous test projects. This information is collected by the Quality Assurance Team and entered into the Lessons Learned Database.
- **New Metrics:** When identifying the need for a new metric stakeholders must:
  - Identify data required
  - Determine a data collection method
  - Determine a means of evaluating the data
  - Evaluate the cost/benefit of the metric

6.4.1 Choose the Appropriate Metrics

The project's metrics should be selected based on project goals, stakeholder needs, and the cost/benefit provided by each metric. In addition, ensure that the data needed to conduct the measurement can be acquired within reason, and that metrics have actual significant value to the test and development activities. The number of metrics should be kept to a minimum to ensure that the most critical information is collected, evaluated, and reported.

Accurate collection and appropriate analysis of metrics is important in ensuring that defects do not escape to production and to facilitate process improvement throughout the development life cycle. For example, a detailed analysis should be performed to

determine why each issue occurred in production and not during the construction and validation phase. In addition, testing should be structured to “break the system” and not demonstrate functionality.

6.4.2 Metric Descriptions

The Test Manager may be required to provide the following information for each metric selected: name, benefit, stakeholders, description, measure procedure, measurement frequency, and reporting.

6.5 Metrics Calculation and Reporting

Metrics calculation and reporting processes include:

- Capturing and verifying data
- Analyzing and processing data
- Reporting metrics

6.5.1 Capturing and Verifying Data

Whenever possible, testers should simplify the capture and verification of test metric data by:

- Making reasonable efforts to have a testing tool perform data capture during the testing effort
- Storing all relevant testing data in a location that is easily accessible to all team members

6.5.2 Analyzing and Processing Data

The Test Manager is responsible for creating the metrics and analyzing as follows:

- Reviewing data for correctness/reasonableness
- Reviewing the metrics and determining if the correct information conveys the progress of the test effort
- Automating the calculation of derived metrics, whenever possible and feasible

6.5.3 Reporting Metrics

Federal Student Aid requires the Test Manager to report all metrics in a consistent manner and understandable format.

The approved formats for reporting the metrics in **Section 5.6 Required Metrics** include:

- Supplementing the required metrics with a metric dashboard
- Supplementing the required metrics with graphs and trend lines
- Indicating acceptable results and out of range positive and negative variances using color coding

Note: *Metrics must be made available to the Federal Student Aid Enterprise Testing Team.*

6.6 Required Metrics

6.6.1 Test Schedule and Effort

Test schedule and effort metrics show the scope of the planned test, the results of testing and the derived metrics that allow comparison of the testing to other testing efforts.

Required test schedule and effort metrics include the following information:

- Planned and actual test start and end dates and variances
- Number of test cases executed, passed, failed, blocked
- Total number of test cases executed and planned
- Schedule and effort variance for each of the above elements

All of the metrics listed above are required for all projects and are to be summarized by scenario and phase when reported with “AS OF” dates included on all metrics reports. Additional metrics may be required by Federal Student Aid project management.

Blocked test cases are cases that could not be executed because of a failure that occurred in a previous test case. Blocked defects are test cases that cannot be executed on a defect because another active defect makes the test of the original defect impossible to execute. In these cases, testing of the original defect must be effectively and explicitly deferred until the other blocking defect has been corrected.

The Test Manager will report test schedule and effort metrics using the template in Appendix D.

6.6.2 Test Defect Metrics

Test defect metrics provide information showing the extent of defects numerically and by various categories.

Required defect metrics include the following information:

- Total number of defects
- Number of defects entered by severity (urgent, high, medium, low)
- Number of defects entered by status
- Number of defects entered by defect type (described in Section 4.3.1)
- Number of defects fixed
- Number of defects cancelled
- Number of defects closed and to be verified by the testing team
- Number of defects deferred

The Test Manager will report test defect metrics using the Test Defect Metrics and the Defect Report matrices that follow, or a similar one that contains this data, and additional data if needed.

A Defect Trend Analysis should be done with the data to determine if the defects are increasing in subsequent phases of testing. This analysis examines the number of defects found as the testing life cycle progresses. When analyzing the number of defects found during Integration Testing versus the number found during System Testing, for example, the Test Manager could uncover a risk in the testing process and conduct corrective actions to mitigate the impact.

Templates for Test Defect Metrics and the Defect Report can be found in Appendix D.

Note: *If testing activities are performed by the contractor, the contractor's proposal must state how the test estimates are calculated.*

6.6.3 Defect Severity and Current

Defect severity and current state metrics provide a snapshot of the number and types of defects at a point in time.

The Test Manager will report defect severity and current state using the template in Appendix D.

6.6.4 Test Execution

Test Execution metrics show the results of testing and derived metrics that express the level of effort and progress accomplished. These metrics may also be used to compare data with the testing effort of other projects.

Required test execution metrics include:

- Total Number of Test Cases or Scripts
- Total Number of Test Cases or Scripts Executed
- Total Number of Test Cases or Scripts Attempted
- Total Number of Test Cases or Scripts Passed
- Total Number of Test Cases or Scripts Failed
- Total Number of Test Cases or Scripts Blocked

The Test Manager will report test execution metrics using the template in Appendix D.

6.6.5 Code Coverage Metrics

The Code Coverage metrics (which may be required for Unit Testing as determined by the Federal Student Aid architect team) shows the results of statement coverage, condition coverage, and path coverage. Required Code Coverage metrics include the following information:

- Total Number of Lines in the Code
- Total Number of Lines Unit Tested
- Total Number of Conditions in the Code
- Total Number of Conditions Unit Tested
- Total Number of Test Paths in the Code
- Total Number of Paths Unit Tested

The Development Team will report code coverage metrics using the template in Appendix D.

6.6.6 Turnover Metrics

The test team turnover metric may be required to be provided to the Federal Student Aid Project Team. This is a non-technical, management metric that shows the attrition rate in the testing team over a specified period.

The turnover rate is useful in evaluating the stability of the test team and the overall effectiveness of the testing activities. A lower rate of turnover may indicate a more effective testing process, while a higher rate of turnover can indicate a less efficient effort. While the turnover rate is not definitive in predicting testing effectiveness and efficiency, the information may help identify additional underlying issues (i.e., contractor organizational concerns or contractor Project Management problems). The Test Manager will submit staff turnover information to the Federal Student Aid Test Manager at a frequency required by the Federal Student Aid Test Manager. This information must be provided at the end of the project.

A graphical representation of test summary information must be presented to Federal Student Aid senior management. This representation must include a summarization of the test metrics for the entire project. This information may be provided in an exportable file.

Appendix A: Acronyms and Abbreviations

Acronym	Description
ACS	Administrative Communication System
API	Application Programming Interface
ASD	Adaptive Software Development
ASG	Architecture Support Group
ATP	Assistive Technology Partnership
C&A	Certification & Accreditation
CC	Carbon Copy
CCB	Change Control Board
CCRB	Change Control Review Board
CFR	Code of Federal Regulations
CICS	Customer Information Control System
CIO	Chief Information Officer
CMMI	Capability Maturity Model Integration
COD	Common Origination and Disbursement
COTS	Commercial-off-the Shelf
CTE	Certified Test Engineer
CVE	Common Vulnerabilities and Exposures
DCOM	Distributed Component Object Model
DOB	Date of Birth

Acronym	Description
DOS	Denial of Service
DSDM	Dynamic Systems Development Method
E&IT	Electronic and Information Technology
ESB	Enterprise Service Bus
ED	Department of Education
EOCM	Enterprise Operational Change Management
ETMS	Enterprise Test Management Standards
FAFSA	Free Application for Federal Student Aid
FDD	Feature Driven Development
FSA	Federal Student Aid
GUI	Graphical User Interface
HTML	Hypertext Markup Language
IA	Information Assurance
ICD	Interface Control Document
ID	Identification
IEEE	Institute of Electrical and Electronics Engineers
IT	Information Technology
ITA	Integrated Technical Architecture
IVRU	Interactive Voice Response Unit
JAWS	Job Access With Speech

Acronym	Description
JMS	Java Message Service
LCM	Life Cycle Management
LOV	List of Values
MTP	Master Test Plan
NIST	National Institute of Standards and Technology
OMB	Office of Management and Budget
PC	Personal Computer
PIN	Personal Identification Number
PIR	Post Implementation Review
PISP	Post Implementation Support Period
PRR	Production Readiness Review
RCS	Reusable Common Services
RMI	Remote Method Invocation
ROE	Rules Of Engagement
RTM	Requirements Traceability Matrix
SA	Security Architecture
SAIG	Student Aid Internet Gateway
SAT	System Acceptance Testing
SCR	System Change Request
SLA	Service Level Agreement

Acronym	Description
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SOO	Statement of Objectives
SOW	Statement of Work
SSN	Social Security Number
Std	Standard
TDD	Tests Drive Design
TNR	Times New Roman
TRR	Test Readiness Review
TSV	Target State Vision
U.S.	United States
U.S.C.	United States Code
UAT	User Acceptance Tests
UDDI	Universal Description, Discovery and Integration
UI	User Interface
URL	Universal Resource Locator
VDC	Virtual Data Center
WAN	Wide Area Network
WBS	Work Breakdown Structure
WSDL	Web Services Definition Language

Acronym	Description
XML	Extensible Markup Language
XP	Extreme Programming

Appendix B: Glossary

Term	Definition
Acceptance Criteria	Exit criteria that a component or system must satisfy to be accepted by a user, customer, or other authorized entity.
Accuracy	Capability of the software to provide the right or agreed results or effects with the needed degree of precision.
Actual Result	Behavior produced/observed when a component or system is tested.
Ad hoc Testing	Testing carried out informally; no formal test preparation takes place, no recognized test design technique is used, there are no expectations for results and arbitrariness guides the test execution activity.
Agile	A software development methodology for providing a continuous stream of value to the customer at the minimum cost.
Application	Computer software/programs designed to perform work for end users of the computer – compare with system software, including middleware and operating systems, which perform functions within the computer and support the operation of application software/programs.
Application Owner	Government representative responsible for the application being developed. The representative is responsible for project initiation, budget approval and management, deliverable signoff and confirmation of project closure. Responsibilities during the test effort may be delegated to the test manager or project manager.
Audit	Independent evaluation of software products or processes to ascertain compliance to standards, guidelines, specifications, and/or procedures based on objective criteria.
Audit Trail	Path through a sequence of events that enables reconstruction of the history of the events. Typically the path from the output of a process to the original input (e.g. data), i.e. the history of the transaction. An audit trail facilitates defect analysis and allows conduct of process audits.

Term	Definition
Automated Testing	Testing that employs software tools that execute tests without manual intervention.
Baseline	Specification or software product that has been formally reviewed or agreed upon; that thereafter serves as the basis for further development and that can be changed only through a formal change control process.
Behavior	Response of a component or system to a set of input values and preconditions.
Best Practice	Superior method or innovative practice that contributes to the improved performance of an organization.
Black-Box Testing	Testing, either functional or non-functional, without reference to the internal structure of the component or system.
Capture/Playback Tool	Type of test execution tool where inputs are recorded during manual testing in order to generate automated test suites that can be executed later (i.e. replayed). These tools are often used to support automated Regression Testing.
Certification	Process of confirming that a component or system person complies with its specified requirements.
Change Control Board	Stakeholders representing the interests of Federal Student Aid and the contractor reviewing change requests, making recommendations, requesting impact analyses of proposed changes, and approving change requests.
Client/Server	Network architecture in which each computer or process on the network is either a client or a server.
Code	Computer instructions and data definitions expressed in a programming language or in a form output by an assembler, compiler, or other translator.
Code Coverage	Analysis method that determines which parts of the software have been executed (covered) by the test suite and which parts have not been executed, e.g. statement coverage, decision coverage or condition coverage.

Term	Definition
Code review and walk though	Peer review of source code intended to find and fix mistakes overlooked in the initial development phase, improving overall code quality.
Commercial Off-The-Shelf	Applications that are manufactured commercially and may then be tailored for specific use.
Compliance	Capability of the software product to adhere to standards, conventions, or regulations in laws and similar prescriptions.
Compliance Testing	Process of testing to determine the compliance of the component or system.
Component	Minimal software item that can be tested in isolation.
Component Interface Testing	Testing performed to expose defects in the interfaces and interaction between integrated components.
Component Testing	Testing of individual components of the code. These are the smallest product of code that accomplishes a specific action.
Condition	Logical expression that can be evaluated as True or False.
Condition Coverage	Percentage of condition outcomes that have been exercised by a test suite. 100% condition coverage requires each single condition in every decision statement to be tested as True and False.
Condition Outcome	Evaluation of a condition to True or False.
Configuration	Composition of a component or system as defined by the number, nature, and interconnections of its constituent parts.
Configuration Control	Element of configuration management, consisting of the evaluation, co-ordination, approval or disapproval, and implementation of changes to configuration items after formal establishment of configuration identification. [IEEE610].

Term	Definition
Configuration Control Board	Group of people responsible for evaluating and approving proposed changes to configuration items, and for ensuring implementation of approved changes. Also uses the acronym CCB.
Correctness	IEEE term for freedom from faults, meeting specified requirements, and meeting user needs and expectations.
Defect	Non-conformance of the application under test to its requirements. This includes any flaw in a component or system capable of causing the component or system to fail to perform its required function, e.g. an incorrect statement or data definition.
Defect Life Cycle	Cycle of finding a defect, reporting, assigning, coding, retesting, and closing the defect.
Defect Management	Process of recognizing, investigating, taking action and disposing of defects. Involves recording/classifying defects and identifying the impact.
Defect Management Tool	Tool that facilitates the recording and status tracking of defects.
Defect Priority	Process of assigning the priority in which a defect should be fixed.
Defect Report	Document that reports any flaw in a component or system that can cause the component or system to fail to perform its required function.
Defect Tracking Tool	See *Defect Management Tool*.
Deliverable Denial of Service Attack	Work product that must be delivered to someone other than the (work) product's author based on contractual guidelines. An attempt to make a computer resource unavailable to its intended users.
Development Team	Responsible for designing and performing unit testing and supporting the subsequent testing efforts by analyzing change requests and updating code as required.

Term	Definition
Development Testing	Formal or informal testing conducted during the implementation of a component or system, usually in the development environment by developers.
Documentation Testing	Testing the quality of the documentation, e.g. user guide or installation guide.
Enterprise Test Management	Responsible for the definition and compliance of organization wide testing standards and providing leadership and direction to the Test manager and Test team.
Entrance criteria	Set of conditions for permitting a process to go forward with a defined task, e.g. test phase.
Exit Criteria	Set of conditions agreed upon with the stakeholders, for permitting a process to be officially completed.
Expected Result	Behavior predicted by the specification, or another source, of the component or system under specified conditions.
Exploratory Testing	Interactive testing that combines test design with test execution and focuses on learning about the application. As the tester performs tests, the tester learns more about the application and uses that information to help design new tests.
Fail	Actual result does not match expected result.
Failure	Deviation of the component or system from its expected delivery, service, or result.
Feature	Attribute of a component or system specified or implied by requirements documentation (e.g., reliability, usability, or design constraints).
Post Implementation Verification	Period of time after the system enters production where defects are monitored closely and fixed via emergency fixes, etc. Also known as the stabilization period.
Post Implementation Verification Testing	Validation of critical functions of applications once the software is installed into production.

Term	Definition
Formal Review	Review characterized by documented procedures and requirements, e.g. inspection.
Functional Requirement	Requirement that specifies a function that a component or system must perform.
Functional Testing	Subset of system testing. Testing based on an analysis of the requirements of a component or a system.
Functionality	Capability of the software product to provide functions meeting stated and implied needs when the software is used under specified conditions.
Impact Analysis	Assessment of change to the layers of development documentation, test documentation, and components, in order to implement a given change to specified requirements.
Integration	Process of combining components or systems into larger assemblies.
Integration Testing	Phase of software testing in which individual software modules or units are combined and tested as a group.
Interface Control Document	Inputs and outputs of a single system or the interface between two systems or subsystems. The Interface Control Document (ICD) is used as a communication tool for test teams for Intersystem Testing.
Intersystem Testing	Integration test type that is concerned with testing the interfaces between systems.
Maintenance	Modification of a software product after delivery to correct defects, to improve performance or other attributes, or to adapt the product to a modified environment.
Management Review	Systematic evaluation of software acquisition, supply, development, operation, or maintenance performed by or on behalf of management. Monitors progress, determines the status of plans and schedules, confirms requirements and the system allocation, and evaluates the effectiveness of management approaches to achieve fitness for purpose.

Term	Definition
Master Test Plan	Test plan that typically addresses multiple test levels. See also *Test Plan*.
Metric	Measurement scale and the method used for measurement.
Middleware	Middleware (ITA/SA/ESB/EPI) manages the interaction and communication between disparate applications across heterogeneous platforms. It sits “in the middle” between application software working with different operating systems. Middleware is computer software that connects software components or applications such as Google (search), Team site (web content), WebSphere, etc. The software consists of a set of services that allows multiple processes for enterprise systems to work together and share data.
Module	See *Component*
National Institute of Standards and Technology	A measurement standards laboratory which is a non-regulatory agency of the United States Department of Commerce. The institute's mission is to promote U.S. innovation and industrial competitiveness by advancing measurement science, standards, and technology in ways that enhance economic security and improve quality of life.
Pass	Actual result matches its expected result.
Pass/Fail Criteria	Decision rules used to determine whether a test item (function) or feature has passed or failed a test.

Term	Definition
Peer Review	Review of a software work product to identify defects and improvements.
Penetration Testing	A method of evaluating the security of a computer system or network by simulating an attack from a malicious source, known as a Black Hat Hacker, or Cracker. The process involves an active analysis of the system for any potential vulnerabilities that may result from poor or improper system configuration, known and/or unknown hardware or software flaws, or operational weaknesses in process or technical countermeasures. This analysis is carried out from the position of a potential attacker, and can involve active exploitation of security vulnerabilities. Any security issues that are found will be presented to the system owner together with an assessment of their impact and often with a proposal for mitigation or a technical solution. The intent of a penetration test is to determine feasibility of an attack and the amount of business impact of a successful exploit, if discovered. It is a component of a full security audit.
Performance	Degree to which a system or component accomplishes its designated functions within given constraints in regards to processing time and throughput rate.
Performance Indicator	High-level metric of effectiveness and/or efficiency used to guide and control progressive development, e.g. lead-time slip for software development.
Performance Testing	Process of testing the degree to which a system or a component accomplishes its designated functions within given constraints in regards to processing time and data transfer rates.
Performance Testing Tool	A tool to support performance testing that usually has two main facilities: load generation and test transaction measurement. Performance testing tools normally provide reports based on test logs and graphs of load against response times.
Phase Level Test Plan	Test plan that typically addresses a single test phase. See also *Test Plan*.

Term	Definition
Post Implementation Support Testing	Post Implementation Support Period is the phase following the initial implementation of the application into the production environment during which the development contractor is responsible for defect resolution.
Post Implementation Verification	Post Implementation Verification is the final phase of testing that occurs after the application is deployed in production.
Production Environment	Hardware and software products installed at users' or customers' sites where the component or system under test will be used. The software may include operating systems, database management systems, and other applications.
Production Readiness Review	Final, formal, and documented decision point before a new application or a significant release of an existing application enters Federal Student Aid's production environment and is exposed to end-users.
Project	Unique set of coordinated and controlled activities with start and finish dates undertaken to achieve an objective conforming to specific requirements.
Project Manager	Responsible for the overall success of a development or maintenance project.
Program	See *Application*
Quality	Degree to which a component, system or process meets specified requirements and/or customer needs and expectations.
Regression Testing	Testing of a previously tested program following modification to ensure that defects have not been introduced or uncovered in unchanged areas of the software.
Requirement	Condition or capability needed by a user to solve a problem or achieve an objective that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.
Requirements Liaison	Responsible for facilitating communication between the project's Requirement Manager and Test Manager.

Term	Definition
Re-Testing Risk Based Testing	Re-running test cases that failed in order to verify the success of corrective actions. Testing based on identification of potential risks (or "candidate risks"), which should be analyzed by the project stakeholder or which might appear during the project's development. The tests may be created to address risks that are prioritized by how likely they are to manifest, and/or by the impact they would have if they do manifest.
Risk Identification	Process of identifying risks using techniques such as brainstorming, documents, and failure history.
Risk Management Rules of Engagement	Systematic application of procedures and practices to the tasks of identifying, analyzing, prioritizing, and controlling risk. Agreement between the Federal Student Aid CIO and application owner that specifies the depth of penetration testing, tools to be used, actions to be taken in the event a system is exploited or made unavailable.
Security	Attributes of software products that prevent unauthorized access, whether accidental or deliberate, to programs and data.
Security Testing	Testing to determine the security of the software product. See also *Functionality Testing*.
Security Testing Tool	Tool that provides support for testing security characteristics and vulnerabilities.
Service-Oriented Architecture	Architecture that relies on service orientation as its fundamental design principle. Service-orientation describes an architecture that uses loosely coupled services to support the requirements of business processes and users.
Software	Computer programs, procedures, and associated documentation and data, pertaining to the operation of a computer system – see also *Application.*
Specification	Document that specifies, in a complete, precise and verifiable manner, the requirements, design, behavior, or other characteristics of a component or system, and, the procedures for determining whether these provisions have been satisfied.

Term	Definition
Subsystem Testing	Testing of integrated components that form a major part of the whole system.
System	Collection of components organized to accomplish a specific function or set of functions.
Test	Demonstration of how well an application functions based on business objectives and requirements.
Test Approach	Implementation of the test strategy for a specific project.
Test Automation	Use of automated software to perform or support test activities, e.g. test management, test design, test execution, and results checking.
Test Case	Set of input values, execution preconditions, expected results and execution post conditions developed for a particular objective or test condition, such as to exercise a particular program path or to verify compliance with a specific requirement.
Test Case Management Tool	Tool for management of test cases for software and hardware testing. The standard test case management tool within FSA is Defect Management Quality Manager.
Test Condition	Item or event of a component or system that could be verified by one or more test suites, e.g., a function, transaction, feature, quality attribute, or structural element.
Test Cycle	Execution of the test process against a single identifiable release of the test object.
Test Data	Data that exists before a test is executed, and that affects or is affected by the component or system under test.
Test Environment	Environment containing hardware, instrumentation, simulators, software tools, and other support elements needed to conduct a test.
Test Execution	Process of running a test on the component or system under test, producing actual result(s).

Term	Definition
Test Execution Automation	Use of automated software, e.g. capture/playback tools, to control the execution of tests, the comparison of actual results to expected results, the setting up of test preconditions, and other test control and reporting functions.
Test Execution Phase	Period of time in a software development life cycle during which the software product is evaluated to determine whether or not requirements have been satisfied.
Test Input	Data received from an external source by the test object during test execution.
Test Item	Individual element to be tested.
Test Lead	Responsible for creating, tracking and maintaining phase level test plans, assigning tasks to the test team and reviewing deliverables.
Test Level	Group of test activities that are organized and managed together. Examples of test levels are component test, integration test, system test, and acceptance test.
Test Log	Chronological record of relevant details about the execution of tests.
Test Manager	Responsible for project management of testing activities and resources and evaluation of a test object.
Test Management	Planning, estimating, monitoring and control of test activities, typically carried out by a Test Manager.
Test Management Tool	Tool that provides support to the test management process. Contains capabilities, such as test ware management, scheduling tests, logging results, progress tracking, incident management, and test reporting. The standard test management tool within FSA is Defect Management Quality Manager.
Test Phase	Distinct set of test activities collected into a manageable phase of a project, e.g. the execution activities of a test level.

Term	Definition
Test Plan	Document describing the scope, approach, resources, and schedule of intended test activities.
Test Procedure	Document specifying a sequence of actions for the execution of a test. Also known as test suite or manual test script.
Test Process	Fundamental test process comprised of planning, specification, execution, recording, checking for completion and test closure activities.
Test Readiness Review	A multi-disciplined technical review to ensure that the subsystem or system under review is ready to proceed into formal test. The Test Readiness Review assesses test objectives, test methods, and procedures, scope of tests, traceability of planned tests to program requirements and user needs and safety and confirms that required test resources have been properly identified and coordinated to support planned tests. The Test Readiness Review determines the completeness of test procedures and the compliance with test plans and descriptions.
Test Scenario	See *Test Procedure Specification.*
Test Specification	Document that consists of a test design specification, test case specification and/or test procedure specification.
Test Strategy	High-level description of test levels to be performed and testing within those levels for an organization or program (one or more projects).
Test Suite	Set of several test scenarios, test cases, test procedures and test scripts for a component or system under test.
Test Summary Report	Document summarizing testing activities and results. Contains an evaluation of the corresponding test items against exit criteria.
Test Type	Group of test activities aimed at testing a component or system focused on a specific test objective.
Tester	Involved in the testing of a component or system.

Term	Definition
Testing	Process consisting of all life cycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that requirements are satisfied, to demonstrate that the application is fit for purpose and to detect defects.
Traceability	Ability to identify related items in documentation and software such as requirements with associated tests.
Unit Testing	Test used to validate that individual units of source code are working properly. A unit is the smallest testable part of an application.
Unit Integration Testing	Testing two or more software units that have been integrated to ensure that it works together as intended.
Usability Testing	Measuring how well people can use some human-made object (such as a web page, or a computer interface) for its intended purpose.
User Acceptance Testing Vulnerability Scanning	Formal testing with respect to Application Owner needs, requirements, and processes conducted to determine whether a system satisfies the acceptance criteria and to enable the user, customers, or other authorized entity to determine whether to accept the system. The automated process of proactively identifying vulnerabilities of computing systems in a network in order to determine if and where a system can be exploited and/or threatened. While public servers are important for communication and data transfer over the Internet, they open the door to potential security breaches by threat agents, such as malicious hackers. Vulnerability scanning employs software that seeks out security flaws based on a database of known flaws, testing systems for the occurrence of these flaws and generating a report of the findings that an individual or an enterprise can use to tighten the network's security.
Web-Based Architecture	Composed of services and technologies that enable applications to function in an internet and/or intranet environment through a web browser user interface.

Term	Definition
White Box Testing	Testing the internal structure of a component or a system with access to source code and configuration parameters of components. Also known as clear box testing, glass box testing, or structural testing.

Appendix C: Testing Techniques

1. Overview

To the untrained observer, software testing may appear to be intuitive, involving little more than running a program several times to see whether it meets requirements. In fact, software testing techniques are largely systematic, beginning with an analysis of the requirements before any code is written, and involving the need for testers to thoroughly analyze the system under development using various methods and tools. The following subsections provide an introduction and overview of the most common testing techniques.

Test Managers, Test Leads, Developers, and Test Engineers should utilize these techniques in their testing processes to obtain what the IEEE Standard Glossary of Software Engineering Terminology defines as "*Correctness"* or freedom from faults, meeting specified requirements, and meeting user needs and expectations.

2. Specialized Testing Techniques

Effective testing is critical to the success of every application development or enhancement effort. In addition to core application functionality, effective testing also includes the use of solid testing methodology and appropriate testing techniques on specialized areas related to the application (for example, Web based testing, 508 Compliance Testing, Commercial-off-the-Shelf (COTS) Products Testing, Disaster Recovery of Continuity of Services (CoS) Testing, Evergreen Testing, testing of specific architectures, or multiple environments, and Security Testing). In each case, finding, logging, and classifying defects by type are required to facilitate prompt correction.

3. ESB/ITA/SA Guidance

The test planning and test phase sections of this document apply to ESB/ITA (Infrastructure Technical Architecture/SA (Security Architecture) testing. The following sections provide a brief overview and examples of ESB, ITA and SA testing.

ESB TESTING

Enterprise Service Bus (ESB) testing tests the validation of the messaging infrastructure and integration capability. The main goal is to standardize interfaces to new and legacy systems in support of Federal Student Aid's Target State Vision (TSV).

ESB performs the appropriate application development tests on adapters and software components the ITA team has written (Application Programming Interfaces, Dynamic Link Libraries, etc.), using test procedures developed by ITA. ESB tests COTS components that support their mission prior to installing those components in the production environment (e.g., WebSphere, MQ (MQ Series and upgrades from IBM). In these cases, ESB uses test procedures developed by the VDC to verify that the infrastructure created is acceptable to perform COTS testing and is functioning properly, before and after installation of the COTS components. ESB supports developers by creating the middleware infrastructure in a development environment to allow

developers to operate in environments that mimic the production environment where the software will eventually operate.

An example of ESB Testing is testing the messaging capabilities of IBM WebSphere MQ during batch and real time transaction processing.

Impact of ESB Testing on Test Planning

During Test Planning, ESB works with ITA and the application teams to plan application tests and to create the required test suites. ESB and ITA create test environments to support the application developers. This environment helps to identify infrastructure issues and provides collaborative assistance with developers during the test effort.

Test Planning efforts conform to the procedures in Section 3 of this document. If steps are excluded, the Federal Student Aid Project Manager or Federal Student Aid Test Manager must approve the exclusions. Key planning considerations include selecting applications that could be affected by the updated ESB source code and the messaging infrastructure and appropriate mapping of the infrastructure to test activities. Test planning needs to define who is responsible for obtaining the test plans and test scripts from the application teams, updating them and executing the test suites and test scripts. Whenever there is a change in the functional requirements of an application, the appropriate test plans and test scripts must be updated. The plan should also address how the test results and metrics will be communicated to the Project Manager and Test Manager who will be included in the plan.

Impact of ESB Testing on Test Phases:

Unit Testing

Unit testing is performed whenever there is a modification in either the ESB source code or the MQ Series custom code. The procedure for Unit Testing is described in Section 3.2.

Integration Testing

Integration Testing is not applicable to ESB Testing.

System Testing

System Testing involves Intersystem Testing, Regression Testing, and Performance Testing. The procedure for System Testing is described in Section 4.4. The key element of System Testing is to validate whether the updated messaging infrastructure and/or ESV Source code integrates with the overall application infrastructure.

During System Testing, test scripts are created, automated testing is performed using E-metRx, and individual test results are documented. These scripts are stored in libraries and can be accessed at later date to carry out regression testing on the applications as changes take place in the middleware infrastructure components. For ESB to be effective in system testing, the recommendation is for the application team to store updated test plans, test suites and version/release information in an enterprise repository that is accessible to all teams as well as the VDC and ITA. It is also recommended that the application teams store other associated application artifacts, such as application architecture diagrams, requirements, and PRR documents.

Each test suite will be executed as a standalone unit, and the results will be documented in a Test Run Table in the format shown below. Output files, where indicated, will be checked for status. If the status is "true" and all observed results meet the expected results, the test will be deemed a success (Pass). If a "false" status is found, one or more issues will be logged and submitted for resolution with the test deemed not successful (Fail). When modifications to correct the problem(s) have been applied to the test environment, the failed test suite will be rerun. This process will be repeated as necessary until all exceptions are satisfactorily resolved. Specifically, when a failure is encountered, the issue will be identified, logged, and corrected, and the test suite will be run in its entirety from start to finish. If a test does not produce the "Expected Result(s)," then the test will be deemed a "Fail." If a test suite does produce the "Expected Result(s)," then the test will be deemed a "Pass."

Test Run Table

Test Suite #	Step #	Date	Exp. Results = Actual Results	Pass / Fail	If Fail, enter Issue #

UAT

UAT involves Intersystem Testing, and Regression Testing of the affected applications to test updates in the messaging infrastructure and the ESB source code. The procedure for UAT is described in **Section 4.5**. UAT involves test execution of test suites written for infrastructure and service orchestration, migration, mediation, governance, and security. UAT includes verification, management and monitoring requirements, inspection of documents, logs and dash boarding components, verification of training and deliverable requirements and conducting a training session.

Post Implementation Verification

Post Implementation Verification is carried out as shown in Section 3.6 of this Document.

ITA TESTING

Integrated Technical Architecture (ITA) testing validates the integrated, enterprise wide technical architecture on which the Federal Student Aid applications are deployed. The testing mainly involves validating the COTS products, Architectural Upgrades, and Development and Support.

ITA creates and maintains a series of RCS components that provide connectivity between individual applications, and software appliances such as Google's search engine. Federal Student Aid has acquired these appliances to avoid recreating functionality already available in COTS products. ITA currently performs four types of tests as listed below:

RCS – Testing to support the development and maintenance of components that provide interfaces with appliances that are integrated with Federal Student Aid

applications (testing is similar to testing during application development and maintenance).

Evergreening – Testing of COTS product upgrades during integration into the Federal Student Aid environment. Regression testing is conducted to ensure that introduction of the new COTS product or upgrade doesn't break the existing infrastructure.

Integration – Installation of new or updated applications into the production environment (similar to Evergreening), integration testing is conducted to ensure that the new or updated application doesn't break the existing infrastructure, but this testing ensures that the application functions correctly within the infrastructure.

Password changes – Federal Student Aid policy requires changing applications' database passwords every 60 to 90 days. ITA performs limited regression testing following these changes to ensure that applications continue to function correctly.

An example of ITA Testing is to test the advance search capabilities of the Google appliances installed in the ITA environment.

Impact of ITA Testing on Test Planning:

Test Planning efforts must be followed in the same manner as other test efforts and must conform to the procedures in **Section 2** of this document. If steps are excluded, the Federal Student Aid Project Manager or Test Manager must agree to the steps. Test planning needs to take into account who will execute the test suites and test scripts (the Application Test Team (Federal Student Aid Application Owners) or the ITA Team) and how the test results and metrics will be communicated to the Project Managers and Test Managers.

The functional and non-functional requirements for COTS products upgrade testing are defined using expert judgment and vendor support documentation. Since each product upgrade affects specific functionality, the scope of testing requirements is defined by the scope of the changes in the upgrade.

Based on input from vendor documentation, vendor support, and internal team discussions, ITA identifies the critical tests in order to ensure the success of the infrastructure component upgrade.

As part of ensuring communication of enterprise wide changes to infrastructure components, the EOCM process monitors all major changes. ITA provides testing schedules, and the EOCM committee oversees the coordination of the change process. For smaller changes, the team that requests and/or performs the change monitors the testing.

Impact of ITA Testing on Test Phases:

Unit Testing

Unit testing is carried out whenever there is a modification to either the RCS code or the Custom Code of the COTS products. The procedure for Unit Testing has been described in Section 3.2.

Integration Testing

Integration Testing is not conducted in ITA Testing.

System Testing

System Testing involves Intersystem Testing, Regression Testing, and Performance Testing. The procedure for System Testing has been described in **Section 3.4**. The key element of System Testing is to validate whether the updated architecture, RCS Code, and/or the COTS products integrate well with the overall application infrastructure.

ITA uses internally developed procedures for testing infrastructure component upgrades, tailored for the specific environment where the upgrade is performed. The baseline for developing the test procedures is the vendor support documentation (release notes, installation and administration guides, and best practices).

For each testing activity, ITA records the outcome either in a test matrix (developed internally) or in the change request record.

Currently, the ITA tech lead is responsible for validating the testing procedures, Test Readiness Review, and testing activities to be performed. The ITA Test Team performs the testing for each upgraded component.

UAT

UAT involves Intersystem Testing, and Regression Testing of the affected applications as a result of updates in the Architecture, RCS, and/or COTS products. The procedure for UAT is described in **Section 3.5**.

The testing requirements in this category are defined by the application teams and incorporated in the documented testing processes. ITA provides test requirements input, based on expert judgment, when requested by the application teams.

Post Implementation Verification

Post Implementation verification is conducted as shown in **Section 3.6** of this document.

SA TESTING

Security Architecture (SA) testing involves validation of access controls, provisioning, de-provisioning, self-registration, delegated administration, and simplified sign-on across the Federal Student Aid enterprise. The architecture includes access management tools, identity management tools, enterprise policy repositories, enterprise user repositories, and other related security components.

SA tests fall into one of three categories:

1. In an SA only change, a document is used to evaluate the change within the security architecture. SA then invites all of the application groups to test applications in the new environment.
2. For an application only modification, SA performs an application specific review of the modified application to ensure that the application still works within the SA environment.
3. For Integration related changes, SA performs a high-level review to ensure that other applications continue to work with the new application.

Note: *Integration related changes occur the first time an application and the SA environment are integrated. When this occurs, SA invites the other production applications to retest SA related functionality to determine whether the integration of the new application broke any existing functionality.*

As shown below, the Security Architecture Test/Certification Matrix document provides a document for testing changes and certifying the integrity of the Security Architecture Production environment. The matrix is used to ensure that the functionality and operability of the Security Architecture environment has not been adversely affected by configuration changes, patch installation, or any other changes. The tester should follow the test matrix and provide answers along with comments when appropriate. The current process ends when the Go/No Go decision is made for each change.

Security Architecture Test/Certification Matrix

Critical or Non-Critical Item	Test/Verification Item	Results (Yes/No)	Item Verified by	Comments

An example of SA Testing is to check the password parameters enforced by Tivoli Access Manager by validating the access rights of appropriate users.

Impact on Test Planning:

Test Planning efforts must be followed as they are in other test efforts and must conform to the procedures in **Section 3** of this document. If steps are excluded, the Federal Student Aid Project Manager or Test Manager must agree to those steps. The test planning needs to take into account how the test results and metrics will be communicated to the Project Managers and Test Managers.

Testing requirements are driven by the upgrade functionality. The SA Test matrix lists the functional requirements and non-functional requirements for each COTS Upgrade. The SA team manages the test effort. In order to mitigate risks, the SA Team and SA integrated application representatives are involved in testing. The SA Team and integrated application Federal Student Aid Application Owners decide what will be tested and how to measure success or failure.

Impact on Test Phases:

Unit Testing

Unit Testing is not conducted in SA Testing.

Integration Testing

Integration Testing is not conducted in SA Testing.

System Testing

System Testing involves Intersystem Testing, Regression Testing, and Performance testing. The procedure for System Testing has been described in **Section 4.4**. The key element of System Testing is to validate whether the security architecture integrates well with the overall application infrastructure. The Test/Certification Matrix is used for System Testing.

UAT

UAT involves Intersystem Testing, and Regression Testing of the affected applications as a result of the updates in the Security Architecture. The procedure for UAT has been described in **Section 3.5**. The Test/Certification Matrix is used for UAT.

Post Implementation Verification

Post Implementation Verification is conducted as shown in **Section 3.6** of this document. The Test/Certification Matrix is used for Post Implementation Verification.

4. Infrastructure/Application Testing

Testing is mandatory for all changes to the Data Center infrastructure, such as servers, operating systems, system software, network connectivity, passwords, etc. in the Development, Test, Performance, Staging and Production environments. Testing is mandatory, for all Production changes, such as middleware, application, databases, etc. Testing is highly recommended for all non-production environments, such as Development, Test, Performance Test, and Stage. Testing of application software changes is mandatory in a pre-production environment prior to the movement of those changes to the production environment.

If there is no participation for mandatory testing, such as a delay with testing or the level of testing needs to be determined, etc., a justification is required from the Data Center Vendor or the FSA Application Business Technical Lead to the Technology Office VDC Management Team. Also, if the Data Center Vendor, FSA Application Business Technical Lead or designee and Technology Office Middleware Technical Lead or designee decides, not to participate in mandatory testing, the VDC Management Team will facilitate the discussion on whether or when testing will occur. The process for non-participation in testing is as follows:

1. The FSA Application Business Technical Lead, Technology Office Middleware Technical Lead, or the Data Center Vendor, or designees, will notify the Technology Office VDC Management team of non-participation of mandatory testing.

2. Technology Office VDC Management, the Data Center Vendor, FSA Application Business Technical Lead, Technology Office Middleware Technical Lead or designees will collaborate to determine whether or when the test will be performed.
3. If there is not an agreement between Technology Office VDC Management, Data Center Vendor, FSA Application Business Technical Lead, Technology Office Middleware Technical Lead or designees, each one is to elevate the issue to the next level of management.
    a. Technology Office Management is notified by the Technology Office VDC Management lead or designee
    b. Technology Office Management is notified by the Technology Office Middleware Technical Lead or designee
    c. Application Owner is notified by the FSA Business Technical Lead or designee
4. Decision to test or not test is decided by Technology Office Management and the Application Owner

Required Infrastructure Testing at a minimum includes the following:

- Network/Firewall Updates
- Testing is mandatory for all mission critical and mission important applications.
- Domain Name System (DNS) (Example: Mail/Relay Services)
    - Testing is mandatory for those projects specifically impacted by the DNS change or the new DNS entries.
- Server Patches (Example: Daylight Savings Time)
- New Servers (Includes New Databases and new Custom Systems)
    - Testing of configurations (capacity/settings) of servers.
    - The server must be tested for stability, functionality and security.
    - All new servers must have security scans to point out security vulnerabilities (based on security requirements).
- COTS Upgrade (Examples: TIM/TAM or Informatica)
    - Integration, system and stress testing is mandatory.
    - Security Scans prior to production are required to point out security vulnerabilities.
- Application/System Maintenance Releases (Code Promotion/Install to Production; Example: XML Registration)
    - Security testing is part of C&A requirements.
    - Middleware team will be required to perform system and integration testing.

  - Deployment/installation testing will be performed as part of Post Implementation Verification.
- Application/System Level Database Changes
  - Security scans must be performed to identify potential security vulnerabilities.
  - Validation by systems team mandatory for all mission critical and mission important applications.
  - System component testing of database changes will follow the standards set in this document.
- Application/System password Changes by data center (Example: Oracle Database Password)
  - Validate that access is not in jeopardy.
- Operating System Upgrades (Example: Mainframe, CA-Gen, Unix/AIX)
  - Security scans must be performed to identify potential security vulnerabilities.
  - Intersystem testing may be necessary to ensure continued connectivity.
- In addition, remaining configured items not listed and items noted by the Data Center Vendor and the Technology Office VDC Management Team.

5. Service Oriented Architecture Testing

Service Oriented Architecture (SOA) is an approach to constructing enterprise systems from a set of collaborating services which, when combined, are comprised of business use cases. The services are self-describing and can be assembled ad hoc to comprise business processes.

Web services are a common platform that enables SOA. There are three main components to web services:

1. **Simple Object Access Protocol (SOAP):** Provides the envelope for sending messages to web services
2. **Web Services Definition Language (WSDL):** Forms the basis for web services
   - A service provider describes its service using WSDL
3. **Universal Description, Discovery and Integration (UDDI):** Registries can be searched to quickly, easily, and dynamically find and use web services

Note: *The services used in SOA are not limited to web services, but can include other technologies such as Distributed Component Object Model (DCOM) and XML over Remote Method Invocation (RMI).*

The following are key best practices when testing SOA and, in particular, testing web services:

- Unit and Functional Testing should occur as early as possible in the development cycle and should provide a basis for developing regression test cases

- Regression test cases must be used frequently to ensure the robustness of each service.
- Intersystem Testing practices must be considered during System Testing and UAT, since SOA often implies interfacing with external systems.
- Test suite management and setup is paramount to reducing the time required for setting up and maintaining functional and regression tests.
- Processing of services can involve complex interaction of legacy and third party systems.
- End-to-end visibility of each component involved in the processing of a specific service to identify and resolve potential issues must occur.
- Test Harness – is frequently a good way to "institutionalize" exposing functionality of the system (e.g., via Web Services) for coarse-grain functionality typical for acceptance tests and major system scenarios (and even external invocation). Wiring up a test harness enables ease of extending the breadth of the tests as new functionality is added. You can build test harnesses that are very automated in terms of capturing the user input (using a record mode) and capturing the output, and getting a user's (in this case typically a tester) acknowledgement that the output is correct. Each captured test suite is added to a test database for later playback orchestration.
- Test coverage in SOA must consider how each element operates within a larger process, not just how an element operates in isolation.
- SOA testing best practices need to assign risk weights to each element as part of the larger overall process.
- When testing SOA, understanding how each service functions within dependent processes and how the dependent processes function end-to-end, is critical to determining proper test coverage.
- Testing and de-bugging issues in SOA requires a team effort that provides visibility into the entire path of the service.
- Identifying proper test coverage and the ability to diagnose and fix issues will often involve not only the tester, but operations experts, developers, process matter experts, and business analysts.
- The process for testing best practices for SOA should be as follows:
- Component interfaces should be tested as they are when added to the overall integration architecture.
- If the process requires systems to be available for complete end-to-end testing that are unavailable or not built, this problem can be reduced significantly by simulating missing components in the testing process.
- Testing of SOA should focus on the following areas:

- **Functionality**: The service must return the expected response for all possible input values.
- **Performance**: The response time must be acceptable for a single instance of the service call and under anticipated load conditions. The test must check for scenarios that may cause performance degradation.
- **Backward Compatibility**: As business evolves, new consumers of the service may force changes in the input and output values. These changes must not compromise functionality for the original consumers.
- **Inter-operability**: The service must maintain correct functionality on all platforms and with all communication protocols intended for use. Proper inter-operability testing requires a test environment that simulates as closely as possible the real world production environment.
- **Compliance**: Each service must adhere to governing standards to maintain integrity and robustness. A compliance-testing tool can be used to verify compliance of each service during development.
- **Security**: Each service must comply with security constraints. Web services exposed to the Internet can pose a significant security risk.

An example of an SOA developed for Federal Student Aid is the PIN web service. This service provides validation that a specific combination of SSN, first two letters of a last name, DOB, and PIN exists in the PIN database. The service returns either a correct response if the record exists or an incorrect response if the record does not exist. Multiple applications may use the service, and this should be considered when planning the testing.

Impact on Test Planning

Planning for SOA testing differs from traditional enterprise application test planning, primarily in the area of project test strategy. SOA testing adds considerable complexity to and emphasis on the following areas:

- **Test Coverage**: Testing must encompass how each service operates within the broader business processes.
- Since there will likely be many different business processes utilizing each service, process visibility is critical to defining process coverage.
- Traditional application testing assures that the user interface performs as expected. SOA testing needs to go behind the user interface to expose the entire process and the data driven through the process.
- Since SOA often implies interfacing with external systems, coordination with other groups should take place early for Intersystem Testing. This can be a major factor in determining testing schedule. (See Appendix D.23 Intersystem Testing.)

- **Test Tools**: Test automation plays a more significant role in SOA Testing because of the necessity to run Functional, Regression, and Performance Testing more frequently within the development lifecycle.
- Testing tools must be able to test other technologies that can expose services, such as Java Message Service (JMS), which is a middleware; Application Programming Interface (API) for sending messages between two or more clients; and RMI, which is an API for performing remote procedure calls.
- **Training Requirements**: SOA testing may require more specialized skills in areas such as writing custom interfaces to access services or to simulate missing components required to run services (e.g., simulate a back-end or third party component).
- **Level of Regression Testing**: For SOA, regression testing typically occurs more frequently due to changes or additions in dependent services and processes.
- Because services are shared resources, a change to one service can have potential impacts on many processes.

Impact on Testing Phases

SOA testing encompasses the same testing phases as traditional enterprise applications; however, emphasis is placed on the following:

- Unit and Functional testing should occur as early as possible in the development cycle and should provide a basis for developing regression test suites.
- Within the Functional Testing phase, the ability to perform asynchronous testing is often required.
- With SOA, services are often not synchronized so at times there may be a need to test business processes out of sequence.
- The test team needs the ability to address asynchronous communications mechanisms such as the polling of data, polling-through-proxy, and callbacks through WS-Addressing as well as other standards.
- More frequent Regression Testing is typically necessary to cope with potential impacts to changes in services shared by many different processes.
- Intersystem Testing practices must be considered during System Testing and UAT, since SOA often implies interfacing with external systems.
- Network sensitivity testing is more crucial for SOA because components typically are distributed over a Wide Area Network (WAN), which can impose significant performance limitations.
- Within all testing phases, it is important to analyze and understand the impact of changes.
- One of the major challenges in SOA testing is the decision of how to effectively test changes that occur.

- A determination must be made as to which changes potentially cause major risks and where the best places are for the test team to invest efforts, as well as where minimal validation is needed to optimize resources.
- In traditional application testing processes, the testing challenge has been to understand the changes development made and decide whether testing is needed. Understanding the impact of changes is more significant in shared services due to the number of services involved, the number of internal dependencies, and the number of applications using shared services.

6. Evergreen Testing

Federal Student Aid needs to remain up to date with newer versions of COTS software. To facilitate these updates, Federal Student Aid has developed a policy called Evergreen Testing (or Evergreening) for keeping software and hardware resources up to date. The configuration and other changes to the infrastructure and COTS products should be published periodically.

For example, a simple Evergreening policy might involve:

- Replacing all computers older than four years with new models
- Acquiring the most recent operating system release
- Updating the operating system software to a newer version using the existing hardware

Evergreening policy should state whether the purpose is:

- Replacing technology to maintain an acceptable level of reliability
- Acquiring improved speed and capacity at an acceptable price
- A combination of the two

Whatever the purpose, the Evergreening process is vital to ensure that Federal Student Aid systems implement the latest version. By implementing an effective Evergreen Testing policy, Federal Student Aid manages the risks associated with the integration of a new version of COTS software.

An example of Evergreen Testing:

An update to a newer version of COTS software or the testing of an application that has been integrated with a newer version of WebSphere requires:

Test Planning

Test Planning must conform to the procedures in **Section 3** of this document.

Test Phases

- **Unit Testing:** does not require Evergreen Testing.
- **Integration Testing**: does not require Evergreen Testing.
- **System Testing**: may require Evergreen Testing as shown in **Section 4.4** of this document.

- **UAT**: may require Evergreen Testing as shown in **Section 4.5** of this document.
- **Post Implementation Verification**: may require Evergreen Testing as shown in **Section 4.6** of this document.

7. Graphical User Interface (GUI) Testing

GUI Testing, sometimes referred to as User Interface (UI) testing, involves testing the interaction between the software and the user. This includes how the application handles keyboard and mouse input and how the interface displays screen text, images, buttons, menus, dialog boxes, icons, toolbars, and more.

Before GUI testing can take place, the GUI and underlying functionality must exist. Initially, testers normally perform functional GUI testing manually. As the GUI becomes more stable, testing may move into a more automated process where faster Regression Testing can be implemented to validate that functionality has not been broken due to subsequent software changes.

Correctness is important for software in general; however, it is particularly important for user interface software and GUI Testing. The GUI interface represents the aspect of the software that is directly visible to users. If the user interface is incorrect, the user perception will be that the software is incorrect, regardless of the correctness of the underlying functionality. Software correctness includes:

1. **Verification**: Verification failure results in software containing potential faults or flaws.

- Testers need an effective understanding of the purpose and functionality intended for the interface, and testing must validate that all of the interfaces function as designed.
- Testers testing the GUI need to understand that the behavior of the GUI is extremely important.

2. **Validation**: Questions that need to be asked include:

- Can windows be resized, moved, and scrolled?
- Does the window properly regenerate when overwritten and then recalled?
- Are all relevant pull-down menus, tool bars, scroll bars, dialog boxes, buttons, icons, and other controls available and properly displayed?
- Is the name of the window properly represented?
- Do multiple or incorrect mouse clicks within the window cause unexpected side effects?
- Does the window close properly?
- Are consistent color schemes used?
- Overall screen layout – tester should understand the reasoning behind choices of LOVs, checkboxes, and list items, multi-record vs. single record display, push buttons, and radio groups.

- Do pull-down menu operations work properly?
- Are all pull-down menu functions listed correctly?
- Are character text, size, placing, and format correct?
- Does each menu function perform as advertised?
- Does alphanumeric data entered fit within entry restraints?
- Do graphical modes of data entry work properly?

An example of GUI Testing is:

Testing a drop down box on a page to make sure the drop down box populates correctly and that data is selectable.

The testing of the Graphical User Interface can be implemented in all of the five test phases within in the Enterprise Test Management Standards (ETMS), and will have to comply with ETMS rules and standards for each of the phases.

8. Web Based Testing

Web-Based Testing involves testing applications via web browsers. Testing web-based applications is similar to testing any other application in terms of testing the functionality. The differences come from the distributed nature of the web environment. For example, when an error occurs, it is difficult to determine the exact location of the error.

Specific testing areas in web testing include the following:

- **Content Checking**: Inspecting the web application for consistency, accuracy, spelling, and accessibility
- **Browser Syntax Capability**: Even if the web application runs on only one browser (e.g., Microsoft Internet Explorer), the browser syntax must be checked for compatibility since there are many versions of Microsoft Internet Explorer.
- **Functional Testing**: Functional Testing mainly focuses on page level and transaction level testing.
- Page level testing requires client code to check the field validations.
- Transaction level testing mainly focuses on whether the information entered by the user at the web page level is updated in the database and proper data is returned to the user.
- **Performance**: Performance testing examines the throughput of the web application as the number of users and/or transactions is increased.
- **Security**: Security testing checks for web application security vulnerabilities.

An example of Web Based Testing is to validate the consistency of navigation between web pages testing each HTML link.

Test Planning

Test planning must conform to the procedures in **Section 3** of this document.

Test Phases

- **Unit Testing** may require Web Based Testing as described in **Section 4.2** of this document. Validation uses a tool (such as an HTML Validation Service) to validate the HTML syntax and browser compatibility.
- **Integration Testing** may require Web Based Testing as described in **Section 4.3** of this document. Test examples include clicking to check every page and link of the website to ensure content was migrated to the correct page.
- **System Testing** may require Web Based Testing as described in **Section 4.4** of this document.
- **User Acceptance Testing** may require Web Based Testing as described in **Section 4.5** of this document. Test examples include checking the consistency of the look and feel of the web site and checking the ease of navigation.
- **Post Implementation Verification** may require Web Based Testing as described in **Section 4.6** of this document.

9. 508 Compliance Testing

Section 508 of the Rehabilitation Act of 1973 requires Federal agencies to develop, procure, maintain, or use electronic and information technology to ensure that:

- Federal employees with disabilities have access to and use of information and data that is comparable to that available to Federal employees who do not have disabilities, unless an undue burden would be imposed on the agency; and
- Members of the public with disabilities seeking information have access to and use of information that is comparable to that available to individuals without disabilities.

These technology-specific provisions address:

- Software applications and operating systems;
- Web-based information or applications;
- Telecommunications products;
- Video or multi-media products;
- Self-contained, closed products such as information kiosks and transaction machines; and
- Desktop and portable computers.

The four-step procedure to carry out 508 compliance testing includes:

- Developing a test plan based on the Section 508 compliance requirements outlined by the client.

- Conducting 508 compliance testing by a contractor, or by Federal Student Aid staff.
- Conducting 508 compliance testing using the Assisted Technology Partnership Group (ATP) from The Department of Education's AR Process Brief Request Form, which may be found on the ConnectED site.
- Documenting all feedback from outside the organization according to Section 508 compliance.

The Contractor has full responsibility to conduct 508 compliance testing of the application. The Assistive Technology Partnership (ATP) group will verify compliance after the contractor test team has conducted the 508 compliance testing. The Contractor is also responsible for assisting the ATP Group during their test efforts.

The Contractor must include the following in the Test Plan for 508 Compliance Testing:

- Identify the tool being used to perform 508 testing (e.g., JAWS)
- Coordination and support with ATP to include:
    - Scheduling of tests
    - Minimum two weeks' notice (standard is three weeks)
- On site contractor support

Additionally, the results of the 508 compliance testing are required to be provided to Federal Student Aid separately from other test results, as they will be required during the Production Readiness Review.

Note*: Federal Student Aid will provide Section 508 compliance guidelines at the time of contract award.*

An example of 508 testing:

When software is designed to run on a system that has a keyboard, product functions shall be executable from a keyboard where the function itself or the result of performing a function can be discerned textually.

Test Planning

Test planning must conform to the procedures in **Section 3** of this document.

Test Phases

- **Unit Testing** may require 508 Compliance Testing.
- **Integration Testing** does not require 508 Compliance Testing.
- **System Testing** may require 508 Compliance Testing.
- **UAT** may require 508 Compliance Testing.
- **Post Implementation Verification** does not require 508 Compliance Testing.

The following Accessibility Review Process Brief provides additional guidance on Section 508 Testing:

ED Accessibility Review Process Brief, a.k.a., Section 508 Testing

This application is being tested to determine how well it complies with the standards established by the Access Board that implements Section 508 of the Rehabilitation Act of 1973 as amended in 1998. For further information on the applicability of Section 508 to the purchase of electronic and information technology, please refer to the ACS Directive, "Procuring Electronic and Information Technology under Section 508 of the Rehabilitation Act":

In conducting this assessment, the Department of Education's Assistive Technology Team will utilize the following protocol:

1. The testing team shall consist of one or more members of the Assistive Technology Team: an expert user of the application/site who is familiar with keyboard shortcuts, familiar with all modules which will be used at Education and how they will be utilized, and someone with sufficient access to the application development team to intelligently transmit the findings of the review to the development team for purposes of remediation. When possible, a member of the actual development team shall be present, as this is the most effective means of ensuring that accessibility problems are fully understood by the vendor. The review will be canceled if someone with the knowledge and skills described above is not present at the allotted time.
2. The set of standards used (Part 36 CFR 1194) will be comprised of the Section 508 Part B Technical Standards that are directly applicable to the type of application in question. For example, all web sites and applications will be evaluated against 1194.22, the "Intranet and Internet-based Applications" standards. All software applications will be evaluated against 1194.21, the "Software Applications and Operating Systems" standards. Those web-based applications that contain non-HTML content will, when appropriate, invoke the software standards when dictated by web standard 1194.22(m).
3. All tests will take place at Education headquarters, 400 Maryland Avenue, Southwest, in Room BC-101 at one of the designated testing machines. Requests to review applications at locations other than our testing area will be considered on an individual basis and only honored as staff resources permit.
4. In consideration of those with chemical sensitivity, the Assistive Technology Team requests that all persons attending this meeting refrain from wearing any scented products including perfumes, cologne, aftershave, hairspray and other strong fragrances. For more information regarding Multiple Chemical Sensitivities, please reference the Fragrance Free-Environment policy from the United Stated Access Board
5. ***ED or vendor personnel requesting an accessibility review should allow at least two weeks' lead time to schedule the review after contacting the team.** Requests for emergency reviews requiring a quicker turn-around will be honored as staff resources permit.

6. Tests will last up to but be no longer than two hours in length. For in-depth, complex applications that may require additional time, more than one session may need to be scheduled to complete the review process.
7. ***Some applications may require passwords and user names to be supplied, firewall access to a development network, specialized installations of software or drivers on the test machine, or similar pretest arrangements. It is the responsibility of the ED application sponsor requesting the review to work with engineering and assistive technology staff to ensure that all such preparatory actions have been completed prior to the time of testing. All tests for which such preparations have not been completed will be canceled and rescheduled for a future time.**
8. As stated above, when possible, an individual representing the company responsible for developing the application being tested, with sufficient knowledge of programming to communicate specific standard-related findings back to company development staff, shall be present. This is to ensure that detailed information concerning any non-compliant user interface elements discovered during the testing process can be adequately communicated to development staff, making remediation as efficient and effective as possible.
9. Due to the size and complexity of most client and web-based applications or sites, only a representative sample of the user interface elements will be tested during the formal review process. It is the sole responsibility of ED staff and vendor representatives present at the review to guide the testing team to samplings of all user elements relevant to the applicable standards. To assist in this process, the testing team will verbalize and explain each applicable standard, requesting of the project or company representative that samples of each standard be evaluated during the review process. *** Failure to draw attention to controls or elements that are later found to be non-compliant will not exempt these elements from the requirements imposed by the Section 508 standards.** It is for this reason that we require someone present who is extremely familiar with all aspects of the application interface, as it will be deployed in the ED environment.
10. Various testing tools will be used to evaluate the accessibility of interface elements, and may consist of screen readers, screen magnifiers, and manual or automated code examination. The selection of tools is at the discretion of the testing team, which will choose the tools most appropriate for the given situation.
11. A report briefly outlining the findings of the team will be prepared within three business days of the review. If the application fails 508 compliance testing, it will be the sole responsibility of the ED sponsor to proceed with one of the following options: a. require remediation and retest of the application prior to deployment; b. choose a product which conforms to the Section 508 standards; or c. invoke a Section 508 exception and inform the CCRB of the decision to implement. The vendor, in consultation with the ED project manager, shall make every attempt to remediate inaccessible application elements that are discovered to be non-compliant after the formal testing process. Members of the Assistive

Technology Team will be available to assist development staff on a limited basis, providing consultation on the standards and testing methods during the remediation process.

12. All applications that are installed on the Assistive Technology workstation used for testing will reside on the machine for a period not to exceed two weeks. When remediation requires a time period between the initial test and the retest that exceeds this time period, the ED project manager may be responsible for ensuring that the application is reinstalled on the test machine.
13. All upgrades, updates, and new versions of client-server and web-based applications that make any change to the user interface shall be tested for compliance.
14. The results of all accessibility reviews are considered to constitute ED's best understanding of the application of the Section 508 implementing standards, and are provided as advisory guidance to ED customers and vendors. The decision to deploy a given application which does not conform to the set of standards against which it is tested rests solely with the ED project manager after seeking advice from the Assistive Technology Team in such areas as: remediation, potential liability, Section 508 exceptions which may apply, etc.
15. The Assistive Technology Team's Accessibility Review Process, and Section 508 Guidance may be found on the ConnectED site.

10. Commercial-off-the-Shelf (COTS) Software Testing

COTS software is commercially available software sold with the expectation that the source code will not be available and that documentation will be limited to installation and end user instructions and information. Federal Student Aid integrates COTS software with a number of Federal Student Aid applications. Testers use Black Box Testing techniques since there is no access to the product's source code.

Before integrating COTS software, Federal Student Aid must have a thorough understanding of the compatibility of the COTS product. In most integrated COTS software systems, Functional Testing is limited. When working one-on-one with the contractors or developers of COTS software, developers must follow a thorough testing process to insure that software complies with requirements of the COTS software. Access to thorough documentation and a help desk can mitigate the risk and reduce testing failures that can occur when integrating third party software.

One of the least tested but most critical features of software applications is error/exception handling. Error/exception handling routines are the safety net for any system to handle unexpected circumstances, such as when operating system software or hardware failures occur. As more critical applications are developed using or through integration with commercial-off-the-shelf software, the sensitivity of these applications to operating system failures, and in general to failures from third party software, becomes increasingly critical. By working closely with the developers and having proper documentation and test results, these risks can be limited. Maintaining newer versions of the COTS software reduces the number of known issues in the production environment.

An example is testing the integration of a COTS product (e.g., WebSphere MQ) that is installed to enable two applications to transfer data from one to the other. These tests would also include evaluating the COTS product's performance when one of the two applications goes down for a period of time and then comes back up, to evaluate how well the data transmitted by one of the applications during that period is transmitted once both applications are operating normally.

Test Planning

Test planning must conform to the procedures in **Section 3** of this document.

Test Phases

- **Unit Testing** is described in **Section 4.2** of this document.
- Unit Testing is only required when the custom code of the COTS product is modified.
- **Integration Testing** may require COTS Testing as described in **Section 4.3** of this document.
- **System Testing** may require COTS Testing as described in **Section 4.4** of this document.
- **UAT** may require COTS Testing as described in **Section 4.5** of this document.
- **Post Implementation Verification** may require COTS Testing as described in **Section 4.6** of this document.

11. Client Server Architecture Testing

Client Server describes the relationship between two computers in which a program in one computer, the client, makes a request to another computer, the server. Typically, servers are specialized computers dedicated to a single task, e.g., managing disk drives (file servers), printers (print servers), or network traffic (network servers). Clients are PCs or workstations on which users run applications. Clients may rely on servers for resources, such as files, devices, and even processing power.

An example of Client server architecture testing is testing concurrent query updates. The following scenarios can be tested with a client installed on two different machines:

- Both clients access the server at the same time – One updates a record and the other tries to access the same record
- Both clients access the server at the same time – One deletes a record and the other tries to access the same record

Test Planning

Test planning must conform to the procedures in **Section 3** of this document.

Test Phases

- **Unit Testing** may require Client Server Testing as described in **Section 4.2** of this document.

- **Integration Testing** may require Client Server Testing as described in **Section 4.3** of this document.
- **System Testing** may require Client Server Testing as described in **Section 4.4** of this document.
- **UAT** may require Client Server Testing as described in **Section 4.5** of this document.
- **Post Implementation Verification** may require Client Server Testing as described in **Section 4.6** of this document.

12. Security Testing

The Master Test Plan (MTP) must state that Security Testing guidance will be included in the Certification & Accreditation plan. When there is no need for security related testing, the MTP must explain the reasoning for exclusion.

Information security is a top priority within Federal Student Aid. Security testing involves using appropriate methods and procedures to validate the implementation of security controls on the application and/or the environment in which the application operates. Test results show the extent to which the controls are correctly implemented, operating as intended, and producing the desired outcome and accurate implementation of the security requirements of the system.

Note: All teams must contact the Virtual Data Center to schedule security scans.

Security Testing includes the following areas:

Table C-1: Vulnerability Scanning

Area	Responsibility
Upgraded or Downgraded Server(s) and/or Application(s)	If a server(s) or application(s) is upgraded or downgraded, a vulnerability scan must be performed A.S.A.P.
New or Upgraded Server(s) and/or Application(s)	Several tools are run for new or upgraded servers and applications. These tools check for vulnerabilities against the Common Vulnerabilities and Exposures (CVE) database, which is maintained by MITRE for US CERT. The tools also check for missed configurations based upon the NIST hardening guidelines.
Vulnerability Scripts	Scripts are created based on the features and settings within a given product.
WEB Servers and WEB Applications	Web Inspect is used for WEB servers and WEB applications. As a double blind product, Shadow Security Scanner and/or Nexus may be used.
Databases	AppDetective is used to conduct scans on databases.
Generic Servers and Applications	Shadow Security Scanner and/or Nexus are used on generic servers and applications.

An example of Security Testing:

Password Cracking

During these tests, testers identify weak passwords. A strong password contains 10 or more characters that include upper case and lower case letters, numbers and special characters.

Examples of negative Security Tests include the following:

- Trying to access a restricted URL by having a user who is logged in but is not authorized to go to the restricted URL type the restricted URL into the browser's URL window.
- Testing of users' privileges to parts of the applications (no access, read-only access, write access, create, read, update, delete, etc.).
- Policies and procedures governing information security controls required for Federal Student Aid may be found on the FSA Security portal website.
- Policies, procedures, templates, documents, forms, training materials and links to internal department websites related to information security may be found on the FSA Security portal website.

Vulnerability Scanning and Penetration Testing policies may be found on the FSA Security portal website.

13. Disaster Recovery of Continuity of Services (CoS)

The goal of disaster recovery testing is to ensure that if a system experiences the worst-case scenario of a failure (hardware or software) at the VDC, the system is able to fully recover and resume normal business operations at a Federal Student Aid designated backup facility.

The *Virtual Data Center (VDC) Continuity of Services Plan* contains the standards for disaster recovery testing. The plan defines the resources, actions, tasks and data needed to recover systems and the infrastructure that supports those systems.

Business Unit Test Leads and Information System Security Officers are responsible for the following items:

- Test Plan
- Test Scripts
- Business Impact Analysis

The Enterprise Testing Team is responsible for the following items:

- Creating the TRR document for Disaster Recovery Testing
- Working with business units to review test plans and test scripts
- Scheduling and leading the TRR meeting
- Working with the FSA VDC Disaster Recovery Project Lead

Note: *Contingency test plans must be created and are a separate document from the disaster recovery plan.*

The Basic Disaster Recovery Test Readiness Review (TRR) Document template may be found in Appendix E. The TRR document may be modified based on the needs of the system being tested.

14. Review and Testing of Documentation

Software documentation and the testing of software documentation is an important step in understanding and accepting software. Many federal organizations have special teams of "document reviewers." Contractors must understand that documentation is just as important as the functional aspects of their software.

Testing of documentation is a necessity, but this process is not complete until all of the necessary documents have been tested and reviewed. Having thorough documentation from installation guides to help files, test plans to test summaries, as well as thorough software documentation, reduces confusion and may reduce software-testing time.

Some important documentation to consider includes:

- Requirements, Business Use Cases, System Use Cases, User Guides, and Detailed Design Documents: Define how functionality will be implemented, and how to use the application.
- **Test Strategy**: Outlines the high-level process for testing the application.
- **Results Summaries**: Identifies the results of each round of testing.
- **Known Issues Document**: Used primarily for Technical Support and identifies issues Development is aware of but has chosen not to correct, potential problem areas for users, and workarounds.
- **Open, Deferred or Pending Defect Summary**: Summarizes the defect ID, the problem area, and title of defects remaining in the defect management system with a status of Open, Deferred or Pending.
- **Fixed Defects**: Lists defects waiting for verification by Quality Control.
- **Installation and Help Instructions**: Aids system administrators in installation, integration, and trouble-shooting installation problems and identifies how to access help files and where links are displayed for testers and users.

These documents all facilitate the Quality Control process and explain, at a high level, the key documents that must be received, installed, or utilized during any project cycle, integration process, or installation process, in order to assess the readiness of a product for release. Although the level of detail in each document may vary by contractor and test organization, each provides evidence of appropriate testing, test management, and project status.

All software related documentation should be written to avoid ambiguity and inconsistency, and to help avoid future defects or miscommunication. The author should make sure all documentation complies with the FSA Style Guide. It is necessary to user spelling and grammar checking devices if possible. Upon completion of the

initial documentation, the preparer should deliver the documentation to at least one reviewer that is familiar with the type of documentation. The reviewer should reassure the document complies with the FSA Style Guide, there are no grammar or spelling errors, and the content is accurate/up-to-date and technically appropriate for the target audience or to complete the task. Procedures that are conveyed in the document should be tested to ensure that they can be executed in a "real world" environment. If the reviewer identifies any errors, suggestions or have comments, they should be provided to the preparer for correction and returned for another review process until satisfactory.

15. Test Automation

While manual testing is routinely used for some tests, automated testing tools are also a popular way to assess and apply faster Regression and Load Testing to validate a website's performance and stability. The contract or statement of work will identify the tools, or types of tools, to be used for testing, including the name of the publisher and the version number to be used.

Test Suites that contain automated test scripts should be separate from test suites that will be executed manually. The manual tests may be outside the range that the automated tests can execute. In addition, separating the two test types is important to maintain discrete metrics. For example, the number of pass and fail attempts using automated test scripts is generally much larger and should be kept separate from the number of pass and fail attempts that occurred during manual testing.

Test Automation includes the use of software testing tools to control the execution of tests, to compare actual outcomes to predicted outcomes, to create test preconditions, to create test data, and to perform other test control and test reporting functions. Creating automated tests frequently involves automating an existing manual process.

Advantages of Test Automation

There are several advantages to test automation:

- Low Time to market and reduced testing time because scripts only need to be created or modified when changes occur in the related requirements
- Improved Testing productivity because the Testing Team doesn't need to repeat the same test suites manually
- Improved Product Quality

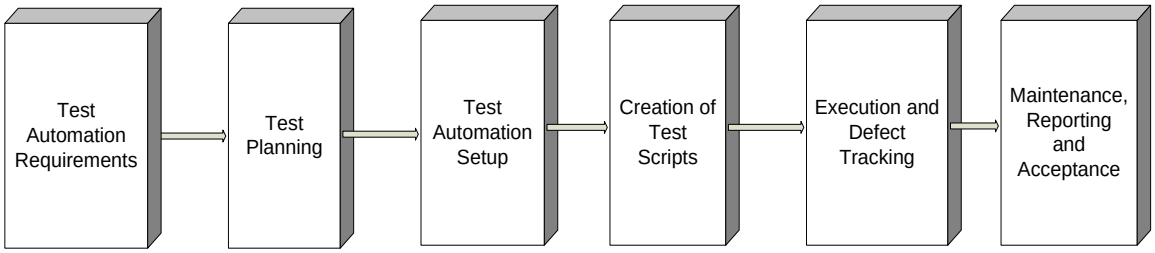
Types of Tools Used

Federal Student Aid requires the use of the Defect Management tools for use in all software engineering/testing efforts. Mercury Load Runner has also been used for performance testing at Federal Student Aid. Rational Functional Tester is being used for functional test automation. FSA must approve any additional tool proposed by the contractor. Some other types of tools include:

- **Automated Regression Testing Tool**: Used primarily as a capture-playback testing tool that records test scenarios, plays them back, and generates reports comparing test results with reference results.
- These tools can also be used to extract data for input to test suites.

Test Automation Life Cycle

The Test Automation Life Cycle Process demonstrates the life cycle:



Test Automation Life Cycle Process

Test Automation Requirements

Before testing can be automated, the following conditions must exist:

- Requirement/Functional Specification documents must exist and be approved
- Design Specification documents (use cases)
- Test Traceability Matrix for identifying Test Coverage
- Functional/Non-Functional and test data requirements

Test Planning

If test automation is to be performed, the following test planning issues must be addressed:

- Automated Software Testing: Scope must be defined to identify the area and percentage of automation
- Automation should be attempted first only on a subset of a system
- Identification, evaluation and procurement of the automated testing tool
- Creation of an automated testing Framework

Test Automation Environment Setup

To create an automated testing environment, the tools required for automation testing must be installed on the appropriate servers/machines.

Creation of Test Scripts

Once the environment has been created, testing with an automated testing tool requires:

- Preparation of test scripts
- Unit testing of the test scripts

- Integration Testing of the test scripts

Execution and Defect Tracking

Following the design of tests, execution of the tests and defect management requires:

- Execution of the automated test suite
- Analysis of the Test Results
- Defects reporting and tracking

Maintenance, Reporting and Acceptance

Following testing, defect management, and defect resolution, results should be reported as follows:

- Automated Software Test Results and summary reports
- Test Reports consisting of metrics
- Getting Acceptance

An example of Test Automation is Regression testing using a testing tool. These tools can create customizable, proprietary scripts by recording User Interface interactions in a test case during an interface testing session. The scripts can be played back to repeat these User Interface interactions, thereby eliminating the need for the tester to manually repeat User Interface interactions.

16. Application Programming Interface (API) Testing

An Application Programming Interface (API) is specialized software providing functionality that is executable by other software. APIs greatly reduce the need for individual applications to perform fundamentally generic functions like some printing and security functions, and database access.

API Testing is performed using test frameworks for Unit Testing, Integration Testing, and Regression Testing performed prior to System Testing. During UAT, the business users test the application functionality that incorporates the API calls.

An example of API testing is Reusable Common Services (RCS) testing within Federal Student Aid. RCS components are reusable components that application groups can use to lessen programming effort, enforce standard modes of business, and implement best practices.

17. White Box Testing

White Box Testing strategy helps to ensure the internal logic and structure of the software's code. White Box Testing is also known as Glass, Structural, Open Box, or Clear Box testing. Tests written using a White Box Testing strategy include coverage of the statements, decisions, conditions, and internal logic of the code.

White Box Testing Types

Statement Coverage

Statement coverage helps ensure that all the statements in the code execute correctly without any unintended side effect. While statement coverage tests the execution of each statement, it does not control structures such as Boolean expressions (for example, determining if an "if" or "while" statement provided the correct result).

Decision Coverage

Decision Coverage is used where Boolean expressions are tested (for example in testing "if" or "while" statements). This method should be used to test both true and false conditions.

Condition Coverage

Condition coverage is used when code contains Boolean expressions that include branches. For example:

Example 1 simplified

If (number of apples > 5 *OR* (number of oranges > 2 AND number of pears > 5)) then [do this] else [do that].

Decision coverage could ignore execution of Function1 whereas the condition coverage reports the true or false outcome of each Boolean sub-expression, measuring each one of them independently.

Example 1 as coded

```
If (expression1 || (expression2 && Function1() ))
        Statement 1
Else
        Statement2
```

An example of White Box Testing is stepping through the code, line by line, to determine whether the code module responsible for login requests properly authenticates valid requests and rejects invalid requests.

18. Black Box Testing

Black Box Testing tests the functionality of a program or application against its specification without knowledge of or direct review of the underlying code or internal logic.

Black Box Testing Types

- Equivalence Partitioning

  Input data and output results often fall into different classes where all members of the class are related. Each member of this class is called an Equivalence Partition. The process of determining these input classes is defined as Equivalence Partitioning.

- Boundary Value Analysis

  Boundary value analysis is a technique that consists of developing test suites and data that focus on the input and output boundaries of a given function. Boundary values include maximum, minimum, just inside/outside boundaries, typical values, and error values. The belief is that if a system works correctly for these special values, the system will work correctly for all other values inside and outside the boundaries.

- Error Guessing

  Error guessing involves developing an itemized list of the errors expected to occur in a particular area of the system, then designing a set of test cases to check for these expected errors. Error guessing is more of a testing "art" than "science" but can be very effective when implemented by a tester familiar with the history of the system.

Below is an example of Black Box Testing:

Testing input values for a student's Social Security Number on the application's registration page to determine whether the characters defined as acceptable are accepted, the characters defined as unacceptable are not accepted, and the appropriate error message is returned.

19. Parallel Testing

Parallel Testing involves operating a new application and an existing application in parallel for a specified period. Each system receives the same data input and performs the same or equivalent workflow with both applications producing the same results.

Note: Parallel Testing is effective when there have been minimal changes between the new application and the existing application. The figure on the following page illustrates the parallel testing process:

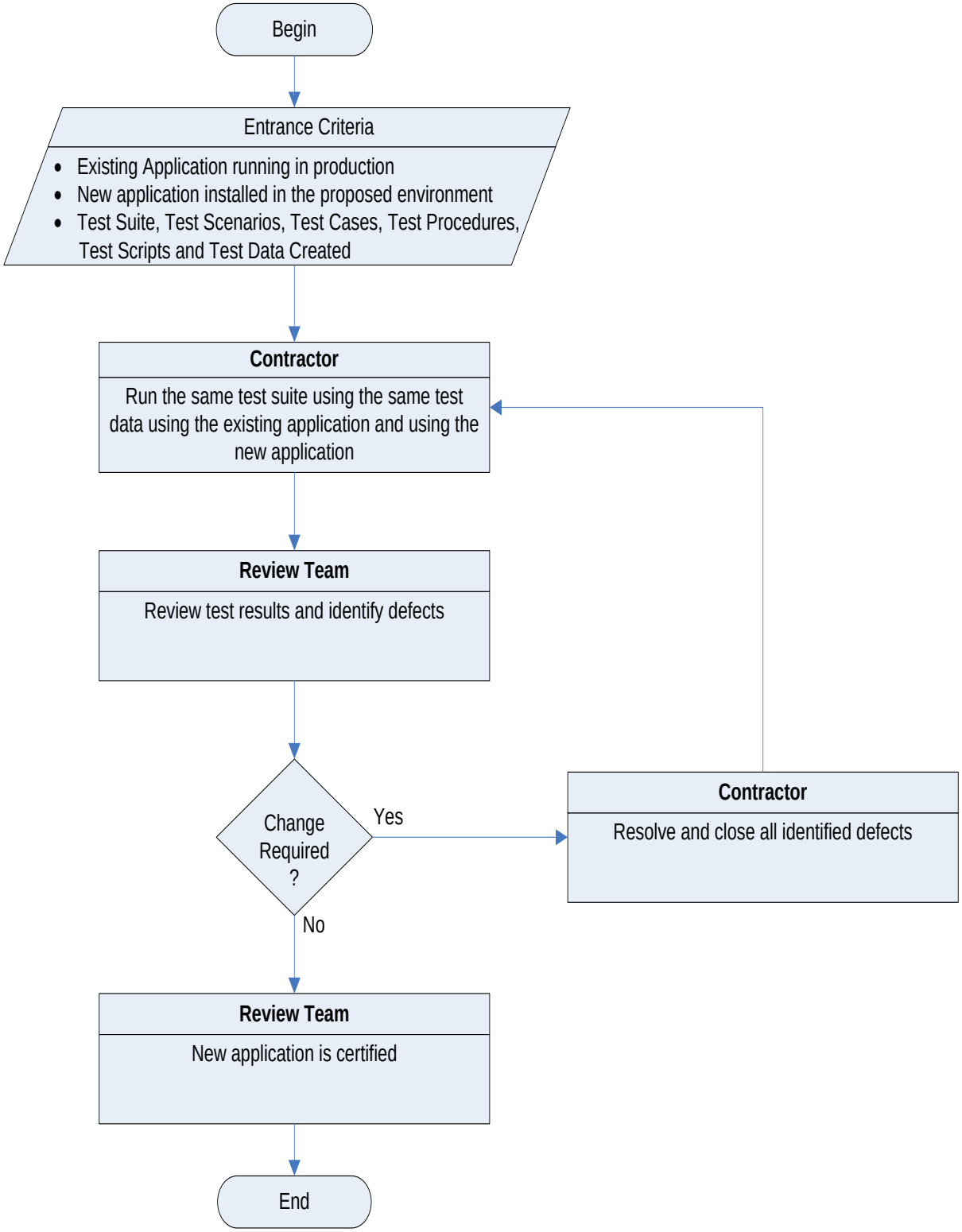


Figure 12: Paralell Testing Process

20. Regression Testing

Testers use Regression Testing to evaluate whether previously tested functions or components still function correctly. Regression Testing is important during application development and maintenance of an existing application. This testing must ensure that code changes have not inadvertently affected functionality of the application.

The most effective Regression Testing involves developing test suites that test common functionality and can be executed every time a new version of the program is built; it is also important that existing functionality is thoroughly tested when new fixes are applied to the software. A thorough Regression Test suite can validate that a change to the program has not inadvertently changed any other application functionality.

An example of Regression Testing:
When a SSN entry defect occurred in production, the application was updated to resolve the SSN entry defect. The entire SSN entry page was retested to ensure that no other functionality on that page was affected in the process.

- A brand new application was created to create a new student database. The System Test Team found a defect while entering the student's last name. All other functionality for the new application worked according to the design. Developers corrected the student last name defect. The System Tester tested the application to ensure that the student last name defect was corrected and that all other functionality for the application still worked according to the design.

21. Agile Testing

CONTENTS OF AGILE TESTING

The following table shows the contents for this section of the appendix.

Contents of Agile Testing

Practice Type	Title	Brief Description
General Agile Testing	Levels of Agile Testing	Different categories of testing for Agile
General Agile Testing	Agile Key Concepts	An explanation of important Agile concepts
FSA Specific Agile Testing	Key Testing Activities in the Agile Lifecycle	Testing activities in the LMM lifecycle for an Agile project
FSA Specific Agile Testing	FSA Definition of "Done"	The definition of "done" usage at FSA
FSA Specific Agile Testing	Impact on Test Planning	Test planning activities for an Agile project
FSA Specific Agile Testing	Impact on Test Phases	Testing activities by phase for an Agile project
FSA Specific Agile Testing	Impact on General Test Practices	Other practices for an Agile project, such as defect management, documentation, and testing with limited time

Practice Type	Title	Brief Description
FSA Specific Agile Testing	Roles and Responsibilities	Who is involved in Agile Testing and what they do
FSA Specific Agile Testing	Implementation Document	Steps that testers do on an Agile project
FSA Specific Agile Testing	Sprint	Scrum structures product development in cycles of work called Sprints, iterations of work which are typically 1-4 weeks in length. The goal is to create a potentially releasable product Increment within the defined Sprint. Sprints consist of the Sprint Planning, Daily Scrums, the development work, the Sprint Review, and the Sprint Retrospective.
FSA Specific Agile Testing	Sprint Review	At the end of the Sprint, there is a Sprint Review where the team and stakeholders inspect what was done during the Sprint. The Sprint Review is an inspect and adapt activity for the product. The review is attended by the Product Owner, The Scrum Team, and Scrum Master; customers and other stakeholders may attend as well. It is a time for the Product Owner and key stakeholders to learn what is going on with product and the team. The review also includes a demo of what the Team built during the Sprint, but the demo is not the focus.
FSA Specific Agile Testing	Sprint Retrospective	Following the Sprint Review, the team gets together for the Sprint Retrospective to identify and discuss what is working and what is not working. It is an opportunity for the Scrum Team to evaluate itself and create a plan for improvements to be implemented during the next Sprint.
FSA Specific Agile Testing	The Product Owner	The Product Owner is responsible for maximizing the value of the product and the work of the team by identifying product features, translating these into a prioritized feature list, and deciding which should be at the top of the list for each sprint. The Product Owner is one person, not a committee. The Product Owner may represent the desires of a committee in the Product Backlog, but anyone wanting to change a Product Backlog item's priority must address the Product Owner. In some cases, the Product Owner and the customer are the same person.
FSA Specific Agile Testing	The ScrumTeam	The Scrum Team consists of a Product Owner, the Development Team, and a Scrum Master. The Development Team in Scrum might include developers, interface designers, business analyst and testers. The team is cross-functional and includes all the expertise necessary to deliver the product.

GENERAL AGILE TESTING PRACTICES

Agile provides a continuous stream of value to customers, or Application Owners. The objective is to produce the maximum value at the minimum cost. The Agile Manifesto lists a set of values that drive all Agile methodologies:

We have come to value:
Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan
That is, while there is value in the items on the right, we value the items on the left more.

Note: *The reference to 'contract negotiation' in the manifesto is not related to a contract between the government and a contractor. In an Agile project, since requirements are not fully collected at the beginning of the development cycle, there is continuous user/customer involvement and negotiated agreement with all team members.*

Since Development and Testing are tightly linked in Agile, there is no separate approach to Development versus Testing. They are one and the same and require continuous participation from the Application Owner.

The testing practices that follow the Agile Manifesto treat development as the customer of testing. There are key strategies to help implement Agile:

1. **Communicate** - Agile development depends on disciplined communication around requirements and testing. Agile teams hold daily meetings to understand and communicate progress. Identifying tools to be used in the project should be made clear and adhered to. Understanding defects found, setting the severity, prioritizing them, and communicating them to the team are essential. Finding time to discuss lessons learned early helps to avoid repeating mistakes and delays.
2. **Commit** - Continuous diligence for excellent delivery and good design should be a commitment by the team including the business users, developers, and testers.
3. **Automate** - It is important to note that with any of the Agile methodologies, automation is typically a requirement of the selected approach given that the main focus of these methodologies is a quick, iterative process. There are many aspects of the approaches that need to be tracked and monitored, and a manual process introduces more risk. FSA is still working towards this strategy.
4. **Adapt** - There will be quick deployments and changes to testing patterns and team members should be able to adapt quickly.

In Agile, testing provides information to the team, so they can make informed decisions. Regression testing is done to ensure that resolution of defects does not affect the functionality of features already tested. Exploratory testing is encouraged to find out

how the implemented functions actually work, and how those functions handle difficult and easy scenarios.[2]

Levels of Agile Testing

Agile testing happens at two main levels. All parties involved should have a clear understanding of the Agile process and what is expected.

Developer Tests define whether the system does what the developers expect ("building the code right"). Developer Tests:

- Include Unit Tests, which take place in conjunction with the development of each component
- Include Integration Tests
- Should be automated, whenever possible
- Enable developers to verify that, after a change, the system still works the same way

Acceptance Tests indicate whether the system does what the customer expects ("building the right code"). The acceptance tests are written with collaboration of the Application Owner and they include a testable statement of what acceptance means. Acceptance Tests:

- Define business requirements and are also called customer tests or functional tests
- Serve as executable requirements documents, however they specify business requirements, not application design
- Can be understood by non-technical people
- Can include Integration Tests, Smoke Tests, Performance Tests, Load Tests
- When available, can be run automatically at any time by anyone
- Should be independent of implementation

Note: *The standard industry meaning of "Integration Tests" is similar to FSA's use of "System Tests".*

Agile Key Concepts

The following table introduces key concepts for Agile projects, which provides context for information provided in the remainder of this "Agile Testing" appendix.

[2] Crispin and Gregory note: "Some testing approaches, such as exploratory testing, are inherently agile…. Testing an application with a plan to learn about it as you go, and letting that information guide your testing, is in line with valuing working software and responding to change." (Agile Testing, p. 7).

Agile Key Concepts

Concept	Explanation
Test Automation	Agile methods emphasize test automation in order to efficiently regression test and deliver in a timely manner. When automated tests are used, the tests require disciplined maintenance as they must be able to be run automatically by anyone. The entire Agile development approach requires a tight build/integration process and rigorous management of the sprint.
Definition of "Done"	Agile projects use a definition of "done", which is a description of the criteria that every increment must fulfill in order for the increment to be considered complete. The "done" criteria typically require that requirements (e.g., user stories) be transformed into working software that is thoroughly tested and adequately documented.[3]
User Story	User stories are the primary requirements type used in Agile projects. A user story is a short, simple description of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system. During the sprint planning meeting, the team gives each user story a priority and estimate. The estimate includes the effort for developing and testing the requirement. User stories are often elaborated into other requirements artifacts during design, coding, and even testing.
Epic	An epic is a high level user story. The team decomposes epics into lower level user stories.
Product Backlog	The product backlog is an ordered list of everything that might be needed in the product, and the single source of requirements for any changes to be made to the product. The product backlog consists mostly of user stories decomposed from epics.[4]
Sprint Backlog	The sprint backlog is the set of product backlog items selected for the sprint plus a plan for delivering the product increment and realizing the sprint goal. The team determines the contents of the sprint backlog during the Sprint Planning Meeting. Most of the work done by the Development Team and the Test Team traces back to the sprint backlog.

[3] See Pichler, Agile Product Management with Scrum, p. 99.

[4] Roman Pichler notes: "Scrum does not mandate how product backlog items are described, but I prefer to work with user stories (Cohn 2004)" and includes epics as a type of user story. But he adds, "If you do use user stories, you should not feel obligated to describe every single product backlog item as a user story"; and says, other helpful artifacts "do not replace the product backlog but instead should elaborate and explain its content." (Agile Product Management with Scrum, p. 53).

Concept	Explanation
Daily Scrum Meeting	The daily Scrum is a short time-boxed meeting for the entire team to synchronize activities, uncover impediments, and plan for the next 24 hours. The team uses the daily Scrum meeting to assess how progress is trending toward completing the work in the sprint backlog. (Note that the phrase "Daily Scrum" is specific to the Scrum implementation of Agile.)
Scrum Master	The Scrum Master is the team's coach. A Scrum Master differs from a traditional project manager in many key ways, including that the Scrum Master does not provide day-to-day direction to the team and does not assign tasks to individuals. (Note that the phrase "Scrum Master" is specific to the Scrum implementation of Agile.)

FSA AGILE TESTING PRACTICES

For Agile projects, Federal Student Aid has used the Scrum framework where sprint planning identifies the set of functions and features to be developed and tested. Still, refinements in sprint planning, managing changing requirements, setting up of the test environment, prioritizing remediation of defects, and managing backlogs are needed and will be improved.

Federal Student Aid may perform the testing itself or a test contractor may perform the test.

Key Testing Activities in the Agile Lifecycle

For Agile projects, Agile testing activities fit into the FSA lifecycle as shown in the figure below. The figure emphasizes iteration, and the descriptions of the activities emphasize collaboration.

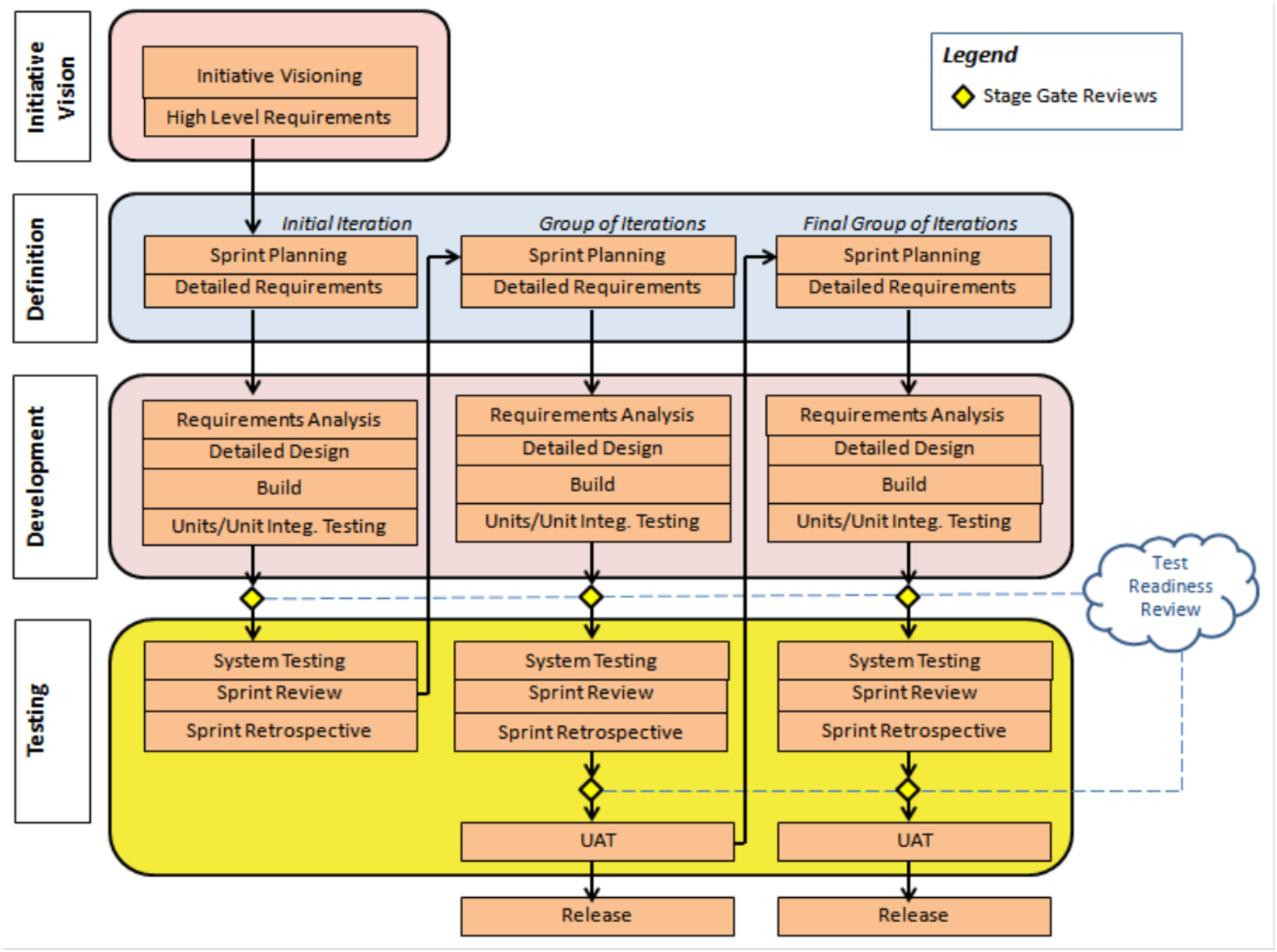


Figure 13: Key Testing Activities in the Agile Lifecycle

Initiative Vision stage

1. **Initiative Visioning:** Initiative Visioning activities on Agile projects are similar to non-Agile projects.
2. **High Level Requirements:** The Business Analysts collaborate with the customer to create high-level requirements.

Definition stage and Development stage

3. **Sprint Planning:** The entire team (including the Test Team) participates in the sprint planning meeting at the beginning of a sprint. The team identifies the requirements (user stories) to be implemented during the sprint, based on the team's productivity and the definition of "done". The Development Team and Test Team work together to determine how much work they can responsibly take on. The Test Team therefore helps ensure sufficient time is available for testing by influencing which requirements are added to the sprint backlog.
4. **Detailed Requirements:** The Development Team, Business Analysts, and Test Team work together with the Application Owner to refine requirements for a sprint. These requirements include sufficiently developed user stories, documented discussions, and linked test cases.
5. **Requirements Analysis:** The entire team analyzes the detailed requirements in order to create the design.
6. **Detailed Design:** The entire team uses the requirements artifacts to develop a Detailed Design.
7. **Build:** The Development Team uses the refined requirements and the design to build the components that make up the sprint.
8. **Unit Testing/Unit Integration Testing:** The Development Team uses the refined requirements to unit test their components. They should unit test in conjunction with building the components. The Development Team performs Unit Integration Testing when they integrate with previously completed components as a part of Unit Testing.

Testing stage

9. **System Test Readiness Review:** The team conducts a Test Readiness Review after Unit Testing/Unit Integration Testing for all components that have been completed. The Test Readiness Review ensures that the team is ready to begin System Testing.
10. **System Testing:** The Test Team uses user stories and other requirements to update System Test Plans, and to write System Test Scenarios and System Test Cases in parallel with the Development Team's component development activities. After completing the System Test Readiness Review, the Test Team executes the system tests for the sprint, and appropriate regression testing for functionality from previous sprints. System Testing should include a mix of manual and automated tests (if available).

11. **Sprint Review:** The entire team holds a Sprint Review at the end of each sprint to inspect the increment. The team and Application Owner reach agreement on what items from the sprint backlog are "done" and not "done". (The items not "done" remain in the product backlog to be addressed in a future sprint.) The Development Team demonstrates the work performed and answers questions about the product. This is an informal meeting, and the presentation of the increment is intended to elicit feedback and foster collaboration.[5]
12. **UAT Readiness Review:** When enough functionality is "done" for the customer to see the value of the development, the team conducts a UAT Readiness Review. The Test Readiness Review ensures that the team is ready to begin UAT for all sprints that have completed. The Project Manager decides how much functionality is needed to demonstrate value to the customer.
13. **UAT:** UAT proceeds with the involvement of the Test Team, Application Owners, and specific end users as appropriate. UAT confirms that the application meets the quality and acceptance criteria. When UAT completes, the team can begin the next iteration.
14. **Release:** Once the application completes UAT, the team releases the application to the customer. The release implements a complete set of requirements and provides useful functionality for the customer.

FSA Definition of "Done"

Before the first sprint, the entire team must agree on a definition of "done" for the project. The team uses this definition for each project sprint.

As shown in **Figure 13** above, sprints consist of Sprint Planning through Sprint Review, and include System Testing. Therefore, the definition of "done" should include System Testing. Using this definition, programmers never get ahead of testers, because a story is not "done" until it has been tested.[6]

Impact on Test Planning

Test planning must conform to the procedures in section 2 of this document. In particular a MTP is required as specified in **Section 3.1** of this document. In addition, one or more Phase Level Test Plans (PLTP) may be required as specified in **Section 3.1** and **Section 3.3.2** of this document.

Special MTP and PLTP considerations for Agile projects are noted in MTP Template Instructions in **Appendix E-**0 of this document, and PLTP Template Instructions in **Appendix E-**3 of this document.

The following items should be noted:

- The Test Team is responsible for creating and filling in the MTP and, if applicable, the PLTPs with feedback from the Application Owner.
- The Test Team is responsible for creating the test plans during the first sprint, and updating them during subsequent sprints, when needed.

[5] See Schwaber and Sutherland, "The Scrum Guide", p. 11.
[6] See Crispin and Gregory, Agile Testing, p. 12.

- Subject matter experts (e.g., Enterprise Testing Team or Extended Integrated Project Team members) and the Project Manager must approve the test plans during the initial sprint; but no subsequent approval is required.
- The use of Change Requests to update the MTP/PLTP is optional, but must follow decisions documented in the Change Management Plan.
- The Test Team must participate in Sprint Planning to influence which requirements are added to the sprint backlog, thereby helping ensure that sufficient time is available for testing.

Impact on Test Phases

Unit Testing/Unit Integration Testing

Unit Testing and Unit Integration Testing activities on Agile projects are similar to non-Agile projects, as described in **Section 4.2** of this document.

The following items should be noted:

- The Development Team performs Unit Testing/Unit Integration Testing in conjunction with building the components.
- Automated testing is recommended as a tool to be leveraged when available.
- Continuous product integration through daily builds is recommended.
- The entire Agile development approach requires a tight build/integration process and rigorous management of the sprint.

Integration Testing

Integration Testing is not applicable for Agile projects.

System Testing

System Testing activities on Agile projects are similar to non-Agile projects, as described in **Section 4.4** of this document.

The following items should be noted:

- The Test Team uses the user stories and other requirements to update System Test Plans, and to write System Test Scenarios and System Test Cases in parallel with the Development Team's component development activities.
- In order to develop System Test Scenarios and System Test Cases, the Test Team helps refine requirements by working with the customer and collaborating with the Development Team.
- System Testing should include a mix of manual and automated tests (if available).
- When defects are found, they are either resolved during the sprint or placed in the product backlog (see the "Defect Management during System Testing and UAT" section below).

UAT

UAT activities on Agile projects are similar to non-Agile projects, as described in **Section 4.5** of this document.

The following items should be noted:

- More information and participation is required from the Application Owners than in a more traditional development approach.
- The Test Lead/Test Team must collaborate with the Application Owner and end users to plan the timing of user acceptance tests (relative to the iteration or group of iterations).
- Creation of user acceptance tests must be collaborative. The Test Team works with the Application Owner to formalize acceptance criteria into scenarios and user acceptance tests.
- UAT proceeds with the involvement of the Test Team, Application Owners, and specific end users as appropriate.
- When defects are found, they are either resolved before the release or placed in the product backlog (see the "Defect Management during System Testing and UAT" section below).

Impact on General Test Practices

Requirements Artifacts Used

On an Agile project, only the requirements needed to complete the coding and testing for a sprint are completed. Those requirements are completed "just-in-time", instead of at the beginning of the project. The requirements artifacts may be in a "draft" state when being used by the Test Team, since the details are being refined during the sprint.

The Test Team uses any available detail-level requirements artifacts to write test scenarios, test cases, and test scripts. These artifacts include:

- User stories
- Use cases
- Business rules
- Reporting requirements
- External data flows
- UI specifications
- DRD supplemental requirements

Defect Management Using Defect Management

In Federal Student Aid, the Defect Management is used for defect management. The Development Team maintains stories and tasks. Defects are input by the Testing Team when they are found, and screen prints, comments, and descriptions are attached to each defect and associated with the stories and/or the requirements. Severity is

assigned as well and the testers work with the Application Owner and the developer to assign the priority for remediation.

When a defect is created, the tester who opens the defect may own the defect until the screenshots have been attached and the description of the defect are clear and understandable to the developer. Once this is done, the tester may change the owner of the defect to the assigned developer. The developers may also take the responsibility to own a defect.

The developers update these defects to “in-progress”, then to “resolved”. Afterwards, the testers re-execute pertinent test cases and update the successfully-remediated defects to “verified”. Otherwise, the status of a failed re-test is set to “re-opened”.

Since the statuses of defects are maintained, metrics are maintained to report in project status meetings. They are also saved to be used in the testing summary portion of the Production Readiness Review presentation of each project.

Defect Management during System Testing and UAT

When the Test Team finds a defect during System Testing, the tester records the defect. The Application Owner prioritizes the defects that were found with assistance from the Test Team. For each defect, either:

- The Development Team investigates and resolves the defect during the sprint.
- The defect is deferred and the Project Manager places the defect in the product backlog (to be addressed in a future sprint).

When a tester finds a defect during UAT, the defect is recorded. The Application Owner prioritizes the defects that were found with assistance from the Test Team. For each defect, either:

- The Development Team investigates and resolves the defect before the release.
- The feature remains in the product with a known issue, and the Project Manager places the defect in the product backlog (to be addressed in a future sprint).
- The feature is “backed out” of the release, the associated work item is considered not “done”, and the item remains in the product backlog (to be addressed in a future sprint).

Documentation

For Agile software projects, it should be kept in mind that one of the Agile values is “Working software over comprehensive documentation”, which does not mean “no” documentation. Here are some tips for documentation:

- On Agile projects, the entire team has a role to play in developing documentation.
- The MTP and PLTP documents drive testing activity. Completing these test plans during the initial sprint should be a top priority.

- The product backlog should focus on the minimum functionality necessary to bring the product to life.[7] It consists primarily of user stories and epics. It does not need to contain every requirement document. The backlog items can be thought of as a reminder to have a conversation with the customer.
- The team updates requirements and testing documents over the course of a sprint.
- For a given sprint, the Detailed Requirements document cannot be finalized until completion of System Testing.
  Note: See the Enterprise Requirements Management Standards and Tailoring Guidance document for more information.
- The Business Analyst, Development Team, and Test Team all contribute to the requirements Traceability Matrix.
- Put documentation in a document repository. Check documents in often, even if they are not finalized.
- Within many systems, there are a few key tricky or subtle elements or themes that you would not want to lose. Find those, highlight them, and write a short "technical memo" page for each on the project website.
- Update test documentation as needed; updates may be done even at the end of the testing phase, prior to the final Production Readiness Review.

Coordinating Sprint and System Testing/UAT Activities

On Agile projects, the Test Team and the Development team work in unison: they start the sprint together, participate in the sprint planning meeting together, work development and test activities in parallel, participate in the sprint review together, and finish the sprint together. Because of the collaborative nature of Agile projects, the Development Team does not complete one sprint and start another without the Test Team. The fact that the FSA definition of "done" includes System Testing helps ensure that the Development Team and Test Team are working together on the same sprint and that programmers do not get ahead of testers.

If development and testing are proceeding at different paces, then situations may arise when the Development Team is ready to move on to the next iteration, but the Test Team is still conducting System Testing. In these situations, there are two possible strategies that can be followed:

- Resynchronize the development and test efforts by adjusting staff levels on the team (e.g., by adding new members to the Test Team). When practical, reallocating members from the Development Team to the Test Team helps preserve project knowledge continuity.
- Part of the Test Team starts the next sprint while some Test Team members complete System Testing for the current sprint. All team members (including those still supporting System Testing) participate in the sprint planning meeting

[7] Pichler, Agile Product Management with Scrum, p. 52.

and daily Scrum meetings for the next sprint. This enables the Test Team members performing System Testing to efficiently re-engage with the rest of the team after System Testing completes. The sprint review for the current sprint is deferred until System Testing completes. All team members participate in the sprint review.

Due to logistical considerations, situations may arise when the Development Team is ready to move on to the next iteration, but the Test Team is still working with the Application Owner and end users on UAT for a release. In these situations, there are two possible strategies that can be followed[8]:

- The Development Team and part of the Test Team start the next sprint while the remaining Test Team members provide UAT support for the existing release.
- The Development Team and part of the Test Team work on preparations for the next sprint ("Iteration 0") while the remaining Test Team members provide UAT support for the existing release. During Iteration 0, the team may perform refactoring or experiments related to new functionality in order to prepare for the next sprint. This prevents the team members supporting UAT from being left behind.

With either strategy, all team members (including those still supporting UAT) participate in all sprint planning meetings, daily Scrum meetings, and sprint reviews. This enables the team members supporting UAT to efficiently re-engage with the rest of the team after UAT completes.

Testing in a Time-Boxed Sprint

Through the Test Team's participation in the daily Scrum meetings, the team recognizes when it is falling behind schedule. To adjust for this, the team may drop one or more user stories from the sprint backlog in order to complete its work within the time box.[9] (These user stories return to the product backlog to be addressed in a future sprint.)

Testing in a Compressed Timeline

The Test Team can apply various techniques when time for testing is extremely limited. Best practices include:

- **Risk-Based Testing:** Start by testing the "happy paths". After the happy paths have been tested, focus on the highest risk scenarios—cases that not only have a bad outcome but also have a good possibility of happening. Specify tests to cover potentially risky outcomes of an action.[10] (See **Appendix D-22** "Risk Based Testing" for details.)

- **Exploratory Testing:** Exploratory testing is interactive testing that combines test design with test execution and focuses on learning about the application. As the tester performs tests, the tester learns more about the application and uses that information to help design new tests. This knowledge reveals areas of the

[8] See Crispin and Gregory, Agile Testing, p. 467-468.
[9] See Crispin and Gregory, Agile Testing, p. 463.
[10] See Crispin and Gregory, Agile Testing, p. 147-148.

product that need additional testing. (Note that exploratory testing is substantially different than "ad hoc testing" because exploratory testing applies discipline and critical thinking to leverage knowledge gained during the testing cycle to focus on the most effective an important tests.)[11]

Roles and Responsibilities

The following table shows the roles of participants in Agile testing.

Roles and Responsibilities in Agile Testing

Title	Responsibilities
Project Manager	• Approve the MTP and PLTP documents • Determine if unresolved defects should be fixed in the current sprint, or placed in the product backlog • Determine when to release (based on team input and progress) • Oversee the development/testing effort. This includes holding sprint planning meetings, daily Scrum meetings, and sprint reviews. • Understand and communicate how reprioritization may affect the development/testing effort • Participate in Test Readiness Reviews
Application Owner	• Work with the team to refine requirements • Work with Test Lead to setup User Acceptance Testing • Work with Test Lead to schedule User Acceptance Testing and schedule the testing times for the UAT testers • Support the end users in formalizing user acceptance tests with the Test Team • Work with end users to execute User Acceptance Testing • Participate in Test Readiness Reviews
Development Team	• Participate in sprint planning meetings, daily Scrum meetings, and sprint reviews • Help refine requirements for a sprint • Update the requirements Traceability Matrix based on design • Build code • Write Unit Test Cases • Execute Unit Testing • Research and fix defects found in Unit Testing • Help the Test Team connect the testing framework to the actual system during System Testing • Research and fix defects found in System Testing • Participate in Test Readiness Reviews

[11] See Crispin and Gregory, Agile Testing, p. 195-202, 494-495.

Title	Responsibilities
FSA Test Manager/Test Leads	• Participate in sprint planning meetings, and sprint reviews • Participate in daily Scrum meetings, providing the status of testing (including details on any blocking defects or test environment issues) • Ensure all Entrance Criteria are satisfied prior to System Testing • Review all testing artifacts, including the defect log and test results • Lead Test Readiness Reviews • Work with Application Owner to setup User Acceptance Testing • Work with the Application Owner to schedule User Acceptance Testing and schedule the testing times for the UAT testers • Support the Test Team and end users in formalizing user acceptance tests
Test Team	• Develop MTP and PLTPs, if applicable • Participate in sprint planning meetings, daily Scrum meetings, and sprint reviews • Work with Application Owners and Business Analysts to refine requirements for a sprint • Develop Test Scenarios accordingly to use as Test Cases and to verify and validate user stories • Write System Test Cases • Create the System Test Framework, including Test Scripts and Procedures • Execute System Testing, and record defects • Update the requirements Traceability Matrix based on test cases • Prepare for and participate in Test Readiness Reviews • Work with the end users to formalize user acceptance tests • Execute User Acceptance Testing, if needed
Business Analyst	• Collaborate with customers to create high-level requirements • Participate in sprint planning meetings, daily Scrum meetings, and sprint reviews • Write and refine requirements for a sprint • Complete requirements documentation • Help define or review test scenarios and test cases • Create the requirements Traceability Matrix • May review the requirements Traceability Matrix that the Development Team and Test Team has updated • Participate in Test Readiness Reviews

Title	Responsibilities
Scrum Master	• Help determine what goes into product backlog • Lead daily Scrum meetings • Adjust product backlog as needed based on the findings of System Testing/User Acceptance Testing

Note: *For information about how these roles interact to produce the requirements, see the FSA Enterprise Requirements Management Standards and Tailoring Guidance document.*

Implementation Document

The following table summarizes test activities that should be performed for an Agile project.

Agile Document for Testers

Item #	Agile Document for Testers
1	Communicate with the Development Team • Participate in daily Scrum meetings • Help Development Team and Business Analyst to refine user stories and other requirements
2	Contribute to user story (component) estimates • Participate in sprint planning meeting • Consider historical productivity and user story complexity • Make sure system test estimates are included in user story estimates
3	Develop Test Plans • Develop MTP during first sprint by following template in Appendix E-0 • Develop PLTPs (if required) during first sprint by following template in Appendix E-3 • Have Project Manager and SMEs approve test plans during first sprint • Update MTP and PLTPs during subsequent sprints, as needed
4	Participate in System Test Readiness Review

Item #	Agile Document for Testers
5	Write System Test Scenarios and Test Cases • Leverage user stories and other requirements • Use examples provided by the Application Owner and end users • Start by testing the “happy paths” • Focus on the highest risk scenarios—cases having a bad outcome and a good possibility of happening • Build up test case complexity (e.g., enforcement of required/optional fields, edits/valid values for individual fields, edits/valid values for combinations of interdependent fields, boundary conditions/edge cases) • Identify regression tests needed for System Testing
6	Perform System Testing • Automated testing is recommended as a tool to be leveraged when available • Build System Testing framework, including Test Scripts and Procedures • Execute System Tests • Perform exploratory testing when possible • Record System Test Results • Inform Project Manager of defects, and include sufficient documentation • Assist Application Owner in prioritizing defects • Work with Development Team to identify defect causes • Test defect fixes
7	Participate in Sprint Review • Contribute ideas about what worked well, and what didn't • Constructively submit ideas to help team improve
8	Participate in User Acceptance Testing Readiness Review
9	Assist with UAT • Consider whether System Testing artifacts can be reused • Collaborate with the Application Owner and end users to plan the timing of user acceptance tests (relative to the iteration or group of iterations) • Collaborate with the Application Owner and end users to formalize UAT acceptance criteria, scenarios, and tests • Execute user acceptance tests, as needed

22. Risk Based Testing

Overall, testing is the means used in software development to reduce risks associated with an application or system. The goal of testing is to identify many of the problems before they get to production, thereby reducing the system's risk. Unfortunately, testing alone cannot find all of the defects, and with the rapid pace of application development, testing has become a challenging proposition.

Trying to meet even tighter deadlines while still delivering applications that meet requirements is the greatest challenge for testers. Formulating answers to questions like “What should we test?” and “How long do we test?” requires different strategies in fast-paced environments.

- Does the application meet our quality expectations?
- Is the application ready for users?
- What can we expect when 2,000 people hit the site?
- What are we risking if we release now?

Risk is the possibility of suffering harm or loss. In software testing, we think of risk on three dimensions:

- A way the program could fail
- How likely it is that the program could fail in that way
- What the consequences of that failure could be

Risk analysis assesses damage during use and usage frequency, and determines probability of failure by looking at defect introduction. Testing is always a sample. You can never test everything, and you can always find more to test. Thus, you will always need to make decisions about what to test and what not to test. The general goal is to find the worst defects first, the ones that NEED TO BE FIXED BEFORE RELEASE, and to find as many such defects as possible.

This means the defects must be important. The problem with most systematic test methods, like white box testing, or black box methods like equivalence partitioning, boundary value analysis or cause-effect graphing, is that they generate too many test suites; some of which are less important. A way to lessen the test load is to find the most important functional areas and application properties. Finding as many defects as possible becomes more likely when more testing occurs in known problematic or error prone areas of the application. This means you need to know where to expect more defects.

At the Test Suite execution level, the Test Suites are prioritized according to their importance.

- High priority Test Suites will be executed first
- Next medium priority Test Suites will be executed
- Finally, low priority Test Suites will be executed

Typically, high priority will be given to functional requirements test suites; medium priority will be given to non-functional requirements test suites; and low priority will be given to user interface requirements test suites. At the higher project planning level, Risk Based Testing means how adaptively the testing process will be changed to overcome the risk.

In general, the Federal Student Aid process for categorizing risk for purposes of scheduling testing includes the business analysts, lead testers, and application owners

as participants. The list of high priority functions of the application should not take much time to create, given the participants' knowledge of the application, and the prioritization should be based on a focus of the critical areas below:

- Business functions that are performed most often
- Areas of the application that most likely contain show-stopping defects. This may be based on complex requirements, requirements that required more clarification than others or past history (areas that have caused problems in the past)
- Functions that are critical to the business
- Functions that are federally mandated
- Defects released in applications that are exposed to outside entities, can have an adverse effect on Federal Student Aid depending on whether the errors are "limited, serious, or severe."
- Critical functions of interest to senior management and stakeholders.

Some functions may be listed more than once. When this happens, those functions should be listed as a high priority for testing, and risk should be documented along with each function.

Once your list is complete, determine if there are any functions that can be tested at a later time. The total list must be prioritized. In the end, the group should agree to the test suites that will be created for the prioritized functions.

Here is an exercise that your team may use to try to find pertinent test cases.

Participants: Business analysts, lead testers, application owners
Focus: Critical Areas
Create a list of functions that focus on:

- The functions that users perform most often
- The areas of the application that most likely contain show-stopping defects. This may be based on complex requirements, requirements that required more clarification than others, or past history (areas that have caused problems in the past)
- The functions that are critical to the business
- Functions that are federally mandated
- Functions exposed to outside entities that would cause embarrassment to FSA if a defect is released in production
- Critical functions of interest to senior management and stakeholders

Some functions may be listed more than once. When this happens, those functions should be listed as high priority functions for testing. Once your list is complete, determine if there are any functions that can be tested at a later time. The total list must be prioritized. In the end the group should agree to the test cases that will be created for the prioritized functions.

Timeframe: Quick Fire > Creation of list should not take more than ½ hour > Prioritization should also be Quick Fire

23. Intersystem Testing

Intersystem Testing is the testing of the interactions between the system under development and external systems. Whenever an application is developed that interacts with other applications, Intersystem Testing is required. Intersystem Testing must be conducted during System Testing and User Acceptance Testing (UAT). Because additional application groups are involved in the Intersystem Testing activities, communication is paramount.

Contents of Intersystem Testing

The following table shows the contents for this section of the appendix.

Contents of Intersystem Testing

Title	Brief Description
Impact on Test Planning	Steps to take when planning Intersystem Testing
Initial Planning Meeting	Topics to address during the initial planning meeting for Intersystem Testing
Test Planning Documents	Test planning documents used in Intersystem Testing
Impact on Test Phases	A list of test phases impacted by Intersystem Testing
Roles and Responsibilities	A table of who is involved in Intersystem Testing and what they do
Intersystem Testing While Interfacing Applications Are Changing	Considerations when multiple interfacing applications are changing concurrently
Managing Intersystem Testing For a Separate Team	Steps for a Test Lead to manage Intersystem Testing for a team they have not been involved with

Impact on Test Planning

Intersystem Test planning must conform to the procedures in section 2 of this document.

Initial Planning Meeting

Since Intersystem Testing involves external applications, the Intersystem Test Leads for those external applications must be involved in the planning. All Intersystem Test Leads must participate in an initial planning meeting. The initial planning meeting must:

- Identify areas to be tested and associated dependencies.
- Identify each test team's responsibilities.
- Establish schedules for intersystem testing activities, including executing tests.

- Establish defect resolution procedures.
- Establish procedures for handling non-defect incidents.
- Coordinate change request procedures.

For UAT, the Test Team and Application Owner must determine what level of Intersystem Testing will take place. With this information, the Project Manager must hold an additional planning meeting with the affected Intersystem Test Leads to coordinate UAT Intersystem Testing.

All Intersystem Test planning meetings must have an agenda, and resulting meeting minutes must be distributed to all attendees and other stakeholders.

Test Planning Documents

The MTP and PLTPs (if applicable) must address Intersystem Testing. **Appendix E-**0 and **Appendix E-**3 contain the MTP and PLTP templates, and specific guidance for addressing Intersystem Testing when writing the MTP and PLTPs.

A separate Intersystem Test Plan may be required, depending on the complexity of the application, the number of interfacing systems that need to be tested, and the criticality of those interfaces.

For both System Testing and UAT, test scenarios, test cases, and test scripts must include intersystem tests.

The external interface requirements and interface design are used to create the Intersystem Testing aspects of the MTP, PLTPs (e.g., System Test Plan and User Acceptance Test Plan), and associated test cases. Factors to consider include scheduling, inputs received, outputs exchanged, return codes, messages, and security information. Each external interface requirement should trace to one or more test cases.

Note: *External interface requirements describe system-to-system data movement interfaces. See the FSA Detailed Requirements Document Exemplar for an example of an external interface requirement.*

Impact on Test Phases

Intersystem Testing is not a separate test phase by itself, but must be conducted during specified test phases when applicable:

- **Unit Testing** does not require Intersystem Testing.
- **Integration Testing** does not require Intersystem Testing.
- **System Testing** requires Intersystem Testing when the application interfaces with other applications, as described in **Section 4.4** of this document.
- **UAT** requires Intersystem Testing when the application interfaces with other applications, as described in **Section 4.5** of this document.

Roles and Responsibilities

The table below lists the roles and responsibilities for Intersystem Testing. This table includes items from Table 16: System Testing Roles, Responsibilities, and Artifacts and Table 17: UAT Testing Roles, Responsibilities and Artifacts, as well as additional roles and responsibilities specific to Intersystem Testing.

Intersystem Testing Roles, Responsibilities, and Artifacts

Role	Responsibility	Artifact(s)
FSA Project Manager	Coordinate with representatives of other applications that interface with the system under development to establish schedules for Intersystem Testing.	• Meeting Minutes • Schedules for Intersystem Testing
Test Manager	Establish the Intersystem Test strategy Ensures that all Entrance Criteria are satisfied prior to beginning Test Preparation. Minimum Entrance Criteria include having: • Test Environment • Application login and password assigned • Interface Control Document for Intersystem Testing • Intersystem Testing activities coordinated Oversees testing activities Review all testing artifacts once testing completes Ensures the following Exit Criteria are satisfied at the end of testing: • All planned System Test Cases pass • All urgent and high priority defects identified during System Testing are resolved • All artifacts, reports, and metrics are updated by the Test Team to reflect the current state of the tested system	• Test Environment exists • Defect Reports • Completed to approve TRR • Test Summary Report • Reviewed System Test Cases • Reviewed System Test Results

Role	Responsibility	Artifact(s)
Development Team	Provide the Federal Student Aid test team the Interface Control Documents at least one month prior to the Intersystem Testing effort unless the contract states otherwise.	• Interface Control Document • Update to defect reports and impact analysis documentation
Test Team	• Creating and updating Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform each of the types of tests they will conduct during testing • Obtaining logins and passwords to interface(s) • Generating test data • Executing Test Cases using the Test Framework • Evaluating Test Results • Performing Defect Management to verify, log and retest defects	• Updated Test Plans (if needed) • Test Framework • Executed Test Suite (including test scenarios, test cases, test scripts, and test procedures) • Test Data • Evaluated System Test results • Complete documentation of defects in the defect management tool

Role	Responsibility	Artifact(s)
Intersystem Test Leads	Participate in planning meeting with Project Manager and other Intersystem Test Leads Ensures that all Entrance Criteria are satisfied prior to beginning Test Preparation for their own system. Minimum Entrance Criteria include having: • Test Environment • Application login and password assigned • Interface Control Document for Intersystem Testing • Intersystem Testing activities coordinated Oversees testing activities for their own system Review all testing artifacts once testing completes for their own system Ensures the following Exit Criteria are satisfied for their own system at the end of testing: • All planned System Test Cases pass • All urgent and high priority defects identified during System Testing are resolved All artifacts, reports, and metrics are updated by their Test Team to reflect the current state of the tested system	• Schedules for Intersystem Testing • Executed Tests (for tests they were responsible for) • Complete documentation of defects in the defect management tool

Intersystem Testing While Interfacing Applications Are Changing

If an external system needs to be modified to provide the required external interface, then special consideration must be given to the testing and release of the system being developed. It may be necessary to initially perform Intersystem Testing against a dummy interface (which doesn't perform any real functionality but simply sends and returns the correct messages)[12]. However, it is imperative to test both systems in a completed state before releasing the applications. In this situation, special attention

[12] See Crispin and Gregory, Agile Testing, p. 213.

must be given to coordinating the Intersystem Testing. Testing schedules must include extra time for these activities.

If multiple applications are changing, each application should have its own test plan document(s). If the interfacing systems are not changing, Intersystem Testing can proceed as normal. Even if the external interface is not changing, the Project Manager should consider regression testing the interfacing system.

Managing Intersystem Testing For a Separate Team

Recommendations for managing Intersystem Testing when you are not a part of the Integrated Project Team:

1. Review the requirements for the enhancement being made. Focus on the documentation applicable to Intersystem Testing, which may include:
    a. Requirements artifacts
    b. Design documents
    c. The Interface Control Document
    d. The MTP
    e. The PLTP
2. Receive an overview of the project from the requesting team Business Analyst or Technical Lead and identify interfacing applications.
3. Identify Project Managers of all interfacing applications.
4. Schedule walk-thru session with Project Managers/Technical Leads of all interfacing applications.
5. Identify Intersystem Test Team Lead(s) and pertinent stakeholders
6. Facilitate a coordination meeting with Intersystem Test Leads.
    a. Identify areas to be tested and associated dependencies.
    b. Identify responsible parties and responsibilities for executing tests.
    c. Establish schedules for executing intersystem tests.
    d. Establish defect resolution procedures, since multiple teams may need to be involved in tracking down the root cause of a defect.
    e. Establish procedures for handling non-defect incidents, and schedule a time to discuss these incidents.
    f. Establish change control procedures with the requesting team Project Manager and Change Management Team
    g. Create meeting minutes and distribute to all stakeholders.
7. Work with the Project Manager as needed to determine schedules, issues, and priorities for the Intersystem Testing.
8. Oversee testing activities.

9. Establish frequency of intersystem team meetings
10. Facilitate meetings (across multiple teams) to discuss the defects, prioritize them, and establish a plan to resolve the defects.
11. Identify the required testing artifacts and ensure the appropriate artifacts are stored in the project documentation repository
12. Coordinate intersystem test data collection for the Production Readiness Review. Ensure testing results on the PRR match to the numbers previously reported throughout the intersystem test

Note: *All Intersystem Test meetings must have an agenda, and resulting meeting minutes must be distributed to all attendees and other stakeholders.*

Appendix D: Templates

This section includes template documents, instructions and forms:

- Document templates required to support Federal Student Aid's testing requirements
- Template instruction sets for completing the attached documents
- A common set of formatting instructions for the templates
- Template forms

The template instruction sets are:

- **Template 1, Master Test Plan** – the plan that provides a central artifact to govern the planning and control of the test effort. It defines the general approach that will be employed to test the software and to evaluate the results of that testing, and is the top-level plan that will be used by managers to govern and direct the detailed testing work.
- **Template 2, Test Readiness Review Report** – the report for the multi-disciplined technical review to ensure that the subsystem or system under review is ready to proceed into formal testing. This review assesses test objectives, test methods, and procedures, scope of tests, traceability of planned tests to program requirements and user needs and safety, and confirms that required test resources have been properly identified and coordinated to support planned tests.
- **Template 3, Phase Level Test Plan(s)** – the plan that describes scope, approach, resources, and scheduled of intended test activities for a single test phase.
- **Template 4, User Acceptance Test Plan** – the plan that outlines the formal testing with respect to the Application Owner's needs, requirements, and processes conducted to determine whether a system satisfies the acceptance criteria and to enable the user, customers, or other authorized entity to determine whether to accept the system.
- **Template 5, User Acceptance Test Support Plan** – the support plan that describes the items necessary to conduct Acceptance Testing; and outlines contractor responsibilities and provides guidance for how the contractor supports Federal Student Aid.
- **Template 6, Post Implementation Verification Test Plan** – the test plan that documents the final phase of testing that occurs after the application is deployed in production.
- **Template 7, Test Summary Report** – the report summarizing testing activities and results; contains an evaluation of the corresponding test items against exit criteria.

- **Template 8, Performance Test Plan** – the plan that documents how to create and gain approval of the mechanisms used to do a variety of types of tests to gather these kinds of statistics:
  - Network and/or Web server throughput limits
  - Individual server resource utilization under various loads
  - Search speeds, query optimization, table/row locking , and speed versus capacity measurements for databases
  - Load balancing overhead/capacity/effectiveness
  - Speed and resource cost of security measures
- **Template 9, Performance Testing Test Report** – the output of Performance Testing is a collection and presentation of data in ways that show whether the system is missing or meeting goals, in what areas, and by how much, without requiring the viewer of this data to have any special technical knowledge of the system under testing.
- **Template 10, Hardware Test Plan** – the Hardware Test Plan starts with a brief synopsis of the background of the project under test. It also includes topics such as: the schedule information, hardware details, environment location, test facilities that support the test effort, Configuration Management Process, roles and responsibilities, assumptions and constraints, and Risks and Lessons Learned.

The template forms are:

- Template 11, Performance Test Daily Reports
- Template 12, Defect Severity and Current State
- Template 13, High Level Test Summary
- Template 14, Incident Report By Testing Phase
- Template 15, Disaster Recovery Test Readiness Review

Template Formatting Instructions

Component	Description
Overall Guidance	Text enclosed in square brackets is included to provide guidance to the author and should be deleted before publishing the document. A paragraph entered following this style will automatically be set to normal (style=Body Text).
Template Guidance	Before you start filling out the content of the new document, go to File/Properties/Custom. Enter the Document Name, Version Date (in DD/MM/YYYY format), and Version Number (in x.x format) in the custom properties that bear these respective names. In the document, press CTRL+A, then F9, and click OK. Go into every footer for each section, press CTRL+A, then F9, and click OK. If a section marked "Optional" is deemed unnecessary or inapplicable to the document, the section should be retained and the content for the section should be "Not applicable".
Formatting	**Body Text** The text in the document is the body text style, Arial 12, color-black, left-justified. **Numbered Lists (for Sections)** Numbering for lists should follow the [lower-case letter].[number].[number] pattern. All levels in the list will be flush-left with the main text in the section. Example: 1. List item a. List item i. List sub-item **Bulleted Lists (Do not change the font once it is set)** • Bulleted Lists: Level 1 bullets should be Symbol 9 (Format, Bullets & Numbering, Customize, Font-Symbol, Size-9) for level 1 of indentation – black bullet symbol (two spaces in between bullet and first letter of sentence—see ruler at top of document.) Indent first bullet three spaces from left margin.) – Level 2 bullets should be dash symbol for level 2 of indentation. (Font-Symbol, Size-9.) (Two spaces in between bullet and first letter of sentence—see ruler at top of document.) Indent second bullet three spaces from start of Level 1 bullet. ◦ Level 3 should be open bullet symbol for level 3 of indentation. (Font-Symbol, Size-9) (Two spaces in between bullet and first letter of sentence—see ruler at top of document.) Indent level 3 bullets three spaces from start

Component	Description
	of Level 2 bullets. **Page setup:** Margins: 1 inch Top, Bottom, Left, and Right; “From Edge”= .5 header, .5 footer. **Section Style Data** Sections should start on their own page. **Heading 1** Font: Arial 16 pt., bold, black Paragraph spacing: “12 pt spaces Before, 18 pt After” Keep with next, Level 1, Outline numbered, tabs 0.5 **Heading 2** Font: Arial 14 pt., bold, black Paragraph spacing: “12 pt spaces Before, 12 pt After” Keep with next, Level 2, Outline numbered, tabs 0.5 **Heading 3** Font Arial 13 pt, unbold Paragraph spacing: “6 pt spaces Before, 6 spaces After” Keep with next, Level 3, outline numbered, tabs 0.5, Indent: 0.5 **Numbered Lists (for Sections)** All major section headings (Heading 1) should be in numerical order (e.g., Section 1, Section 2, Section 3, etc.) Secondary headings (Heading 2) should be in numerical order (e.g., 1.1, 1.2, 2.1, 2.2, 3.1, 3.2, etc.) Sub-secondary headings (Heading 3) should be in numerical order (e.g., 1.1.1, 1.1.2, 2.1.2, 2.1.3, 3.1.1, 3.1.2, etc.)
Tables	You may utilize a few styles for table text and table headings. Caption: Table Caption Heading style: Arial Bold, Arial 12, paragraph spacing “18 pt Before, 3 pt After” Table Heading: Arial 11, Bold, paragraph spacing “18 pt Before, 3 pt After” Table Text: Can vary from Arial 9-11 (because documents can vary in size and length, the author may choose which is suitable for his/her document) (The author may choose from Table Text 9, Table Text 10, and Table Text 11 styles.) Bullets within tables: Font of bullet is 6 pt; bullet position: indent at .1”, text position: indent at .15”
Headers	The header contains the title of the document on the left, and the section on the right. **How to add section titles in the header:**

Component	Description
	1. Create a section break in the page ahead of the page of the header you want to change. 2. Click View, Headers and Footers. 3. Put your cursor in the header. Ensure that “Same as Previous” is turned off. 4. Click Insert, Cross-Reference. a. In the drop-down box, ensure it states “Heading” and that the “Insert Reference To” states “Heading Text”; then, select the heading in the dialog box that reflects the heading you want to insert. b. Click Insert.
Footers	The footers contain the version number of the document on the left, page numbers in the middle, and the version date on the right. (Regarding version numbers, keep the numbering system to sequential numbering. (For “in-house,” you may use alternate numbering systems like “1.1.1” and so forth.)
Appendices	A cover page is required before each Appendix. (Heading 7, Arial 12, centered, paragraph spacing 156 pt “before.”) **Appendix A** Is reserved for acronyms and abbreviations only. List all the acronyms used in the document. Describe all associated terms for this project/document methodology, etc. Footers: page numbering should continue. **Appendix B** Is reserved for the glossary only. List all glossary terms used in the document. Footers: page numbering should continue.
Additional Appendices	A cover page is required before each Appendix. (Heading 7, Arial 12, centered, paragraph spacing 156 pts “before.”). Additional Appendices should be in sequence to the previous (e.g., Appendix A, B, C, D, etc.). Any appendices created after Appendix A and B are document-specific. Footers: continue page numbering

1. Master Test Plan Template

Instructions

For Agile projects, not all sections of the Master Test Plan need to be filled in initially. It is a living document. Sections expected to be filled in at a later time during the course of the project should be marked as "To Be Determined". Sections that need not be filled in during any sprint should be marked as "Not Applicable". All templates must follow the formatting guidelines set forth in the Template Formatting Instructions.

Component	Description
Cover Sheet	A sample of the cover sheet is provided in the template.
Document Version Control	Provide information concerning the version of the document, the date of the change, and a description of the change.
Table of Contents	A sample of the table of contents is provided in the template.
Project Name	Provide information about the project, including the name.
Executive Summary	• A one-page high-level introduction describing, in brief: • Document purpose • Document users • Document uses (what, when, how) • Document owners • Document main point of contact for questions or feedback]
Section 1. Introduction	
1.1 Purpose	Describe the purpose of the Test Plan • For example: • Provides a central artifact to govern the planning and control of the test effort. It defines the general approach that will be employed to test the software and to evaluate the results of that testing, and is the top-level plan that will be used by managers to govern and direct the detailed testing work. • Include the complete lifecycle, the specific phase, and the project name. • Identify the items that should be targeted by the test. • Identify the levels of testing to be performed (e.g., Unit, Integration, System, and User Acceptance level testing). • Identify the motivation for and ideas behind the test areas to be covered. • Outline the testing approach that will be used. • Identify how defects will be categorized and managed. • Identify the required resources and provide an estimate of the test efforts. • List the deliverable elements of the test project.

Component	Description
1.2 Scope	Defines the types of testing, such as Functionality, Usability, Reliability, Intersystem, Performance, and Supportability, and if necessary the levels of testing, (e.g., Unit, Integration, System, and User Acceptance) that will be addressed by this Test plan. It is also important to provide a general indication of significant elements that will be excluded from scope, especially where the intended audience might otherwise reasonably assume the inclusion of those elements. For Agile projects, this document should contain general information that applies to all sprints, even though the contents of each sprint will not be decided until the sprint planning meeting. This document may need to be updated often. Note: Be careful to avoid repeating detail here that you will define in section 2, Overview of Planned Tests.
1.3 Intended Audience	Provide a brief description of the audience for whom you are writing the Test plan. This helps readers of your document identify whether it is a document intended for their use, and helps prevent the document from being used inappropriately. Note: The document style and content usually alters in relation to the intended audience. This section should only be about three to five paragraphs in length.
1.4 Document Organization	This section outlines how the contents (sections and appendices) of the test plan will be organized in the document.. Section 1 - Introduction: is always the introduction, and a brief description should be provided. Section 2 - [Section Name]: includes the name of the section as well as a brief description. This format must be followed for all of the sections that are included in the document. Appendix A – Acronyms and Abbreviations: this appendix letter is reserved for the acronyms and abbreviations and is a standard titled section. Appendix B – Glossary: this appendix letter is reserved for the glossary and is a standard titled section. Appendix C - [Appendix Name]: includes the name of the appendix as well as a brief description. This format must be followed for all of the appendices that are included in the document.

Component	Description
1.5 References and Related Documents	This subsection should provide a complete list of all documents referenced elsewhere in the Test Plan. Each document should be identified by title, report number (if applicable), date, and publishing organization. Specify the sources from which the references can be obtained. This information may be provided by reference to an appendix or to another document.
Section 2. Overview of Planned Tests	This section provides a high-level overview of the testing that will be performed. This section will address all phases of testing including Unit, Integration, System, and User Acceptance phase of testing. The outline in this section represents a high level overview of both the tests that will be performed and those that will not.
2.1 Features to be Tested	Provide a high-level overview of the major testing planned for project/ phase. Note what will be included in the plan and record what will explicitly not be included in the following section titled Features not to be Tested. Provide a high level list of the major target test items. This list should include both items produced directly by the project development team, and items that those products rely on; for example, basic processor hardware, peripheral devices, operating systems, third-party products or components, and so forth. This may simply be a list of the categories or target areas. For Agile projects, this section should be filled in for the initial sprint, but may be appended during subsequent sprints.
2.2 Features not to be Tested	Provide a high-level overview of the potential tests that might have been conducted but that have been explicitly excluded from this plan. If a type of test will not be implemented and executed, indicate this in a sentence stating the test will not be implemented or executed and stating the justification, such as: "These tests do not help achieve the evaluation mission." "There are insufficient resources to conduct these tests." "These tests are unnecessary due to the testing conducted by xxxx." As a heuristic, if you think it would be reasonable for one of your audience members to expect a certain aspect of testing to be included that you will not or cannot address, you should note its exclusion.

Component	Description
Section 3. Test Approach	The Test Approach presents an overview of the recommended strategy for analyzing, designing, implementing and executing the required tests as well as managing the defects identified during testing. Section 2 identifies what items will be tested and what types of tests will be performed. This section describes how the tests will be realized and communicated (e.g., how to handle testing with other entities, meeting frequency to discuss defects, etc.). As you identify each aspect of the approach, you should update Section 7, Responsibilities, Staffing, and Training Needs, to document the test environment configuration and other resources that will be needed to implement each aspect. The testing approach might incorporate the governing mission statement(s) and one or more concerns including: • Find as many defects as possible; • The test approach must also address security, Section 508 compliance, Intersystem Testing, etc. • Find important problems, assess perceived quality risks; • Advise about perceived project risks; • Certify to a standard; • Verify a specification (requirements, design or claims); • Advise about product quality, satisfy stakeholders; • Advise about testing; • Fulfill process mandates and so forth; and • Each mission provides a different context to the test effort and changes the way in which testing should be approached. • Testing will be motivated by many things, quality risks, technical risks, project risks, use cases, functional requirements, non-functional requirements, design elements, suspected failures or faults, change requests, and so forth. For Agile projects, this section may not be extensive. Agile projects must document what levels of testing take place for each sprint (i.e., Unit Testing, Unit Integration Testing, System Testing) and what levels of testing take place only at the end of a group of sprints (i.e., UAT). *More specific details of the individual testing tasks are defined in a number of different ways, depending on project culture. For example:* *Defined as a list of tasks in this section of the Test Plan, or in an accompanying appendix* *Defined in a central project schedule (often in a scheduling tool such as Microsoft Project).*

Component	Description
3.1 Measuring the Extent of Testing	Describe what strategy you will use for measuring the progress of the testing effort.[13] A good measurement strategy will report on multiple dimensions including coverage (against the product and/or against the plan), effort, results, obstacles, risks (in product quality and/or testing quality), and historical trend (across iterations/sprints and/or across projects). Consider appropriate traceability strategies for Coverage of Testing against Specifications enables measurement of the extent of testing. For Agile projects, this section is optional.
3.2 Identifying and Justifying Tests	Describe how tests will be identified and considered for inclusion in the scope of the test effort covered by this strategy. Describe the process that will be used to determine test suites/cases/procedures and test scripts. Provide a listing of resources that will be used to stimulate/ drive the identification and selection of specific tests to be conducted, such as Use Cases, Requirements documents, User documentation and/ or Other Reference Sources.
3.3 Conducting Tests	One of the main aspects of the test approach is an explanation of how the testing will be conducted, covering the selection of quality-risk areas or test types that will be addressed and the associated techniques that will be used. If you are maintaining a separate test strategy artifact that covers this, simply list the test types or quality-risk areas that will be addressed by the plan, and refer to the test strategy artifact for the details. If there is no separate test strategy artifact, you must provide an outline here of how testing will be conducted for each technique: how design, implementation, and execution of the tests will be done, and the criterion for knowing that the technique is both useful and successful. For each technique, provide a description of the technique and define why it is an important part of the test approach by briefly outlining how it helps achieve the Project and Evaluation Missions. For Agile projects, this section should be filled in for the initial sprint, but may be updated during subsequent sprints. It should emphasize the importance of regression testing – both manual and automated (when available).

[13] When deciding on a measurement strategy, it is important to consider the following advice from Cem Kaner, 2000 "Defect count metrics reflect only a small part of the work and progress of the testing group. Many alternatives look more closely at what has to be done and what has been done. These will often be more useful and less prone to side effects than defect count metrics."

Component	Description
Section 4. Entry and Exit Criteria	
4.1 Test Plan Entry Criteria	Specify the criteria (i.e., in addition to the Test Readiness Review) that will be used to determine whether the execution of the Test plan can begin.
4.2 Test Plan Exit Criteria	Specify the criteria that will be used to determine whether the execution of the Test plan is complete, or that continued execution provides no further benefit.
4.3 Suspension and Resumption Criteria	Specify the criteria that will be used to determine whether testing should be prematurely suspended or ended before the plan has been completely executed, and under what criteria testing can be resumed.
Section 5. Deliverables	In this section, list the various artifacts that will be created by the test effort that are useful deliverables to the various stakeholders of the test effort. Also, list the due dates of each deliverable. Don't list all work products; only list those that give direct, tangible benefit to a stakeholder and those by which you want the success of the test effort to be measured. The Master Test Plan is required for all projects. For large scale projects the following artifacts are required in addition to this Test Plan: • System Test plan and System Test Summary Report • User Acceptance Test (UAT) Test Plan and User Acceptance Test Summary Report • User Acceptance Test (UAT) Support Plan • Data Conversion Balancing Report • Post Implementation Verification Plan • Defect Reports and Test Status Reports • Testing Metrics • **TRR Information goes here**
5.1 Test Evaluation Summaries	Provide a brief outline of both the form and content of the test evaluation summaries, and indicate how frequently they will be produced. For Agile projects, this section is optional.
5.2 Reporting on Test Coverage	Provide a brief outline of both the form and content of the reports used to measure the extent of testing, and indicate how frequently they will be produced. Give an indication as to the method and tools used to record, measure, and report on the extent of testing. For Agile projects, this section is optional.

Component	Description
5.3 Perceived Quality Reports	Provide a brief outline of both the form and content of the reports used to measure the perceived quality of the product, and indicate how frequently they will be produced. Give an indication about the method and tools used to record, measure, and report on the perceived product quality. You might include some analysis of Incidents and Change Request over Test Coverage. For Agile projects, this section is optional.
5.4 Incident Logs and Change Requests	Provide a brief outline of both the method and tools used to record, track, and manage test incidents, associated change requests, and their status.
5.5 Detailed Test Results	Identify how test results will be delivered to Federal Student Aid. Either a collection of Microsoft Excel spreadsheets listing the results determined for each test suite/case, or the repository of both test logs and determined results maintained by a specialized test product is acceptable.
Section 6. Environmental Needs	*This section presents additional resources, other than staff, that are required for the Test plan. This section shall describe the test environment and test lab facilities required to perform the testing. Early Intersystem Testing may require a special harness, dummy program, and/or emulator to supply the messages needed to test interfaces. Intersystem Testing may also make use of the Enterprise Service Bus to verify particular messages are sent between interfacing applications (service providers and service consumers).*
6.1 Base System Hardware	This section contains a list of the hardware elements required in the test environment. The specific elements of the test system may not be fully understood in early iterations, so expect this section to be completed over time.
6.2 Base Software Elements in the Test Environment	This section contains a list of the software elements required in the test environment. The specific elements of the test system may not be fully understood in early iterations, so expect this section to be completed over time.
6.3 Productivity and Support Tools	Include the specific tools used to support the test effort. Also, include the tool category or type, brand name of the tools, and the version of the tools. For Agile projects, this section may be “To Be Determined” for the first sprint. It may be populated/updated during subsequent sprints. It should emphasize the creation and reuse of automated testing techniques (when available).

Component	Description
6.4 Test Environment Configurations	This information may not be available during the definition phase of the lifecycle. Therefore, this information must be detailed in the individual Test Phase test plans. After the test environment configurations are defined, list the configuration name and description in this section. For Agile projects, this section may be “To Be Determined” for the first sprint. It may be populated/updated during subsequent sprints.
Section 7. Responsibilities, Staffing, and Training Needs	This section presents the required resources to address the test effort outlined in the Test plan, the main responsibilities and the knowledge or skill sets required of those resources.
7.1 People and Roles	Add or delete items as appropriate. This section should list the responsibilities of Intersystem Test Leads, if Intersystem Testing is performed.
7.2 Staffing and Training Needs	This section describes the training required to enable the testing team to effectively test the system. Training topics may include the use of testing tools, testing procedures and techniques, or other related information. Give thought to your training needs, and plan to schedule this based on a Just-In-Time (JIT) approach. There is often a temptation to attend training too far in advance of its usage when the test team has apparent slack. Doing this introduces the risk of the training being forgotten by the time it's needed. Look for opportunities to combine the purchase of productivity tools with training on those tools, and arrange with the vendor to delay delivery of the training until just before you need it. If you have enough headcount, consider having training delivered in a customized manner for you, possibly at your own site. The test team often requires the support and skills of other team members not directly part of the test team. Make sure you arrange in your plan for appropriate availability of System Administrators, Database Administrators, and Developers who are required to enable the test effort.
7.3 Security	This section identifies the individual / team responsible for security testing.
Section 8. Key Project/Phase Milestones	Identify the key schedule milestones that set the context for the Testing effort. Avoid repeating too much detail that is documented elsewhere in plans that address the entire project. Key Milestones will be identified for all levels of testing per phase – Unit, Integration, System and User Acceptance. The required components are included in the template for the schedule of key milestones. Also included are blank rows for additional components.

Component	Description
Section 9. Master Plan Risks, Dependencies, Assumptions, and Constraints	The following contents of this section should be listed in table format: Table 9-1: Risks List any risks that may affect the successful execution of this Test plan, and identify mitigation and contingency strategies for each risk. Also indicate a relative ranking for both the likelihood of occurrence and the impact if the risk is realized. Table 9-2: Dependencies List any dependencies identified during the development of this Test plan that may affect its successful execution if those dependencies are not honored. Typically, these dependencies relate to activities on the critical path that are prerequisites or post-requisites to one or more preceding (or subsequent) activities. You should consider responsibilities you are relying on other teams or staff members external to the test effort to complete, timing and dependencies of other planned tasks, and the reliance on certain work products being produced. If Intersystem Testing is performed, indicate the dependencies with the Intersystem Test Leads regarding application availability, data in the application, and human resource availability. Include the respective responsibilities for all testing activities, including developing test cases, executing test cases, and resolving defects. Table 9-3: Assumptions List any assumptions made during the development of this Test plan that may affect its successful execution if those assumptions are proved incorrect. Assumptions might relate to work you assume other teams are doing, expectations that certain aspects of the product or environment are stable, and so forth. Table 9-4: Constraints List any constraints placed on the test effort that have had a negative effect on the way in which this Test plan has been approached. For Agile projects, this section should be filled in for the initial sprint, but may be appended during subsequent sprints.
Section 10. Management Process and Procedures	Outline what processes and procedures are to be used when issues arise with the Test plan and its enactment.
10.1 Acceptance of Test Plan	Define any management and procedural aspects of the measurement and assessment strategy outlined in Section 3.1, Measuring the Extent of Testing.
10.2 Assessing the Deliverables of this Test Plan	Outline the assessment process for reviewing and accepting the deliverables of this Test Plan.

Component	Description
10.3 Problem Reporting, Escalation, and Issue Resolution	Define the reporting and communication process mechanism that impact the success of the test effort.
10.4 Managing Test Cycles	Outline the management control process for a test cycle.
10.5 Lessons Learned	This section describes the lessons learned from testing.
10.6 Approval and Signoff	Outline the approval process and list the job titles (and names of current incumbents) that initially must approve the plan, and sign off on the plan's satisfactory execution. List those that have sign-off responsibility (required); those that are accountable; those that were consulted and those that were informed.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions*. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Template Document	Please contact the Enterprise Test Team for the Word document.

2. Technical Stage Gate 2, Test Readiness Review Template

Instructions

The following information provides instructions for completing the Test Readiness Review template. Instructions are provided for each major section of the template.

2.1 Header Information

The test manager / test lead should provide the following template header information:

- Project - The common name that has been assigned to the project
- Scope - A description of the scope of the project
- Team – The name of the team responsible for executing the test phase
- Date – The date the TRR will be held and the date the testing will be scheduled.
- Phase – Select the appropriate phase from the provided drop down list or enter the applicable test phase name.

  Note: *If you select one of the options from the drop down list, the template will automatically reflect default information that needs to be documented for the testing phase.*

Provide as necessary any significant dates and descriptions in the benchmark section.

2.2 Items 1.1 – 1.8: Test Phase Participants

Provide information concerning expected test plan participants and their status. In the appropriate columns of the template, record the information provider (owner), expected due date, actual due date completion, confirmation and any accompanying artifact references, explanations or comments.

2.3 Items 2.1 – 2.6: Test Phase Logistics

The test manager / test lead should identify for inclusion in the TRR relevant test phase logistics such as space requirements, applicable batch job schedules, the schedule and plan for daily meetings that will take place to review testing results, any special technical requirements, and the defect management plan. In the appropriate columns of the template, record the information provider (owner), expected due date, actual due date completion, confirmation and any accompanying artifact references, explanations or comments regarding test phase logistics.

2.4 Items 3.1 – 3.4: Test Phase Scope

Confirm that detailed requirements have been documented and base-lined. The test manager / test lead should also contact the FSA test team lead to confirm that application scope of the test phase and the functional scope of the test phase has been defined, delivered and reviewed by the test lead.

2.5 Items 4.1 – 4.8: Documentation

Confirm the development of documentation that supports the applicable test phase:

- Final Design Document
- Functional Specs baseline
- Traceability Matrix baseline
- Master Test Plan
- Phase Test Plan
- Requirements Traceability Matrix
- System Test Plan Approval (applicable for system test phases)
- User Acceptance Test Plan Approval (applicable for user acceptance test phases)

2.6 Items 5.1 – 5.6: Current Test Phase (Entrance Criteria)

Confirm the test entrance criteria have been provided or are scheduled to be provided prior to the onset of testing. If the TRR is being prepared to support system testing, then confirm the development and documentation of applicable release-specific test cases and regression test cases. The test cases should also include the anticipated results of the tests. If the TRR is being prepared to support system testing then confirm the identification and documentation of any testing dependencies.

Regardless of the type of testing that is to be performed, the test manager / test lead of the TRR should ensure that data has been prepared for testing and should provide any known testing limitations.

2.7 Items 5.7.1 – 5.7.4: Test Environment

The test environment is a critical component of testing regardless of the phase. The TRR test manager / test lead should confirm that all testing tools have been installed and are operational. The test manager / test lead should confirm that all required code has been deployed and should know what versioning of the code is being used in the test environment. If more than one test environment exists, the test manager / test lead should ensure that adequate separation exists for each test environment. The TRR test manager / test lead should also be aware of any test environment limitations that may exist.

2.8 Items 6.1 – 6.4: Previous Test Phase (Exit Criteria)

The TRR test manager / test lead must gather information concerning any previously completed test phases. The TRR test manager / test lead should determine if independent system testing has been completed. If any urgent or high defects have been identified, the test manager / test lead must gather information concerning the status of those defects. The test manager / test lead should know if urgent defect have been resolved and communicated to the members of the test team. The test manager / test lead should also know if defects, regardless of their type / urgency are open or closed and whether or not their current status has been communicated to FSA. The TRR test manager / test lead should contact the lead test manager and obtain formal information regarding whether or not the maturity of the application is acceptable for entry to the next test phase. The test manager / test lead should also make sure that all independent system test documentation has been delivered to FSA.

2.9 Items 7.1 – 7.3: Inter-system End-to-End Test Cases

Concerning intersystem testing, the TRR test manager / test lead should determine if end-to-end integrated test cases have been defined, that data has been created for intersystem testing, and that all end-to-end testers have been identified.

2.10 Items 8.1 – 8.2: Configuration Documents

The TRR test manager / test lead should determine if documentation of a configuration management plan has begun and should determine the status of any configuration change requests. Have required configuration change requests been submitted? Have submitted change requests been approved?

2.11 Items 9.1 – 9.4: Risks

The TRR test manager / test lead should gather information concerning all risks that have been identified for the system. The test manager / test lead should have a description for each risk that has been identified. The test manager / test lead should determine for each risk if a mitigation or contingency has been defined. For each risk the test manager / test lead should determine what the probability, consequence and overall risk is. For every test, the potential exists for something to go wrong. The TRR test manager / test lead should identify if a fall-back plan has been established to address unforeseen problems that may be encountered during testing.

2.12 Items 10.1 – 10.7: Pretesting / Access

With each test phase there are a number of pre-testing activities that need to be completed or planned for, prior to the approval of testing. The TRR test manager / test lead should identify all pre-test activities that need to be completed or planned for prior to testing. Some of these activities may include:

- Ensuring FSA staff have access to the system
- Ensuring applicable contractor testers have access to the system
- Ensuring that FSA staff have access to testing tools

2.13 Items 11.1 – 11.12: LMM Compliance Requirements

There are a number of LMM requirements that the TRR test manager / test lead should gather for inclusion in the template. Some of this information includes:

- Implementation/Transition Management Plan
- UI Specifications
- Source Code Deployable Packages
- Solution User Manual
- Release Version Description
- Operations and Maintenance Plan
- Business Impact Analysis
- Information Technology Contingency Plan
- Disaster Recovery Plan
- Memorandum of Understanding

- Lessons Learned Documentation
- LMM Tailoring Plan

2.14 Items 11.13.1 – 11.13.5: Security

There are a number of security related issues that need to be addressed prior to a system being authorized to operate. The TRR test manager / test lead should determine the status of the system security plan, risk assessment and security authorization package that has been prepared for the system. For systems that contain personal information the TRR should confirm that a system of record notice (SORN) has been prepared. For systems that transmit or receive significant information to or from other systems the TRR should determine if interconnection security agreements have been established and implemented.

2.15 Certification Section:

Reflect the certification status of the test plan in the TRR template. One of three certification results should be selected from the drop-down list that exists in the Certification section of the TRR template:

1. Testing is ready to begin
2. Testing is ready to begin with the following conditions:
3. Testing will be delayed due to the following reasons:

Certification results **2) Testing is ready to begin with the following conditions** and **3) Testing will be delayed due to the following reasons** must be accompanied by applicable conditions or reasons.

All templates must follow the formatting guidelines set forth in the Template Formatting Instructions.

Instructions

Component	Description
Cover	A sample of the cover sheet is provided in the template.
Test Readiness Review Form	Complete form.
Document Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
1. Creation of Test Readiness Review Document	This section contains the test readiness review document. The list can be tailored based on the test phase requirements. Once complete, all (Integration, System and User Acceptance Tests) Test Readiness Review Documents, meeting information and review results for a project must be included in this Test Readiness Review Report.
2. Test Readiness Review Meeting	This section contains details about the test readiness review meeting. This also includes the meeting date, test phase for which the readiness review meeting is going to be held, the list of participants and their roles.
3. Test Readiness Review Results	This section provides the Federal Student Aid decision on whether or not an application can move to the next test phase or into production. The rationale for the decision is to be provided below.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions*. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Template Document	Please contact the Enterprise Test Team for the Word document.

2.16 Signatures

The actual signatures from the following authorities must be obtained in order to certify the Test Readiness Review:

- Program Manager
- FSA Test Manager
- FSA Project Manager
- Development Team Lead
- Business Owner
- Contractor Test Manager
- Contractor Project Manager

Note: A proxy signature can be used in place of the identified role provided it has been authorized.

The test manager / test lead should ensure that authorities are informed that, by certifying and signing the TRR, they consider plans to be sufficiently complete to begin testing for the agreed upon system functionality and that process designs have been reviewed and agreed to with respect to what is to be tested and how it should be tested.

Submit the completed and signed TRR to FSALMMStageGate2TRR@ed.gov

3. Phase Level Test Plan Template

Final plans should follow the formatting guidelines set forth in the Template Formatting Instructions. The guidance in this template is what is required. The creator of this test plan may add other details as needed based on the type of application under test and the requirements that are being tested.

Instructions

Component	Description
Cover Sheet	A sample of the cover sheet is provided in the template.
Table of Contents	A sample of the table of contents is provided in the template.
	Provide phase and system name, Contractor Project Manager signature and date, requirements status, comments if applicable, Federal Student Aid Test Manager or Project Manager signature and date, requirements status, and comments if applicable.
Document Version Control	Provide document information, document history, and release authorization.
1. Introduction	This section privides an overview of the test phase and provides reference information concerning materials that are relevant to testing.
1.1. Overview	This section provides a brief synopsis of the background of the project under test. State the purpose of the [Phase] test plan and state the sub phases that would be covered in this test plan. This template can be used to create a Unit Test plan, an Integration Test Plan, and a System Test Plan. For Agile projects, state which iteration(s) would be covered in this test plan.
1.2. References	This section identifies the reference material that will be used or has been used in order to support the test effort. This section may also include additional reference material to support any specialized testing If performance testing is included as part of the PTP, include the following statement, “Performance Test Plan - See the Enterprise Performance Test Team for the performance testing documentation”
2. Planning	The section provides information concerning preparation activities and requirements relevant to the test phase.

Component	Description
2.1. Schedule	This section contains the schedule information. Include milestone dates. This information could be in the form of a work breakdown structure. For Intersystem Testing, consideration should be given to the availability of interfacing application groups in order to establish a realistic schedule. For Agile projects, the schedule should take into account the Integration Testing performed at the end of each sprint, as well as the System Testing and User Acceptance Testing performed before a customer release.
2.2. Environmental Needs	This section contains the hardware details, environment location, and test facilities that support the software test effort.
2.2.1. Test Environment	This section describes the test environment. For example – Windows 2000, WebSphere, Internet Explorer (versions 9-11), Firefox, etc.
2.2.2. Testing Tools	This section contains details about the testing tools that will be used and how they will be used. For Agile projects, this section may emphasize the creation and reuse of automated tests and associated testing tools (when available).
2.3. Roles and Responsibilities	This section identifies the roles and responsibilities of the Federal Student Aid and/or Contractor for the testing phase. This section should provide a list of individuals and their roles in supporting the testing activity. The list should be in table format and all contents of the table should be based on the specific project.
2.4. Staffing and Training Needs	This section consists of the following items: Number of resources and skills required for the testing project Identification of areas where training is required and description of the plan for training testers (if applicable, include the training of Federal Student Aid testers) Include a timeframe for training
2.5. Test Team and Development Team Coordination	This section details the development team responsibilities during [phase] testing. This may include analysis of defects, creation of impact analysis statements, attending System Change Control Board meetings, etc. For Intersystem Testing, this section should also list the responsibilities of Intersystem Test Leads. For Agile projects, this section may not be extensive since the test team and development team are intended to interact throughout the project.

Component	Description
3. Test Configurations	This section provides information concerning the features and components to be tested.
3.1. Features to be tested	This section describes the list of features (Functions), which will be tested or demonstrated by the test. The description is provided at a high level without going into the technical details.
3.2. Features not to be tested	This section describes the list of features (Functions), which will not be tested.
3.3. Components to be tested	This section identifies and lists the components to be tested within the scope of this test plan.
4. Test Approach	This section includes details of testing covered by the plan and the objectives, scope, rationale and communication plan. For Agile projects, this section may not be extensive.
4.1. Test Strategy	This section describes the test strategy, which includes the following: • Processes • Assumptions • Test data creation and test data results verification process • Test suite writing process • Peer Review Process • Regression Cycles • Configuration Management Process • Handling of meetings and other Organizational Processes • Intersystem Testing needs (see Appendix D.23) • Add other information, as needed
4.1.1 Test Process	This section provides a brief synopsis of the overall test process. Provide details of testing covered by the plan and the objectives, scope, and rationale.
4.1.2. Assumptions and Constraints	This section identifies the known assumptions and constraints related to the test strategy. If needed, list mitigation strategies.
4.1.3. Test Work Products	This section identifies and lists the documents that will be created to support the test effort. State the purpose of each document that will be created. List names or teams expected to review each document. This section also includes information on how the test results will be delivered to Federal Student Aid for approval purposes.

Component	Description
4.1.3.1. Test Artifact Details	This section describes in detail how each document will add value to the test effort; the process for delivering each document to Federal Student Aid; the Configuration Management process for documentation; peer review process of documentation; test reports; test metrics; how test will be conducted; how test suites will be created and how test results will be presented to Federal Student Aid. Other information may be added to this section. For Agile projects, this section may not be extensive.
4.1.4 Traceability	This section covers how test documentation traces back to the requirements that are under test. Provide a list of the requirements being tested in the testing phase along with the corresponding test cases being executed in the phase. For Agile projects, this section should provide a list of the user stories being tested within the sprint along with corresponding test cases that will be executed in the sprint.
4.1.5. Test Data	This section covers how test data will be created and identified for the test effort.
4.1.6. Regression Strategy	This section covers how regression testing will be performed and the regression cycles. For Agile projects, this section may be substantial since many Agile methodologies develop code iteratively, need many regression tests, and recommend creating automated regression tests. It should address when full regressions should be performed (e.g., during the Component Interface Testing activity, during the User Acceptance Testing phase, etc.).
4.2. [xxx] System Approach and Techniques	This section covers exceptions pertaining to a particular subsystem of an application. If the exceptions are significant, a separate test plan must be created for the subsystem and the test plan will be referenced in the 1.2. References section.. The Contractor and the Federal Student Aid Test Manager and Project Manager will make this decision at the beginning of the project. Examples of a subsystem that has significant exceptions are Mainframe and WEB
4.4. Test Estimation	This section describes the test estimation technique that will be followed for this test effort. The details should cover estimation for the test effort and how estimates will be determined related to regressing defects. For Intersystem Testing, this should take into account coordinating with multiple application groups.

Component	Description
4.5. Test Readiness Review	This section contains a test readiness review process. Include a reference to the Test Readiness Review document, which must be an appendix to this [phase] test plan. For Agile projects, Test Readiness Reviews are not held for every sprint, but only in preparation for System Testing/ User Acceptance Testing.
4.6. Pass/Fail Criteria	This section contains the conditions of the pass/fail criteria.
4.7. Entry and Exit Criteria	This section contains the criteria that determine when testing can begin and when the testing process can be stopped.
4.7.1. Service Level Agreement	This section covers the service level agreement (SLA) for the application under test. Connect the SLA information to exit criteria. Example: If testing a voice response system, how many responses must be recognized before Federal Student Aid will accept the product?
5. Defect Management	This section describes the defect management process that will be used for this project. Include information on documenting issues, defect classification and the change control process. Include the name of the defect management tool. Include how Federal Student Aid will be involved in the defect management process.
5.1. Documenting Issues	This section describes the process for documenting defects. This section must include when an issue is added to the defect tracking tool, meetings that must be held to discuss issues, and making the determination of when an issue becomes a defect.
5.2. Defect Classification	This section describes the different states used for classifying defects. Describe in detail the severity, complexity, status and priority classification.
5.3. Change Request	This section describes the process for handling change request. If this process is detailed in another project document, state the name of the document and provide an abbreviated version of the process.
5.4. Suspension Criteria and Resumption Requirements	This section lists the known conditions that may result in the suspension of testing. The section addresses whether all or a portion of the testing is suspended. Include the process for determining when testing can resume. For example, if an application is under test and the login feature does not work, testing shall be suspended and shall resume only when the login feature has been corrected and a tester has approved that the login feature meets requirements.
6. Risk and Mitigation	This section describes known risks and mitigation strategies for each risk.

Component	Description
7. Lessons Learned	This section describes the lessons learned from this phase of testing.
8. Approvals	This section describes the approval process and includes the job titles and names of current incumbents that must approve the plan. List those that have sign-off responsibility (required), those that are accountable, those that were consulted and those that were informed.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions*. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Template Document➔	Please contact the Enterprise Test Team for the Word document.

4. User Acceptance Test Plan Template

Final plans should follow the formatting guidelines set forth in the Template Formatting Instructions.

Instructions

Component	Description
Cover	A sample of the cover sheet is provided in the template.
User Acceptance Test Plan Form	Complete form.
Document Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
1. Overview	This section provides a brief synopsis of the background of the project under test. State the purpose of the User Acceptance Test Plan, and state the sub phases (Functional testing and/or Intersystem Testing and/or Regression Testing) that will be covered in this test plan.
2. Planning	

Component	Description
2.1 Schedule	This section contains the schedule information. Include milestone dates. This information could be in the form of a work breakdown structure.
2.2 Environmental Needs	This section contains the hardware details, environment location, and test facilities that support the software test effort and any technical need that Federal Student Aid may request from a contractor.
2.2.1 Test Environment	
2.2.2 Testing Tools	This section contains details about the testing tools that will be used and how they will be used.
2.2.3 References	This section identifies the reference material that will be used or has been used in order to support the test effort.
2.3 Roles and Responsibilities	This section identifies the roles and responsibilities of Federal Student Aid and Contractor for system testing. Identify individuals by name, role and list the timeframe of their participation in the User Acceptance test effort.
2.4 Staffing and Training Needs	This section consists of the following items: • Number of resources and skills required for the testing project • Identification of areas where training is required and a plan for training testers • Include a timeframe for training
3. Requirements Phase	This section describes the Federal Student Aid test team's involvement in the requirements phase of the project. This will include checking testability of requirements, review of functional specifications, and detail design and analysis of the requirements tracking matrix (RTM).
4. Review of Contractor Results	This section describes the Federal Student Aid test team's process for reviewing the contractor's System Test results.
5. Test Configurations	
5.1 Features to be Tested	This section describes the list of features (Functions or Requirements), which will be tested or demonstrated by the test. The description is provided at a high level without going into the technical details.
5.2 Features Not to be Tested	This section describes the list of features (Functions or Requirements), that will not be tested.
5.3 Components to be Tested	This section identifies and lists the components to be tested within the scope of this test plan.

Component	Description
6. Test Approach	This section includes details of testing covered by the plan and the objectives, scope, rationale and communication plan. For Agile projects, state whether this testing covers a single sprint or a group of sprints.
6.1 Test Strategy	This section includes the test strategy.
6.1.1 Test Process	This section provides a brief synopsis of what it takes to test. Provide details of testing covered by the plan and the objectives, scope, and rationale.
6.1.2 Assumptions and Constraints	This section identifies the known assumptions and constraints that feed into the test strategy. If needed, list mitigation strategies.
6.1.3 Test Work Products	This section identifies and lists the documents that will be created to support the test effort. State the purpose of each document, list names or teams, create and review the document.
6.1.3.1 Test Artifact Details	This section identifies the process for creating test suites. If the contractor is to provide the test suites to Federal Student Aid, then this document must indicate this and include the process for providing the contractor information for developing the test suites, the approval process and the review process of the test suites.
6.1.4 Regression Strategy	This section covers how regression testing will be performed and the regression cycles.
6.2 Traceability	This section describes how test documentation will trace back to the requirements that are under test.
6.3 Test Data	This section covers how test data will be created and identified for the test effort.
6.4 Test Estimation	This section will describe the test estimation technique that will be followed for this test effort. The details should cover estimation for the test effort and how estimates will be determined related to regressing defects.
6.5 Test Readiness Review	This section contains a test readiness review process. Include a reference to the Test Readiness Review document, which must be an appendix to this User Acceptance Test Plan.
6.6 Pass/Fail Criteria	This section contains the conditions of the pass/fail criteria.
6.7 Entry and Exit Criteria	This section contains the criteria that determine when testing can begin and when the testing process can be stopped.

Component	Description
6.8 Xxx System Approach and Techniques	This section covers minor exceptions pertaining to a particular subsystem of an application. If the exceptions are significant a separate test plan must be created for the subsystem. The contractor and the Federal Student Aid Test Manager and Project Manager will make this decision in the beginning phases of the project.
7. Defect Management	This section describes the defect management process that will be used for this project. Include information on documenting issues, defect classification and change control process. Include the name of the defect management tool. Include how the contractor will be involved in the defect management process.
7.1 Documenting Issues	This section describes the process for documenting defects. This must describe when an issue is added to the defect tracking tool, meetings that must be held to discuss issues and making the determination of when an issue becomes a defect.
7.2 Defect Classification	This section describes the different states used for classifying defects. Describe in detail the severity, complexity, status and priority classification.
7.3 Change Request	This section describes the process for handling change requests. If this process is detailed in another project document, state the name of the document and provide an abbreviated version of the process.
7.4 Suspension Criteria and Resumption Requirements	This section lists the known conditions that may result in the suspension of testing a function of the system under test. Include the process for determining the resumption of the test. For example, if an application is under test and the login feature does not work, testing activity shall be suspended and shall resume only when the login feature has been corrected and a tester has approved that the login feature meets expectations.
8. Software Risk	
8.1 Risk and Mitigation Strategy	This section details known risk and mitigation strategies for each risk.
9. Lessons Learned	This section details the lessons learned process for this phase of testing.
10. Approvals	This section describes the approval process and lists the job titles (and names of current incumbents) that must approve the plan. List those that have sign-off responsibility (required), those that are accountable, those that were consulted, and those that were informed.

Component	Description
Appendices	Instructions for completing the appendices can be found in the Template Formatting Instructions. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Template Document ➔	Please contact the Enterprise Test Team for the Word document.

5. User Acceptance Test Support Plan Template

Final plans should follow the formatting guidelines set forth in the Template Formatting Instructions.

Instructions

Component	Description
Cover	A sample of the cover sheet is provided in the template.
User Acceptance Test Support Plan Form	Complete form.
Document Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
1. Overview	This section will give a brief synopsis of the background of the project under test. State the purpose of the User Acceptance Test Support Plan and state the sub phases (Functional testing and/or Intersystem Testing and/or Regression Testing) that would be covered in this test plan.
2. Schedule	This section will include the timeline of the test. It is acceptable to have a link to the Work Breakdown Structure.
3. Test Environment Requirements	This section will list the hardware and work space required and that will be provided by the contractor during User Acceptance Testing.
4. Responsibilities	This section will list the responsibilities of the contractor during User Acceptance Testing. This section will also include responsibilities that Federal Student Aid has to the contractor during User Acceptance Testing.

Component	Description
4.1 Setup	This section will list the tasks that the contractor will perform to prepare the User Acceptance Testing environment. This may include creation of test suites, tracking tools, etc.
4.2 Execution Support	This section will list the support that the contractor will provide during User Acceptance Testing. This may include defect resolution, meeting attendance, etc.
5. Staffing and Training Needs	This section will list the training that the contractor will provide to Federal Student Aid.
6. Test Configurations	
6.1 Features to be tested	This section describes the list of features (Functions or Requirements), which will be tested or demonstrated by the test. The description is given at a high level without going into the technical details. State that these are the features that Federal Student Aid will approve.
6.2 Features not to be tested	This section describes the list of features (Functions or Requirements) which will be tested or demonstrated by the test. The description is given at a high level without going into the technical details. State that these are the features that Federal Student Aid will not test during User Acceptance Testing.
7. Test Approach	This section will include the approach that will be used. For example, state where the testing will take place, and the access that Federal Student Aid will have to the contractor that is supporting the test effort. State level of participation in the User Acceptance Testing Test Readiness Review process. State the services that the contractor will provide to Federal Student Aid during User Acceptance Testing.
7.1 Pass/Fail Criteria	This section contains the conditions of determining pass/fail criteria during User Acceptance Testing.
8. Suspension Criteria and Resumption Requirements	This section will list the support the contractor will provide to Federal Student Aid in cases where User Acceptance Testing is suspended due to unforeseen suspension of the test process.
9. Test Work Products	This section will list the work products that will be created by the contractor during User Acceptance Testing.
10. Tasks	This section will list the tasks that the contractor will perform during User Acceptance Testing.
11. Data Request	This section will include how data request will be delivered to Federal Student Aid for User Acceptance Testing. Include a data request template in the appendix of this document and provide this template to Federal Student Aid as early as possible and no later than during the beginning phases of System Testing.

Component	Description
12. Software Risk	This section will detail known risk (under contractor control) that may occur during User Acceptance Testing and mitigation strategies for each risk.
13. Test Readiness Criteria	This section will include Test Readiness Review information such as what the TRR will ensure prior to User Acceptance Testing.
14. Test Completion Criteria	This section will include information on how the contractor and Federal Student Aid determine when User Acceptance Testing is complete.
15. Approvals	This section will outline the approval process and list the job titles (and names of current incumbents) that initially must approve the plan, and sign off on the plan's satisfactory execution. List those that have sign-off responsibility (required), those that are accountable, those that were consulted and those that were informed.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions*. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Template Document ➔	Please contact the Enterprise Test Team for the Word document.

6. Post Implementation Verification Test Plan Template

Final plans should follow the formatting guidelines set forth in the Template Formatting Instructions.

Instructions

Component	Description
Cover	A sample of the cover sheet is provided in the template.
Post Implementation Verification Form	Complete form.
Document Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
1. Overview	This section will give a brief synopsis of the background of the project under test. State the purpose of the Post Implementation Verification and state the sub phases (Functional testing and/or Intersystem Testing and/or Regression Testing) that would be covered in this test plan.
2. Planning	
2.1 Schedule	This section contains the schedule information. Include milestone dates. This information could be in the form of a work breakdown structure.
2.2 Roles and Responsibilities	This section will identify the roles and responsibilities of Federal Student Aid and Contractor for system testing.
3. Critical Functional Areas to Be Tested	This section lists the critical features (Functions or Requirements), which will be analyzed during their first cycle in the production environment.
4. Verification of Critical Features	This section describes how each critical feature will be verified and how the feature will be deemed successful.
5. Reporting Results to Federal Student Aid	This section will include details of how test results will be communicated to Federal Student Aid. Must include how major failures will be communicated to Federal Student Aid.

Component	Description
6. Post Implementation Document	This section will include a document for each item under test. The information may be included in an EXCEL spreadsheet. At a minimum the fields included must be: • Verification Item • Verification Method • Verification Timeframe • Review Date • Expected Result • Actual Result • Pass/Fail • Resource that will be verifying item • Comments • Supporting Information • Contractor Sign-off • Federal Student Aid Sign-off
7. Approvals	This section will outline the approval process and list the job titles (and names of current incumbents) that initially must approve the plan, and sign off on the plan's satisfactory execution. List those that have sign-off responsibility (required), those that are accountable, those that were consulted and those that were informed.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions*. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Template Document ➔	Please contact the Enterprise Test Team for the Word document.

7. Test Summary Report Template

Final reports should follow the formatting guidelines set forth in the Template Formatting Instructions.

Instructions

Component	Description
Cover Sheet	A sample of the cover sheet is provided in the template.
Document Version Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Test Summary Report Form	Complete form.
Table of Contents	A sample of the table of contents is provided in the template.
Executive Summary	This section provides a brief summary of the project, critical requirements and the findings of the test effort. This summary must be no longer than 1½ pages.
1.Purpose of the Test Summary	This section describes the purpose of this document.
2. Test Items	This section contains a high-level description of the items under test.
2.1 Reference Material	This section contains the documents that were used for the test effort. Such documentation includes requirements, test suites, test results, data, etc. Provide the document name, state if the document is available, and describe how to obtain access to it.
2.1.1 Environment	This section contains details about the environment that was used for the test effort.
3. Variances	This section describes the variances and reasons for variances that occurred during the test effort.
3.1.1 Test Dates	This section describes variances in the baseline test start dates and end dates.
3.1.2. Outages	This section describes the outages (planned or not planned) that occurred during the test effort that may have caused a delay or workarounds that were not planned.
3.1.3 Others	This section describes each type of variance.
4. Comprehensive Assessment	This section includes an evaluation of the testing and test process in terms of the documented test objectives.
5. Summary of Results	This section describes the overall status of defects. It includes the total number of defects, severity, priority, cost, defect patterns, and any open or unresolved defects. The focus should include positive and negative trends that occurred during the test.

Component	Description
6. Evaluation	This section provides a total evaluation of the test effort. This section should assess the quality of the software. Include both positive and negative results.
7. Summary of Activities	This section includes the planned activities and the changes to the plans. Include reasons for deviations and the impact on the test effort.
8. Approvals	This section lists those responsible for reviewing and approving this document. The approvers must be the same ones that initially approved the test plan associated with the test summary report.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions*. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Template Document ➔	Please contact the Enterprise Test Team for the Word document.

8. Performance Test Plan Template

Final plans should follow the formatting guidelines set forth in the Template Formatting Instructions. All sections outlined below are required. However, additional sections may be included based on the type of system being tested.

Instructions

Component	Description
Cover Sheet	A sample of the cover sheet is provided in the template.
Document Version Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
Section 1. Overview	High-level summary of the application.
Section 2. Introduction	High level view of the application and testing approach.
Section 3. Test Items	The items of software, hardware, and combinations of these that will be tested.
Section 4. Features to be Tested	A list of which functions/features of the application will be tested.
Section 5. Features Not to be Tested	A description of which functions/features of the application will not be tested.
Section 6. Approach	The process used for testing.
Section 7. Item Pass/Fail Criteria	Lists the explicit Pass/Fail criteria for each test run.
Section 8. Suspension Criteria and Resumption Requirements	The circumstances that would cause testing to stop and restart.
Section 9. Test Deliverables	Test documents and other deliverables which will be produced. Deliverables are: Test Plan Daily Test Reports (summary E-mail delivered within 24 hours following each test session) Test Report
Section 10. Testing Tasks	List of tasks, their dependencies, the elapsed time to complete, and resources required.
Section 11. Environmental Needs	What is needed in the way of testing software, hardware, etc.
Section 12. Responsibilities	List of who is responsible for delivering the various parts of the plan.

Component	Description
Section 13. Staffing and Training Needs	The people and skills needed to deliver the plan.
Section 14. Schedule	When testing will occur.
Section 15. Risks and Contingencies	This defines all other risk events, their likelihood, impact and counter measures.
Section 16. Approvals	The signatures of the various stakeholders to show they agree in advance with the plan.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions*. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Template Document ➔	Please contact the Enterprise Test Team for the Word document.

9. Performance Testing Test Report Template

Final reports should follow the formatting guidelines set forth in the Template Formatting Instructions. Sections in this template may be added or removed based on the type of system being tested.

Instructions

Component	Description
Cover Sheet	A sample of the cover sheet is provided in the template.
Document Version Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
Executive Summary	High-level summary of the application.
1.Performance Test Results	This section contains a summary of the major findings during the performance testing.
1.1 Issues	
1.1.1 Open Issues	A list of the open issues.
1.1.2 Resolution	A brief description of the issues and how they were resolved.
1.2 Static Tests	
1.3 Peak Tests	
1.3.1 XXX Peak	
1.3.2 XXX Peak	
2. Capacity Planning and Configuration	***Note:*** *Capacity Planning is for the production system and not for the performance test environment.*
2.1 Capacity Planning	Details of the planning (e.g., the number of servers required to support the load and the number of processes running on servers.)
2.2 Configuration	Details of system architecture of the performance test environment. If available, provide a diagram of the environment.
4. Production Performance Readiness	A determination of the overall evaluation of the readiness for the application to run in production based on the Performance Test Results. This is a recommendation provided by the Performance Test Team to the Application Owner.

Component	Description
Appendices	Instructions for completing the appendices can be found in the Template Formatting Instructions. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Template Document ➔	Please contact the Enterprise Test Team for the Word document.

10. Hardware Test Plan Template

Final plans should follow the formatting guidelines set forth in the Template Formatting Instructions.

Instructions

Component	Description
Cover Sheet	A sample of the cover sheet is provided in the template.
Test Plan Certification	Provide information about the project approvals.
Document Version Control	Provide information concerning the version of the document, the date of the change, and a description of the change.
Table of Contents	A sample of the table of contents is provided in the template.
Section 1. Overview	This section will give a brief synopsis of the background of the project under test. State the purpose of the Hardware Test Plan.
Section 2. Planning	
2.1 Schedule	This section contains the schedule information. Include milestone dates. This information could be in the form of a work breakdown structure.
2.2 Environmental Needs	This section contains the hardware details, environment location, test facilities that support the software test effort.
2.2.1 Test Environment	
2.2.2 Testing Tools	This section contains details about the testing tools that will be used and how they will be used.
2.2.3 References	This section will identify the reference material that will be used or has been used in order to support the test effort.

Component	Description
2.3 Roles and Responsibilities	This section will identify the roles and responsibilities of Federal Student Aid and the Contractor for system testing.
2.4 Staffing and Training Needs	This section consists of the following items: • Number of resources and skills required for the testing project • Identification of areas where training is required and details the plan for training testers (if applicable include training of Federal Student Aid testers) • Include timeframe for training
Section 3. Test Configurations	
3.1 Required Features	This section describes the list of features, which will be tested or demonstrated by the test.
3.2 Optional Features	This section describes the list of hardware features that will not be tested. Only list those that some would naturally believe would be tested given the nature of the project.
Section 4. Test Approach	This section will include details of testing covered by the plan and the objectives, scope, rationale and communication plan.
4.1 Test Strategy	This section will include the test strategy, which includes the following: • Process • Assumptions • Work Products • Test data needed if any to test hardware • Types of reports and metrics • Test case writing process • Peer Review Process • Configuration Management Process • Handling of meetings and other Organizational Processes • Add other information as needed
4.1.1 Test Process	This section gives a brief synopsis of what it takes to test. Provide details of testing covered by the plan and the objectives, scope and rationale.
4.1.2 Assumptions and Constraints	This section will identify the known assumptions and constraints that feed into the test strategy. If needed list mitigation strategies.

Component	Description
4.1.3 Test Work Products	This section will identify and list the documents that will be created to support the test effort. State the purpose of each document that will be created. List names or team expected to review each document. This section will also include information on how the test results will be delivered to Federal Student Aid for approval purposes.
4.1.4 Traceability	This section covers how test documentation traces back to the hardware functions that are under test.
4.1.5 Test Data	This section covers how test data if needed will be created and identified for the test effort.
4.2 Test Estimation	This section will describe the test estimation technique that will be followed for this test effort. The details should cover estimation for the test effort and how estimates will be determined related to regressing defects.
4.3 Test Readiness Review	This section contains a test readiness review process. Include a reference to the Test Readiness Review document, which is found in Appendix C.
4.4 Pass/Fail Criteria	This section contains the conditions for the pass/fail criteria.
4.5 Entry and Exit Criteria	This section contains the criteria that determines when testing can begin and when the testing process can be stopped.
4.5.1 Service Level Agreement	This section covers the service level agreement for the application under test. Include tie to exit criteria. Example: If testing voice response system how many responses must be recognized before Federal Student Aid will accept the product.
Section 5. Defect Management	This section will describe the defect management process that will be used for this project. Include information on documenting issues, defect classification and change control process. Include the name of the defect management tool. Include how Federal Student Aid will be involved in the defect management process.
5.1 Documenting Issues	This section will describe the process for documenting defects. This must include when an issue is added to the defect tracking tool, meetings that must be held to discuss issues and making the determination of when an issue becomes a defect.
5.2 Defect Classification	This section will describe the different states used for classifying defects. Describe in detail the severity, complexity, status and priority classification.

Component	Description
5.3 Change Request	This section will describe the process for handling change requests. If this process is detailed in another project document state the name of the document and provide an abbreviated version of the process.
5.4 Suspension Criteria and Resumption Requirements	This section will list the known conditions that may result in the suspension of testing a function of the hardware under test. Include the process for determining the resumption of the test. For example, if the hardware under test does not work and causes all other testing to fail, testing activity will be suspended and will resume only when the hardware functionality meets expectations.
Section 6. Risk	
6.1 Risk and Mitigation Strategy	This section will detail known risk and mitigation strategies for each risk.
Section 7. Lessons Learned	This section will detail the lessons learned process for this phase of testing.
Section 8. Approvals	This section will outline the approval process and list the job titles and names of current incumbents that must approve the plan, and sign off on the plans satisfactory execution. List those that have sign-off responsibility (required), those that are accountable, those that were consulted and those that were informed.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions*. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Template Document ➔	Please contact the Enterprise Test Team for the Word document.

11. Other Templates

Other Templates	Description
12.	• Performance Test Daily Reports
14.	• Defect Severity and Current State
17.	• High Level Test Summary
18.	• Incident Report By Testing Phase
19.	• Disaster Recovery Test Readiness Review Document

Please contact the Enterprise Test Team for the other template documents.