



White paper

Testing, testing, and more testing

Ingredients of a simple QA plan

www.stdtime.com
Scoutwest, Inc.

Testing, testing, and more testing

Writing a QA plan is similar to writing a business plan or development plan. It is usually written once, and updated from time to time as your organization matures. The main purpose is to communicate your wishes to the entire organization, and define the policies that govern those wishes. Putting your thoughts on paper will solidify them for yourself and others. Keeping them in your head will only open the door to miscommunication and a wide range of tactical missteps. Communicating these concepts to employees will have a direct impact on the quality of your product. The list below offers suggestions for items you might consider discussing in your QA plan.

1) Roles and responsibilities

Define the responsibilities each member of the QA team should have. Look outside the formal QA department for the help you'll need from development, documentation, and product management. If your QA is to be successful, you'll need assistance from, and good working relationships with, those people.

2) Liaisons

Define how QA will communicate with development engineers, product managers, and documentation. This includes how issues are passed between people and how issues are resolved. Include formal and informal meetings between QA, development, and documentation people.

3) Multi-layered testing

Include a brief overview of the hierarchy of testing your product will undergo. Redundancy in testing is necessary for quality. Include developer unit testing, testing by QA engineers, and reviews by QA and development managers.

4) Technical reviews

Experienced development engineers usually carry out this level of testing, which includes architectural, engineering, and source code reviews. Finding engineering defects before they go to QA is an incredible time saver.

5) Usability reviews

Product and QA managers and product planners should get a first-peek at user interface designs before formally going to QA. Oftentimes they can spot usability issues and signal a redesign before QA wastes time testing a bad interface that will have to change later.

6) Unit testing

Development engineers should have the responsibility of testing all aspects of their work before product managers and QA ever see it. Engineers can catch about 75% of the early bugs themselves. This prevents QA from wasting time finding and shepherding unnecessary issues.

7) System testing

After a product release or build has passed the first layers of testing by engineers and product planners, QA must test all aspects of the system in a black-box manner. Their testing must be completely independent and objective.

Development engineers must not take part in this testing. QA must try to put themselves in the place of customers, using the product as it was intended.

8) Defect and issue tracking

Define how you will track defects and other issues. Issues should pass through a thorough routing list to ensure that they are dealt with appropriately, and that everyone concerned has had a chance to evaluate them. Don't sweep issues under the rug or they will come back to haunt you.

9) Defect triage

Every product ships with defects. Define how you will determine defect severity, and the levels of defects you are comfortable shipping. Define your regularity of triage meetings, and who should be in attendance.

10) Trend analysis

Watching trends can help you understand the nature of your particular QA cycle. Trends can help predict how many defects you should expect to encounter, your percentage of completion, when your product will ship, and the percentage of feature coverage you have performed. Determine how this analysis will take place and what reports you'll need.

11) Issue workflow

Define the exact workflow each kind of issue should travel through. Define the rules for passing issues from person to person. This will help ensure that nothing falls through the cracks. If this workflow is not defined, employees will assume that certain issues can be deleted without review. This is dangerous.

12) Time tracking

Define how your QA group will track time on each section of the product, and each phase of the project. This will help make future time predictions more accurate. Do you know how much time the formal QA phase should take?

13) Defect seeding

Try intentionally inserting a number of known defects into the product, and watching for these issues to be found by QA. The number of issues found by QA will help you statistically determine how many defects your product has. Use a formula such as this to determine the total number of defects: $\text{TotalDefects} = (\text{SeededDefects} / \text{SeededDefectsFound}) * \text{DefectsFoundSoFar}$. By using this approach, you'll know approximately how many defects have yet to be found!

14) Alpha releases

Set standards for your use of the term “alpha”. Everyone has differing opinions of what an alpha product is. Define the level of quality, functionality, and usability. Determine who may view an alpha product and what your expectations are from them.

15) Beta

Define your use of the term “beta.” You may even consider defining beta 1, 2, and 3 levels of quality and functionality. Don’t jump the gun and call a prototype a beta. Get a list of your beta testers well in advance. Contact them well before the first beta is available to make sure they have the time and inclination to help you test. Set up lines of communication between external beta testers and development engineers. This facilitates fast turn around of defects found by them. Set up your defect tracking system to allow external testers to participate in the process and let them see the progress their issues are making toward resolution and verification.

16) Release Candidate

Define your strategy for final endgame releases. Who can participate, and how tight are your QA requirements near the end of the release? Obviously, you don’t want sweeping architectural changes made near the end of the QA cycle that may destabilize a release.

17) Gold

What are your signoff requirements for the final release? How much “settle time” do you give the product before release? What are your procedures for releasing the product into the wild?

About Us

Scoutwest, Inc. develops and publishes project management and time tracking products for consulting, manufacturing, government, and general business applications.

Thousands of small to large businesses, in dozens of countries worldwide, trust their mission critical business processes to Scoutwest products. Standard Time® and Standard Issue® work together to offer well-rounded project management solutions.

We specialize in packaged software for timesheets, project management, time tracking, defect tracking, and issue tracking. Standard Time is a web-based timesheet that also runs on Windows, Palm OS, and Pocket PC. It can be used for client billing and task management. Standard Issue is used for bug tracking and general issue tracking.

Please visit these web sites for more information.

www.stdtime.com
www.sdtissue.com