Test Plan for Your Project

Date this plan was last saved:

Author:

1.    Testing Overview

1.1.   Purpose and Scope of the Plan

Provide a brief statement of what systems and test phases (e.g., integration, system, alpha, beta, all) are addressed in the test plan.  Reference the requirements which will be verified by this testing.

1.2.   To be Tested

What functions (applications and subsystems) will be verified during the execution of this test plan?

1.3.   Not to be Tested

What will be excluded from the testing?

1.4.   Other Limitations or Assumptions

Are there things that can't be tested?  Are there test methods that can't be used?

2.    Product Changes

2.1.   New, Changed, Obsolete Items

List (or provide a link to) the specific deliverables to be tested.  Each deliverable should have an associated risk (if it fails in production), needed environmental components, and summary of the changes. This list would be used to determine test priorities, track test status, and to verify production readiness.

2.2.   Maintenance - Resolved Defects/Enhancements

List (or provide a link to) bugs or requests targeted to be fixed for this release, if any.

3.   Test Environment

3.1.    System Configuration

List the components that make up each test environment (e.g., integration, beta) on all platforms.  Include servers and other hardware, systems or other software, databases, MCP user codes, etc. Keep in mind that the test environment should look as much like the production environment as possible, in order to provide the highest level of confidence in the verification of the changes. If the environments are too different, changes that appear to function well in test may fail in production. For example:

|  |  |
| --- | --- |
| **Test Environment** | **Environment Details** |
| Unit Testing | * SQL Server name: xxx, version: SQL 2005 Server * Web Server name: ttt, version: Windows 2008 * .net 3.5 framework * ASTRA (Test version) * Other??? |
| System/Integration Testing | * SQL Server name: YYY, version: SQL 2005 Server * Web Server name: ZZZ, version: Windows 2008 * .net 3.5 framework * ASTRA (Test version) * Other??? |
| Acceptance Testing | Same as System/Integration |
| Production Support Acceptance  (For verifying code merges before deployment) | Same as Unit |

3.2.    Data Requirements

List the data needed for testing, e.g., a full copy of a production database from a given financial cycle. Include any specific test data requirements, such as certain transaction types.

3.3.    Utilities, tools, and software

List supporting tools such as simulators or comparison utilities needed for testing (e.g., Red Gate SQL Data Compare, ERGO).

4.    Strategy / Approach

4.1.    Dependencies, Risks and Risk Management

List any items or actions upon which the execution of the tests is dependent.  List any risks if tests are not executed or fail.  Refer to the risk assessment made of the deliverables in the Product Changes section of this plan.

4.2.    Test Approach

Outline how the test environment and deliverables under test will be handled.  Include how tests will be prioritized, types of tests (e.g., parallel), expected phases (e.g., unit, integration, system, alpha, beta, pilot).  Explain how requirements will be traced to tests and how the project team will know which requirements have had at least one associated test case executed (include a link to the requirements/testing (test coverage) traceability matrix.).  Include a strategy for identifying both positive tests (does it work?) and negative tests (it doesn't break).

4.3.    Software Configuration Management (Version Control) and Environment Migration

Describe how the test environment will be controlled, and how the changes will be stored and migrated from the development environment into the final test environment.  If tools are used, identify them (e.g., VSS, SURE).

4.4.    Problem Reporting and Test Tracking Procedures

Describe how tests will be tracked (executed, failed, associated defects) and the problem reporting procedures, including how problems will be prioritized.  For example:

**Problem Reporting**

* As problems are found during system and acceptance testing they will be recorded and tracked so that the list of problems found and the problem status (open, failed, resolved) can be provided to the project team.
* If a problem is found during testing in another system (e.g., ASTRA), it will be logged and reported to that system's owners.
* Priorities are assigned to defects:
  1. High: These errors cause a "stop work" situation, where there is no functional work-around and testing progress is halted.  Since this type of problem is a critical error, problem resolution must be immediate.
  2. Medium: These errors restrict operation of an important function by impacting business function, procedures, controls or public image.  Testing may be interrupted for the particular function, but other testing can be continued.  The resolution of this level of error is a high priority.
  3. Low: These errors represent minor problems that can be worked around with a minimal effort, or cosmetic errors, and do not affect the overall operation of the application or the usability of the system.

**Test Tracking**

* Team members will function as testers, running test cases, evaluating and recording test results, and generating problem reports.
* A test execution log (tests that have been run, passed, failed) will be kept to produce a test results summary.
* If needed, supporting documentation (e.g., screen prints and reports) will be retained for highly critical test runs.

4.5.    Acceptance Criteria

Identify the tasks, milestones or deliverables that must reach a given state in order for the testing phase(s) to be declared complete.

For example:

* All high-priority defects have been corrected and any of the associated tests rerun successfully
* All outstanding (unresolved) defects have been documented and include workarounds where required
* Requirements coverage is 100% (100% of test cases addressing requirements have been executed) or discrepancies documented and acceptable
* Code coverage (the percentage of code tested) is at least 95%
* The success rate (test cases passed) is at least 95%; the failure rate is documented and acceptable
* Acceptor has signed off on test results and outstanding issues.

5.    Staff Requirements / Roles and Responsibilities

|  |  |
| --- | --- |
| **Role** | **Testing Responsibility** |
| Project Manager / Test Coordinator | * Prepare project risk assessment as input to test plan. * Prepare Overall Test Plan * Assist with preparation and execution of system and acceptance test cases * Coordinate with client for acceptance test planning and execution * Track testing coverage and status * Track defects and enhancement requests resulting from testing |
| Developers | * Perform unit and initial integration tests * Assist with preparation and execution of system and acceptance test cases * Provide new or modified components for testing * Perform test environment set-up and environment configuration tests * Provide input for prioritizing defects * Resolve problems found and prioritized during system and acceptance testing |
| Business Subject Matter Experts | * Prepare Acceptance test cases or checklist * Create or identify acceptance test data * Participate in system test activities * Prioritize defects (with other project team members, as needed) * Execute acceptance tests * Report problems as found * Test resolutions to problems * Document acceptance test results |

6.   Work Plan

Prepare a testing work plan with major testing milestones. Document specific functions to be tested either here or in separate test plan documents. Perform these activities in the sequence identified as much as possible. For example:

6.1.    Pre-test milestones

Identify the milestones to be completed prior to starting test execution.  For example:

1. Components to be tested identified
2. Risk assessment complete (new or changed components have been tagged as high, medium or low-risk, based on risk of failure in production)
3. High Level Test planning complete
4. Tests identified
5. Tests mapped to components
6. High priority Tests details documented
7. Test environment ready (Windows and Unisys)
8. Test data ready (e.g., DMS II databases populated, synthetic data created or defined)
9. Unit tests complete
10. Initial integration testing complete

6.2.    Test milestones

Identify the milestones to be completed during actual test execution.  For example:

1. Installation/Conversion tests
2. Basic Configuration Acceptance test
3. Regression tests
4. Functional tests (e.g., business logic scenarios, data handling tests)
5. Security tests
6. Report tests
7. User interface and usability tests
8. System interface tests
9. Performance tests
10. Volume and stress tests
11. Error recovery tests
12. Documentation/help verification
13. User acceptance tests (see User Acceptance Testing on the Software Testing site.)
14. Implementation verification tests

6.3.    Transition to Production milestones

Identify the milestones to be completed prior to or during transition to production. Suggested milestones:

1. Test results and issues summary review.
2. Deliverables accepted by users.
3. Checkpoint - decide whether to move into production.
4. Production deployment (may be in implementation plan)
5. Implementation verification complete (Basic Configuration Acceptance Test).

7.   Issues

Track issues here or provide a link to testing issues with their status.

8.   Tests

Describe the sets of tests to be run.  List the specific test cases here, or provide an overview of the tests, plus a link to the document(s) describing the test cases.  
Possible test sets are listed in the Work Plan section.

The level of detail included in the test cases will vary with the project; possible test case elements include:

* **TestID**
* **Short description**
* Requirement(s) verified
* Preconditions
* Test data
* Steps
* Expected Results
* Actual results

At a minimum, each test case should include a unique ID and a short description. .

9. Test Results Summary

Include the results summary here or provide a link to the detailed and summary results.  For example:

|  |  |  |  |
| --- | --- | --- | --- |
| **Task/Activity/Function** | **Pass/Fail** | **Who Tested** | **Results / Comments** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |