State of Michigan

(Insert Project, Program or Organization Name Here)

Test Strategy

# General Information

|  |  |  |  |
| --- | --- | --- | --- |
| System or Project ID/Acronym: |  | Creation Date: |  |
| Client Agency: |  | Modification Date: |  |
| Author(s): |  | DTMB Authorized by: |  |

# Privacy Information

This document may contain information of a sensitive nature. This information should not be given to persons other than those who are involved with this system/project or who will become involved during its lifecycle.

# Revision History

The Project Manager will maintain this information and provide updates as required. All updates to the Project Management Plan and component plans should be documented in this section.

| Revision Date | Author | Section(s) | Summary |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

# Overview

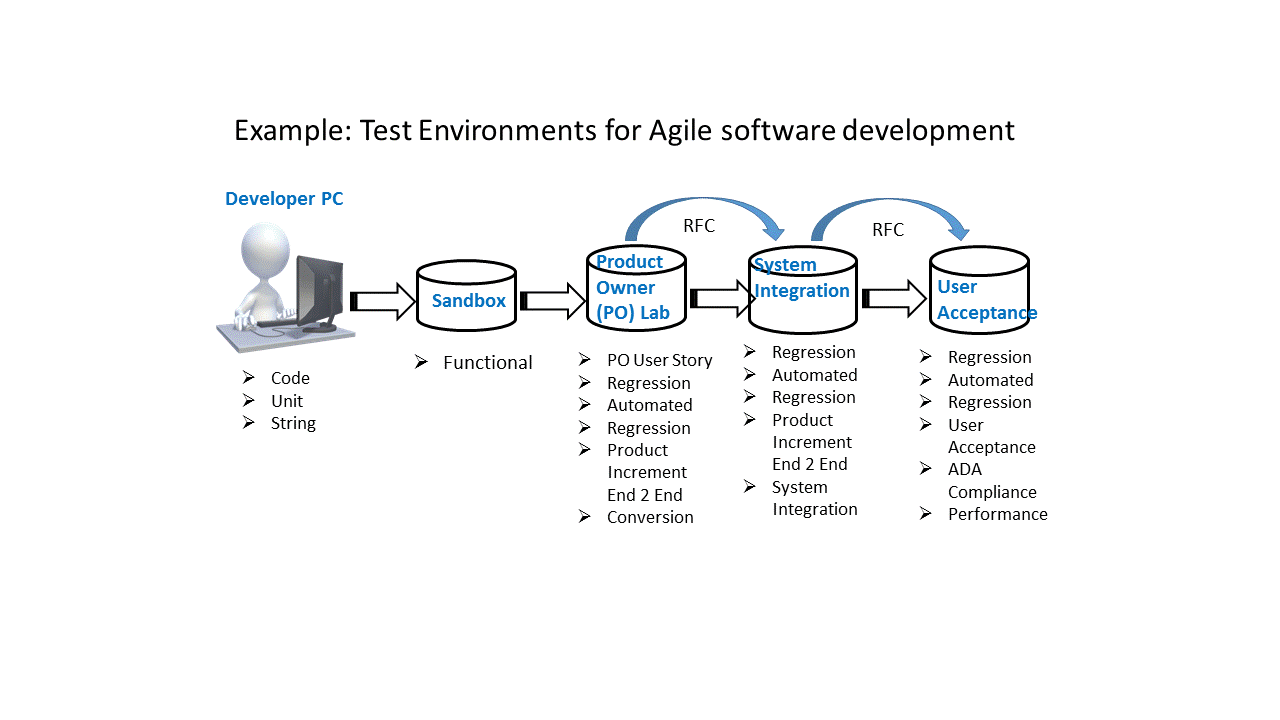
The purpose of the Test Strategy is to communicate the overall testing approach to all appropriate stakeholders (project team, business owners, sponsors and management).  This document identifies the types of tests that will be performed, if any tools are needed to execute the tests, when each of those tests are performed, who is responsible for those tests and how the results of those tests are to be recorded and communicated.  By identifying and defining these testing assets earlier in the process, it should assist with budget and resource estimations/allocations. This document may pertain to a single project, a program or an organization.

Definitions for test types and test roles can be found in Appendix A.

# Testing Strategy

## Test Environments

This section describes number, name, purpose, and location for the test environment landscape. Include a diagram of the test environments, location and relationship (how, when code moves from one environment to the next). Replace the example with a diagram of the projects or organizations’ environment.



**<Insert diagram or a table of test environments with a description for each environment>**

## Testing Tools

This section describes the approach being used for testing (manual and automated), traceability, and test metric reporting). The table listed here has examples of various tools and is for reference. If the organization does not have a standard tool identified check the Architectural Roadmap before making a selection. If the tool selected is not listed here, insert the name of the tool selected.

|  |  |  |
| --- | --- | --- |
| **Tool Function/Purpose** | **Suggested Tools (currently on the Architectural Roadmap** | **Cost** |
| Test Planning,  Test Cases,  User Story or Requirement to Test Case traceability,  Test execution reporting, etc. | Team Foundation Server | Yes |
| Test Planning,  Test Cases,  User Story or Requirement to Test Case traceability,  Test execution reporting, etc. | HP ALM | Yes |
| Test Planning,  Test Cases,  User Story or Requirement to Test Case traceability,  Test execution reporting, etc. | Rational Quality Manager | Yes |
| Test Planning,  Test Cases,  User Story or Requirement to Test Case traceability,  Test execution reporting, etc. | SUITE Templates | No |
| Test Automation | WorkSoft | Yes |
| Test Automation | HP UFT | Yes |
| Test Automation | Selenium | No |
| Test Automation | Rational Functional Tester, |  |
| Test Automation | MS Test Manager | Yes |
| Performance Testing | Rational Performance Test | Yes |
| Performance Testing | HP Load Runner | Yes |
| Performance Testing | DynaTrace | Yes |
| Performance Testing | Team Foundation Server | Yes |
| Virtualization | Rational (RTVS) | Yes |
| ADA Compliance (eMich) | AccVerify, JAWS | Free - limited functionality  Yes - full functionality |
| **<Other>** |  |  |

**<Indicate the Tool/SUITE Templates that will be used>**

|  |  |
| --- | --- |
| **Tool Function/Purpose** | **Suggested Tools (currently on the Architectural Roadmap** |
| Test Planning, | Team Foundation Server, HP ALM, Rational Quality Manager, SUITE Templates |
| Test Case or User Story creation | Team Foundation Server, HP ALM, Rational Quality Manager, SUITE Templates |
| Requirement to Test traceability, | Team Foundation Server, HP ALM, Rational Quality Manager, SUITE Templates |
| Test Metrics reporting | Team Foundation Server, HP ALM, Rational Quality Manager, SUITE Templates |
| Test Automation | WorkSoft, HP-UFT, Selenium, Rational Functional Tester, MS Test Manager |
| Performance Testing | Rational Performance Test, HP Load Runner, DynaTrace, Team Foundation Server |
| ADA Compliance | AccVerify, JAWS, eMichigan |
| **<Other>** |  |

## Test Data Management

List the approach that will be used to create and maintain the test data.

* Approach for creating test data (programs, copy production etc.)
* Frequency/approach for refreshing test data
* Requirements for data masking when using copies of production test data
* Approach for creating conversion data
* Other information needed to manage test data for the project

**<Describe the approach to the creation and maintenance of test and conversion data>**

## Roles and Responsibility

Select from the appendix all the test roles planned for the project. List the responsibilities of each role. If the project has identified a test role not identified in the Test Roles table, add it to the table below.

|  |  |  |
| --- | --- | --- |
| **Tester Role** | **Tester Responsibility** | **Rate** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Estimated Resources

Based on the phases or number of deploy product increments test cycles that are planned complete the table below to account for the resource that will be needed for testing activities.

| **Phase/Product Increment** | **Type of resource needed for the test phase or product increment (i.e., Test Analyst, Developer, Business Analyst, Product Owner, End Users, etc.)** | **Estimated number of resources needed (by type) \*** | **Percent allocation** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

\* Consult Test Center of Excellence or Test Manager for an industry guideline on the ratio of developers to testers.

## Metrics & Reports

Projects need to provide both test execution and defect metrics throughout each test phase or deploy product increment test cycle.

Projects should complete the following table to understand and plan for collecting and reporting the metrics associated to test execution, defect logging, defect correction, etc.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metric** | **What Phase / Product Increment will be collected in:** | **Reporting Frequency** | **Source of Truth** | **Audience** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Test Automation

Describe how test automation will be addressed in each phase/product increment of the project (refer to the Test Type Detail section below for suggestions).

## Critical Success factors

Describe the activities or elements that are necessary for the test phases to be successful.

Examples may include:

* Proper Staffing of qualified test resources
* Enablement of Automation tools
* Availability of stable test environments
* Proper scheduling for test planning and execution activities

# Test Type Detail

The information contained in this table represents industry best practices. Projects and organizations must decide which test types will be included in the project or organizational test strategy and if any test types must be customized to fit the project or organization’s needs.

The tables are separated by development methodology, agile or waterfall. Remove the section(s) that do not apply based on the project approach or organizations default development methodologies. Modify remaining sections to reflect the test activities of the project and/or organization.

Sprint Testing / Product Increment Testing (Agile)

| **Test Type** | **Definition** | **Entry Criteria** | **Minimum Exit Criteria** | **Frequency** | **Test Creator** | **Test Executor** | **Environment** | **Associated Artifact** | **Location of Artifacts** | **Strategy for issues found during stage** | **Record of issues** | **Metric Reported** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Unit Test** | **Verification (Test it was built right)**  Unit Test is the process of testing individual units of functionality. Unit tests verify that individual system components support the related functional, non-functional (technical), and interface requirements as represented in the User Story | Work has begun on a user story | Unit Test Code Coverage goal **<xx %>** | Iterative during the sprint | Developer | Developer | Developer PC | Unit test case | In code repository | Problems resolved within sprint | None | %Test Code Coverage |
| **Functional User Story Test** | **Verification (Test it was built right)** - Functional testing is the process of testing each function of the user story in accordance with their respective user story acceptance criteria. | - All code to satisfy the user story has been completed and tested  - **100%** of Unit Test cases passed  - Unit Test Code Coverage goal met | All documented acceptance criteria met. | Iterative during the sprint. | User Story Developer/  Tester/  Business Analyst | Tester/  Business Analyst/  Alternate Developer | Development Sandbox (AKA Dev Test) | Written Test Case of the User Story (automate when possible). | In Test Case Repository | All identified problems resolved within sprint before passing to PO | None | N/A |
| **Product Owner (PO)** Functional User Story Acceptance Test | **Verification - (Test it was built right)** Functional testing is each function of the user story in accordance with their respective user story acceptance criteria by the PO. | **-** 100% Documented **Functional** Test Cases passed | - PO executes all documented User Story test cases for all features developed within the sprint - **100%** Documented test cases pass | Iterative during the sprint | Product Owner/ Business Analyst | Product Owner / End User Subject Matter Expert (SME) | PO Lab | Written Test Case of the User Story (automate when possible) | In Test Case Repository | Problems resolved within sprint, before Sprint Review/Demo | None | N/A |
| **Sprint Demo (Review)** | **Verification - (Test it was built right)** Final review by the PO of thefunctionality of each user story in accordance with the user story acceptance criteria  **Validates -** that it meets the needs of the end users | User Story has successfully passed:  - Unit Test  - Functional Test  - PO User Story Acceptance Test | PO approves/accepts stories in the automated Backlog Tool  or  - e-mail sent and acceptance received and stored  All feedback items entered Product Backlog as new user stories | Once per sprint | Scrum Team demonstrates completed work | Product Owner and invited observers | Sandbox or PO Lab | User Story and User story acceptance criteria reviewed  Solution presented for review | Automated Backlog tool; Project repository | Defects logged as new defect stories in Product Backlog Product Backlog Item (PBI)  Feedback logged as a new user story in the Product Backlog (new PBI) | Accepted - Product Owner updates status in the tool to "accepted" (or project equivalent)  or responds to e-mail | 4UP Status Report – metrics reported  - Planned  - Accepted / Approved  - Moved to the next sprint (not complete or defect encountered)  Report in story points or number of stories |
| **Regression Test** | **Verification** - Regression testing ensures that existing functionality is not broken with the introduction of new code. Both new and presumably unchanged functionality is executed and validated.  if automated – run daily | **Manual –** Product Increment build complete | **100%** of expected results achieved | **Manual** – Once per Deploy Product Increment (repeated until all agreed to defects corrected)  **Automated** - **once per day** (generally during non-working hours) and at Deploy Product Increment | Product Owner / Business Analyst / Test Analyst/ Automation Engineer | Automation Engineer/Product Owner / End User SME (may vary depending on what is being tested) | PO Lab | All PO user story test for User Stories accepted to date (manual or automated) | Test Tool Repository or  Excel Test Case template  or  Suite Test Case Template | **Manual** – Agreed upon defects resolved before deploying code  **Automated** –defect logged as defect PBI, and defects fixed before next test cycle (when not fixed before next test cycle the test removed from automated script | **Defect** is logged as new defectPBI | **Manual** on the deploy product increment (DPI) Closure report  **Automated** on 4 Up Status Report with sprint metrics  -Number of regression test executions  Number of regression test executions that failed  Number of defects recorded |
| **Product Increment (PI) (End 2 End)**  When the product increment is **NOT** being deployed to production  **System Integration Test (SIT), User Acceptance Test (UAT)**  When the product is being deployed to production | **Verification (Test it was built right) -**  End 2 End and/or System Integration Test **(SIT)** is the process of testing the complete application in a fully integrated testing environment that mimics the real-world use, meeting business scenarios including the interaction with external systems.  User Acceptance Test **(UAT)** is the process of software users testing the software to make sure it can handle required tasks in real-world scenarios, according to User Story specification | **- All** sprints for the Product Increment are complete **- All** user stories for the Product Increment – accepted by the PO  **- 100 %** automated regression tests passed  **-** Unit Test Code Coverage for the PI **(> xx %)** | **All** (PI E 2 E) test scenarios passed and  **no**  Critical / High defects are open  Additional **SIT criteria - All integration test cases** passed and **no** Critical / High defects are open  Optionally PO may require all medium / low defects resolved before completion | **At the end of a Product Increment**  (Repeat testing if defects need remediating) | Product Owner or  End User SME  Maybe be assisted by Tester /  Business Analyst | Product Owner or  End User SME | PO Lab or SIT Environment | Business Test Scenarios for the Product Increment  Regression Tests  **SIT** – additional  SIT Test Cases | Testing Tool Repository or  Excel Test Case template  or  Suite Test Case Template | Defects assessed for impact Critical / High / Medium / Low (optionally) fixed during testing phase | Defects Logged as new defect PBI’s | Total Tests Executed  Total Tests Passed  Total Defects Recorded  Total Defects Corrected |
| **User Acceptance Test (UAT) -** when deploying PItoproduction  **May be a final UAT phase of testing after the final PI is complete** | **Verification (Test it was built right) and**  **Validation (We built the right thing)**  User Acceptance Test **(UAT)** is the process of software users testing the software to make sure it can handle required tasks in real-world scenarios, according to User Story specification | **- All** sprints for the Product Increment are complete **- All** End 2 End testing complete (if separate phase of testing) **- 100%** of Unit Test cases passed **- 100 %** automated regression tests passed  **-** Unit Test Code Coverage for the Product **(> xx %)** | **All SIT** test scenarios passed  **and**  **No** Critical / High / Medium Defects outstanding | When deploying a Product Increment to **production**  (Repeat testing if critical or high defects needed remediating) | Product Owner or Business SME  Maybe be assisted by Tester /  Business Analyst | Product Owner or  End User SME | UAT Environment | UAT Test Cases  Automated Regression Test scripts  UAT Test Cases (optional) | Testing Tool Repository  Or  SUITE templates  Or  Project’s Excel Test Case template | **Defects: Critical / High** must have fixed during testing period  **Medium / Low** maybe fixed during testing period  **or** Added to the Product Backlog as defect stories for resolution in future sprints |  | Total Tests Executed  Total Tests Passed  Total Defects Recorded  Total Defects Corrected |
| Accessibility **(ADA Compliance)** prior to deploy to production  May perform **preliminary** review of ADAcompliance by checking compliance during the build process | **Validation (Built the right thing)) -**  ADA Compliance states that electronic and information technology must be accessible to people with disabilities in accordance with Americans with Disabilities Act (ADA) Standards published by DOJ. | **- Product has on-line /mobile features**  **- PI** is being deployed to **production**  **- UAT complete** and all prioritized defects remediated | **- All ADA** compliance issuers addressed **or**  **- written** authorization from PO to deploy with known issues | When deploying to **production**  (Repeat testing if critical or high defects needed remediating | **e-Michigan** Team  (PM / BA coordinates) | e-Michigan | UAT Environment | ADA Compliance standards | **e-Michigan** Test tools | Out of compliance test results must be remediated  **or**  **- written** authorization from Product Owner or Sponsor to deploy with known issues | Defects logged as new defect PBI’s | ADA Compliance Testing:  Total Tests Executed  Total Tests Passed  Total Defects Recorded  Total Defects Corrected |
| **Conversion** | **Verification (Test it was built right) -**  The data conversion programs populate the new databases correctly | **- All** sprints building conversion features have completed  **- 100%** of User Story Test cases passed | **- All** conversion test scenarios passed | Iterative until all conversion test pass | Product Owner / End User SME  Maybe be assisted by Tester /  Business Analyst | Product Owner / End User SME  Maybe be assisted by Tester /  Business Analyst | Conversion  or  UAT Environment | Conversion User Stories / Test Cases / Scenarios | Testing Tool Repository  Or  SUITE templates  Or  Project’s Excel Test Case template | **All Defects** remediated  Or  Workaround documented | Defects logged as new defect PBI’s | Conversion: Total Tests Executed  Total Tests Passed  Total Defects Recorded  Total Defects Corrected  **Others as defined in the Detail Test Plan** |
| **Performance Test**  **(note:** Only changes to address performance issues can be made to the environment during this test phase**)** | **Validation (We built the right thing)**  A type of non-functional test whose purpose is to identify and fix system performance issues before the system goes live. | **-** End 2 End testing /SIT /UAT has been started and system code base is considered to be in a ‘stable’ state  - Load volumes have been documented | **Performance meets or exceeds**  documented non-functional requirement for response time | **Once** before release to production  (Repeat until all performance goals met) | **Performance Test Specialist**  **Or**  Technical Lead  **Or**  Automation Engineer | **Performance Test Specialist**  **Or**  Technical Lead  **Or**  Automation Engineer | **UAT** (mirror of Production) | Performance Test objectives and performance test scripts | Automated Performance Testing Tool repository  Or  Application code repository | **All non-conformances** remediated  Or  **- written** authorization from Product Owner to deploy with known issues | Defects logged as new defect PBI’s | Performance:  Total Tests Executed  Total Tests Passed  Total Defects Recorded  Total Defects Corrected  **Others as defined in the Detail Test Plan** |
| **Post Deploy Validation** | **Verification & Validation** of deployed functionality | RFC designated Deploy to Production  and Data Conversions (if planned) have been completed | Product Owner and designated testers have concluded the system is ready for use by the intended audience | **Once** | **Product Owner** |  | **Production** | Reuse selected UAT Test Scenarios and cases | Test Case Repository | If issues can be resolved immediately – correct issues  Or  Execute back out procedures | Update RFC as appropriate | Deploy:  Success  Or  Backed out |
| **Add others as needed** |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

Phased Based Testing (Waterfall)

| **Test Type** | **Definition** | **Entry Criteria** | **Minimum Exit Criteria** | **When and How Testing Occurs** | **Test Creator** | **Test Executor** | **Environment** | **Associated Artifact** | **Location of Artifacts** | **Strategy for issues found during stage** | **Record of issues** | **Metric Reported** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Unit Test** | **Verification (Test it was built right)**  the developed component works correctly from the developer’s perspective. | Technical design documents are completed.  Component is developed | **• 100%** of Unit Test cases passed | Testing occurs as components are completed. | Developer | Developer | Developer PC | Unit test case | If automated, Testing Tool Repository | Problem | None | None |
| **String Test** | String Testing is conducted at the sub process level to ensure the process tasks and transactions of the sub process interact correctly. It is typically an exhaustive test focusing on the system functions within the context of a business process.  Successful completion of String Test efforts establishes readiness for Integration Testing. | • **100%** of Unit Test cases passed for the component passed  String test plans have been reviewed and approved. | **All** documented acceptance criteria met. | String testing is conducted after all components comprising a complete business function have been developed and have passed unit testing. | • Developer | Developer | DevTest | String Test Plans | Testing Tool Repository  Or  SUITE templates  Or  Project’s Excel Test Case template | Problem | None | Percent Completed |
| **System Integration Test (SIT) -** | System Integration Test (SIT), and/or End 2 End testing, is the process of testing the complete application in a fully integrated testing environment that mimics real world use, meeting business scenarios including the interaction with external systems. The purpose of SIT is to verify that the solution works end to end, as designed. | • **100%** of String Test plans executed  **Test Planning is completed.**  **Environment is ready for testing.**  **Integrating systems are ready for testing.** | **All SIT** test cases executed **and**  **No** Critical/High Defects exist.  Product meets agreed upon quality standards. | System Integration Testing occurs after all string testing has been completed. Typically, all release code is migrated to the QA environment simultaneously to mimic how code will be deployed to production. | Test Lead, Test Analyst | Test Analyst or Business Analysts | QA Environment | SIT Test Cases  Automated and Manual Regression Test scripts  UAT Test Cases (optional) | Testing Tool Repository  Or  SUITE templates  Or  Project’s Excel Test Case template | **Defects: Critical/High** must have fixed during testing period  **Medium / Low** maybe fixed during testing period **or** deferred to a future release | Defect Logs | **Total Test Cases Executed**  **Total Test Cases Passed/Failed**  **Total Defects (often broken down by criticality, business function, etc.)**  **Defect Detection Trend** |
| **User Acceptance Test (UAT) -** | The last phase of the software testing process. During UAT, the software users test the software to make sure it can handle required tasks in real-world scenarios, according to specifications. | • **100%** of System Integration Test Cases executed  UAT Test Planning completed.  UAT Test Environment ready for testing. | All UAT Test Cases passed  and  Product meets agreed upon quality standards | UAT should be the last phase of testing within the release cycle. | Business SMEs, End Users – often assisted by Test Analysts | Business SMEs, End Users, | UAT Environment | UAT Test Cases  Automated and/or manual Regression Test scripts | Testing Tool Repository  Or  SUITE templates  Or  Project’s Excel Test Case template | **Defects: Critical/High** must have fixed during testing period  **Medium / Low** maybe fixed during testing period **or** deferred to a future release | Defect Logs | **Total Test Cases Executed**  **Total Test Cases Passed/Failed**  **Total Defects (often broken down by criticality, business function, etc.)**  **Defect Detection Trend** |
| Accessibility **(ADA Compliance)** prior to deploy to production  May build **preliminary** review of ADAcompliance **into the Business Requirements/Technical Design phases** | **Validation (Built the right thing)) -**  The developed screens have been coded correctly to support the State of Michigan's direction on ADA compliance. | **- Product has on-line /mobile features**  **- SIT complete** and all prioritized defects remediated | **- All ADA** compliance issues addressed **or**  **- written** authorization from Sponsor/Release Owner to deploy with known issues | Prior to deploying to **production**  (Repeat testing if critical or high defects needed remediating | **e-Michigan** Team  (PM / BA coordinates) | e-Michigan | UAT Environment | ADA Compliance standards | **e-Michigan** Test tools | Out of compliance test results must be remediated  **or**  **- written** authorization from Sponsor/Release Owner to deploy with known issues | Defect Log | ADA Compliance Testing:  Total Tests Executed  Total Tests Passed  Total Defects Recorded  Total Defects Corrected |
| **Conversion** | **Conversion Testing validates that** data conversion programs populate the new databases correctly and completely. | **- Conversion test plan has been completed.**  **- System Integration Testing has started.** | **- All** conversion test scenarios passed | Iterative until all conversion test pass | Test Analyst / End User SME/Business Analyst | Test Analyst / End User SME/Business Analyst | Conversion  or  QA  or  UAT Environment | Conversion Test Cases | Testing Tool Repository  Or  SUITE templates  Or  Project’s Excel Test Case template | **All Defects** remediated  Or  Workaround documented and approved | Defect Log | Conversion: Total Tests Executed  Total Tests Passed  Total Defects Recorded  Total Defects Corrected |
| **Performance Test**  **(note:** Only changes to address performance issues can be made to the environment during this test phase**)** | The system performs to the **documented non-functional requirement for response time** | **-** SIT /UAT has been started and system code base is considered to be in a ‘stable’ state  - Load volumes have been documented | **Performance meets or exceeds**  documented non-functional requirement for response time | **Once** before release to production  (Repeat until all performance goals met) | **Performance Test Specialist**  **Or**  **Technical Lead** | **Performance Test Specialist**  **Or**  **Technical Lead** | **Any environment that mirrors production in size and configuration** | Performance Test objectives and performance test scripts | Automated Performance Testing Tool repository  Or  Application code repository | **All non-conformances** remediated  Or  **- written** authorization from Sponsor/Release Owner to deploy with known issues | Defect Log | Performance:  Total Tests Executed  Total Tests Passed  Total Defects Recorded  Total Defects Corrected  **Others as defined in the Detail Test Plan** |
| **Regression Test** | Regression Test detect if any previously working code is no longer working due to emerging code changes. | **Manual –** SIT build complete  **Automated - Automation** Environment is ready for testing. | **100%** of expected results achieved | **Manual** – Once per System Integration and/or User Acceptance test phases  **Automated** - iteratively as planned. | Test Analyst / End User SME/ Automation Engineer | Test Analyst / End User SME/ Automation Engineer | QA Environment or Automated Testing Environment | Manual or Automated Test Cases | Testing Tool Repository  Or  SUITE templates  Or  Project’s Excel Test Case template | **Manual** – Agreed upon defects resolved before deploying code  **Automated** –defect logged as defect PBI, and defects fixed before next test cycle (when not fixed before next test cycle the test removed from automated script | **Defect** is logged as new defectPBI | **Manual** on the deploy product increment (DPI) Closure report  **Automated** on 4 Up Status Report with sprint metrics  -Number of regression test executions  Number of regression test executions that failed  Number of defects recorded |
| **Post Deploy Validation** | **Verification & Validation** of deployed functionality | RFC designated Deploy to Production  Data Conversions (if planned) have been completed  Operational Readiness criteria has been met. | All operational readiness criteria have been verified and the system is ready for use by the intended audience | Once after all code has been deployed to production and data conversions have happened. | Release Owner | Developers/ Test Analyst /End User SME | **Production** | Operational Readiness Criteria and/or UAT Cases | Test Case Repository | If issues can be resolved immediately – correct issues  Or  Execute back out procedures | Update RFC as appropriate | Deploy:  Success  Or  Backed out |
| **Add others as needed** |  |  |  |  |  |  |  |  |  |  |  |  |

# Approval Information

**Approvers Signatures (must be authorized approvers/members of the project Change Control Board)**

| Role | Name/Title | Signature | Date |
| --- | --- | --- | --- |
| Business Owner (Authorized Approver) |  |  |  |
| DTMB System Owner (if identified) |  |  |  |
| Project Manager |  |  |  |
| Project Test Manager |  |  |  |

# Appendix

## Test Type Definitions

**“Verification** ensures that ‘you built it right"  
"**Validation** ensures that ‘you built the right thing"

| Test Type | Definition |
| --- | --- |
| ADA Compliance | ADA Compliance states that electronic and information technology must be accessible to people with disabilities in accordance with Americans with Disabilities Act (ADA) Standards published by DOJ. |
| Automated Test | Non-attended testing - scripted tests that can be executed repeatedly with known input / output conditions. Automated testing tools can be utilized for functional, regression and performance testing. |
| Functional | Functional testing is the process of testing each function of the software application in accordance with their respective Functional Specifications. In this testing, each and every functionality of the system is tested by providing appropriate input, verifying the output and comparing the actual results with the expected results. This testing involves checking of User Interface, APIs, Database, security, client/ server applications and functionality of the Application under Test. The testing can be done either manually or using automation. |
| Performance | A type of non-functional test whose purpose is to identify and fix system performance issues before the system goes live. This generally includes load testing, stress testing, stability or volume testing, throughput testing, and ongoing performance monitoring. Performance testing also characterizes the performance of the application and infrastructure under test, providing project stakeholders with data that will guide and drive related capacity planning activities. |
| Regression Test | The purpose of Regression testing is to ensure that existing functionality is not broken with the introduction of new code. Both new and presumably unchanged functionality is executed and validated. Regression testing can be performed across all test stages and must be conducted in System Integration Test (SIT) and User Acceptance Test (UAT).  Given the frequency of code changes being introduced using an agile approach, daily regression testing is necessary and is usually automated.  The Regression Suite should be reviewed every cycle to validate if automated scripts need to be removed or added. It is important for the regression test suite to be a solid representation of all the activities that go on in a true production environment. |
| Secure Coding (appscan tool) | The objective of security testing is to uncover vulnerabilities of a system and determine how protected the data and resources are from external and internal threats. This type of non-functional test is performed in accordance with the security specifications and is used to measure the effectiveness of the system’s authorization mechanism, the strength of authentication, the integrity of the data, and the availability of the system in the event of an attack and level of non-repudiation. |
| Smoke Test | Smoke testing is performed during SIT as well as during UAT immediately following a code migration and is designed to detect potential show-stopping defects which may impede tester activities in SIT or UAT. This is an initial testing effort making sure build and environment is stable enough to continue test execution. Prior to the start of test execution, a list of very important test scripts can be identified that will constitute the smoke test scripts. |
| String Testing | String Testing is a development level test that is performed by developers after completing unit testing. In this a business process test conducted at the sub process level to ensure the process tasks and transactions of the sub process interact correctly. It is typically an exhaustive test focusing on the system functions within the context of a business process.  Successful completion of String Test efforts establishes readiness for Integration Testing. |
| **System Integration Test (SIT)**  **(AKA End 2 End)**  **Previously used terms:** Systems and Standards Testing  Integration Testing  Quality Assurance Testing  System Testing | End 2 End and/or System Integration Test (SIT) is the process of testing the complete application in a fully integrated testing environment that mimics the real-world use, meeting business scenarios including the interaction with external systems. This involves utilizing network communications and simulating interactions with other applications or hardware in a manner matching that of the production environment. The purpose of SIT is to verify that the solution works end to end, as designed. |
| User Acceptance Test (UAT) | The last phase of the software testing process. During UAT, the software users test the software to make sure it can handle required tasks in real-world scenarios, according to specifications. |
| Unit test | Unit Test is the process of testing individual units of functionality. Unit tests verify that individual system components support the related functional, non-functional (technical), and interface requirements as represented in the system specifications. This testing is performed by developers which ensures that the smallest testable module of a solution successfully performs a specific task(s). Unit test are “checked-in” with the corresponding code and may be automated to execute with each build. |

## Test Roles

|  |  |
| --- | --- |
| Test Lead/Manager | The Test Manager provides supervision and guidance and is responsible for prioritizing and coordinating the testing activities of a test team. The Test Manager monitors resource allocation for staff and contractors to align with testing priorities and maintain effectiveness of testing operations. |
| Test Analyst | The Tester provides expertise in the planning, constructing and execution of software quality check. Responsible for applying business and functional knowledge to meet the team’s overall test objectives. Has expertise in the testing principles, processes and methods for agile methods  The tester is also responsible for ensuring that the testing standards, guidelines, and testing methodology are applied as specified in the projects test plan and that all testing results are easily accessible and understandable.  The Tester may perform defect coordination functions, ensuring that test defects are tracked to closure and that the defect repository is kept up to date with the current status. Also included in the defect coordination function is the creation and distribution of test status metrics as required by the project. The Tester interact with the Product Owner and other domain Subject Matter Experts to coordinate testing efforts. |
| Automation Engineer | The Automation Engineer is a trained experienced tester that has received additional training in tools used to design, build and execute automated testing scripts. |
| SME | Subject Matter Expert - A user, generally agency personnel with extensive knowledge of the business domain, expectations, and variations. The SME can test the system identifying the normal and unusual business situations that occur in general use of the application. |

**IMPORTANT!  IN ORDER FOR THE REMAINING PAGES OF THIS DOCUMENT TO FUNCTION PROPERLY, PLEASE DO NOT INSERT/REMOVE ANYTHING PAST THIS POINT!  NOTE:  THIS STATEMENT WILL NOT PRINT, UNLESS PROMPTED.  PLEASE DO NOT REMOVE FROM THE DOCUMENT.**

State of Michigan

Project Test Strategy

Instructions

**NOTE:** There is embedded custom XML in the cautionary note above. As long as it remains in the document with a section break continuous the hidden text will not print. If you wish to send an electronic copy the go to “File” “Info” and select “Check for issues”. Remove all items found that you do not want in the electronic copy. Then save the document again.

**Template Revision History**

| Revision Date | Author | Section(s) | Summary |
| --- | --- | --- | --- |
| 07/2017 | SEPG | All | Initial version |
|  |  |  |  |

Select the level (Organization, Program, and Project) that this document will govern and delete those that don’t apply.

Completion of this document is the responsibility of the Project Manager but it can’t be completed in a vacuum. The Project Manager should collaborate with the Project’s Test Lead and Technical Lead to determine the necessary testing, human and computer, resources needed.

List the factors to successfully test (and deliver as defect free as possible) solution to our Agency Partners.

Obtain the required signatures.

Standard definitions across SOM for each tester role.