# Rapid Software Test applied to Acceptance Testing

Oliver Tuschling, 2008

*Acceptance testing is defined as the testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system.*

*Rapid testing is indispensable when you are asked to test something on short notice, off the top of your head, early in the development process, or when you're in the process of developing test cases and procedures for future use. This approach is also useful even when you're called upon to test more thoroughly, and given the time and resources to do so.*

In some software companies, when a new build of a new or an existing version for a specific product is delivered to a Software Quality Assurance (SQA) department, an acceptance test is executed to evaluate if the code provided has the minimum quality to be completely tested as planned. If the acceptance test passes, the product is installed in SQA and the department starts testing it.

This article tries to integrate as much as possible the Rapid Software Test strategy to the Acceptance test phase of the test cycle with the objective of increasing the quality of the test cases in less time and due to this the quality of the product.

Rapid acceptance test may include manual and automated tests. The objective of this test is to find major issues (crashes, hangs, obvious errors …) just when the software is delivered to SQA department.

Depending on the results of this rapid test, the build will be accepted or rejected in the SQA department. If rejected, SQA department will continue testing current build until a new build is provided that fixes detected errors. If errors are found but they're not preventing the department for testing, the build may be accepted with reservations, and in the report all this reservations should be logged and also reported into the used defect tracking system. It is SQA department responsibility to track and report detected issues.

Testers may have the ability to define rapid and changeable test cases that may find these types of errors. The test should be designed to be executed rapidly. User deep experience on the application functionality tested is important as sometimes tester's intuition is the key on detecting errors.

It is not mandatory to have all the test cases previously defined. As told before, it is more important to have efficient results rapidly. For this, there may be times that user starts testing based on the intuition and the deep knowledge of the application, and in parallel, documenting the actions done. Once the execution of the test has been performed and the defects found and reported, the rapid test case document (that does not necessarily accomplish with the standards defined for a test case design) will be the real useful guide to create a more detailed test cases as a next step and include it into the test suite defined for the product. At this point the execution of the rapid test is done and the errors found and tracked.

There is a high risk of not getting the real benefit of the Rapid Test (quality results rapidly) when doing this. As in some cases, people can be confused and run "monkey tests" with low sense and coverage. It may seem that running Rapid Tests for an Acceptance Test can be a bit chaotic and senseless, but as told before. For this, it is important to combine already existing and verified test cases with the new rapid ones.

Before the Rapid Acceptance Test starts, several things should be kept in mind to make sure that the test is not done in a wrong way:

- New functionality (requirements) introduced in the build. Which areas should I attack?
- New code changes done to fix defects. Sometimes fixing a defect implies creating a new one in another area (side effect).
- Usability of the product tested.
- Testers are properly balanced and assigned to areas
- Some testers are reserved to perform some basic and defined existing tests
- Existing functionality that has relation with code changes may be important to be tested
- Documentation provided in the build can be essential to perform some actions against the application
- Database changes (if it is a Client/Server application)
- Existing Automated tests may reduce the acceptance test execution time and will allow having results sooner.