

Aloha Software Quality Assurance Process

Software Quality Assurance (QA) at **Aloha Technology** is oriented to ' *prevention* '. It involves the entire software development process. Prevention includes monitoring and improving the process, making sure any agreed-upon standards and procedures are followed, and ensuring problems are found and dealt with.

Software Testing at Aloha Technology is also oriented to ' *detection* '. Testing involves the operation of a system or application under controlled conditions and evaluating the results.

Every project has unique requirements; hence at the start of a project, a custom optimal SQA process methodology is defined prior to the launch of the project. This SQA process includes:

- Different Levels of Testing
- SQA Resources
- SQA Methodology

Once the SQA process is defined, and approved & signed off by you, it is entered into our version control tool. These SQA standards and templates will be enforced by QA during all levels of testing.

The standards and templates are reviewed and updated periodically by the process owners to ensure they are current and relevant to the ongoing project.

Different Levels of Testing

Aloha Technology has expertise in testing at all of the below testing levels. At each test level, we will document the results. It is recommended to get sign-off and check on all documentation and code, to ensure quality testing according to CM procedures.

Each level of testing is either considered black or white box testing.

Black Box Testing: not based on any knowledge of internal design or code. Tests are based on requirements and functionality.

White Box Testing: based on knowledge of the internal logic of an application's code. Tests are based on coverage of code statements, branches, paths, and conditions.

Unit Testing: Unit Testing is the first level of dynamic testing and is first the responsibility of the developers and then of the testers. Unit testing is performed after the expected test results are met or differences are explainable / acceptable.

Parallel/Audit Testing: Testing where the user reconciles the output of the new system to the output of the current system to verify the new system performs the operations correctly.

Functional Testing: Black-box type of testing geared to functional requirements of an application. Testers should perform this type of testing.

Usability Testing: Testing for 'user-friendliness'. Clearly this is subjective and will depend on the targeted end-user or customer. User interviews, surveys, video recording of user sessions, and other techniques can be used. Programmers and testers are usually not appropriate as usability testers.

Integration Testing: Upon completion of unit testing, integration testing, which is black box testing, will begin. The purpose is to ensure distinct components of the application still work in accordance to customer requirements. Test sets will be developed with the express purpose of exercising the interfaces between the components.

System Testing: Upon completion of integration testing, the Test Team will begin system testing. During system testing, which is a black box test, the complete system is configured in a controlled environment to validate its accuracy and completeness in performing the functions as designed. The system test will simulate production in that it will occur in the "production-like" test environment and test all of the functions of the system that will be required in production.

End-to-End Testing: Similar to system testing, the 'macro' end of the test scale involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

Regression Testing: The objective of regression testing is to ensure software remains intact. A baseline set of data and scripts will be maintained and executed to verify that changes introduced during the release have not "undone" any previous code. Expected results from the baseline are compared to results of the software being regression tested. All discrepancies will be highlighted and accounted for, before testing proceeds to the next level.

Sanity Testing: Sanity testing will be performed whenever cursory testing is sufficient to prove the application is functioning according to specifications. This level of testing is a subset of regression testing. It will normally include a set of core tests of basic GUI functionality to demonstrate connectivity to the database, application servers, printers, etc.

Performance Testing: Although performance testing is described as a part of system testing, it can be regarded as a distinct level of testing. Performance testing will verify the load, volume, and response times as defined by requirements.

Load Testing: Testing an application under heavy loads, such as the testing of a web site under a range of loads to determine at what point the system's response time degrades or fails.

Installation Testing: Testing full, partial, or upgrade install/uninstall processes. The installation test for a release will be conducted with the objective of demonstrating production readiness. This test is conducted after the application has been migrated to the client's site. It will encompass the inventory of configuration items (performed by the application's System Administration) and evaluation of data readiness, as well as dynamic tests focused on basic system functionality. When necessary, a sanity test will be performed following the installation testing.

Security/Penetration Testing: Testing how well the system protects against unauthorized internal or external access, willful damage, etc. This type of testing may require sophisticated testing techniques.

Recovery/Error Testing: Testing how well a system recovers from crashes, hardware failures, or other catastrophic problems.

Acceptance Testing: Acceptance testing, which is a black box testing, will give the client the opportunity to verify the system functionality and usability prior to the system being moved to production. The acceptance test will be the responsibility of the client; however, it will be conducted with full support from the project team. The Test Team will work with the client to develop the acceptance criteria.

Alpha Testing: Testing of an application when development is nearing completion. Minor design changes may still be made as a result of such testing. Alpha Testing is typically performed by end-users or others, not by programmers or testers.

Beta Testing: Testing when development and testing are essentially completed and final bugs and problems need to be found before the final release. Beta Testing is typically done by end-users or others, not by programmers or testers.

SQA Resources

The following testing roles and skill sets are standard for most testing projects. Depending on the project, one or more of these roles may be combined for one person.

Test/QA Team Lead: Coordinates the testing activity, communicates testing status to management and manages the test team.

Testers: Develops test script/case and data, script execution, metrics analysis, and evaluates results for system, integration and regression testing.

Test Build Manager/System Administrator/Database Administrator: Delivers current versions of software to the test environment. Performs installations of application software, and applies patches, both application and operating systems. Performs set-up, maintenance, and back up of test environment hardware.

Technical Analyst & Test Configuration Manager: Performs testing assessment and validates system/functional test requirements. Maintains the test environment, scripts, software, and test data.

SQA Methodology

Aloha Technology's three step testing process will be employed and molded to the organization's structure. We believe using this methodology is important in the development and in ongoing maintenance of our customer applications.

For each testing level, as appropriate, the following activities will be performed:

STAGE 1 – CREATE TEST STRATEGY

INPUTS:

A description of the required hardware and software components including test tools (Test Environment, Test Tool Data). A description of roles and responsibilities of the resources required for the test and schedule constraints (Staff, Schedules). Testing methodology (Standards). Functional and technical requirements of the application (Requirements, change requests, technical and functional design documents). Requirements that the system can not provide (System Limitations).

OUTPUTS:

An approved and signed-off test strategy document, test plan, test cases. Testing issues requiring resolution (usually requires the coordination of client project management).

STAGE 2 – CREATE TEST PLAN / DESIGN

INPUTS:

Approved Test Strategy Document. Automated testware and previously developed scripts, if applicable (Test Tools). Test document problems uncovered as a result of testing (Test Document Problems). Understanding of software complexity and module path coverage derived from General and Detailed Design documents (Software Design, Code, Complexity Data).

OUTPUTS:

Problems with the design fed back to the developers (Software Design, Code Issues). Approved test scenarios, conditions and scripts (Test Design, Cases, Scripts). Test data.

STAGE 3 – EXECUTE TESTS

INPUTS:

Approved test documents (Test Plan, Cases, Procedures). Automated testware, if applicable and developed scripts (Test Tools). Changes to the design (Change Requests). Test data. Availability of the test and project teams (Project Staff, Test Team). General and Detailed Design Documents (Requirements, Software Design). A complete development environment that has been migrated to the test environment (Unit Tested Code) via the Configuration/Build Manager. Test Readiness Document. Update documents.

OUTPUTS:

Changes to the code (Test Fixes). Test document problems uncovered as a result of testing (Test Document Problems). Problems with the design fed back to the developers and clients (Requirements, Design, Code Issues). Formal record of test incidents (Problem Tracking - PT). Baseline package ready for migration to the next level (Tested Source and Object Code). Log and summary of the test results (Test Report). Approved and signed-off with revised testing deliverables (Updated Deliverables).