ISSN: 1934- 9955www.ijise.net

Vol-06 Issue-01 Nov 2011

# The Role of Testing in Ensuring Software Quality

**Narender Mk,kerala**

**Abstract:** How effectively a piece of software satisfies coding standards, customer needs, and business requirements is a measure of its quality. There are two main groups into which it falls: Qualities that determine how well software serves end-user needs and fulfills functional criteria are known as software functional quality. Characteristics that determine the extent to which structural criteria are satisfied in order to facilitate the fulfillment of functional needs are known as software non-functional quality. It often pertains to the inner workings and coding of software. The first stage in ensuring high-quality software is testing. This article explains how testing is useful for gauging software quality, as the goal of testing is to identify and repair bugs as soon as they are discovered. In addition, the article explains how the Selenium IDE tool facilitates quality assurance and how the Bugzilla tool facilitates communication between developers and testers. This article will mostly concentrate on functional testing using the ORANGE HRM application. Testing is an integral aspect of quality assurance, the primary goal of which is to eliminate bugs and other system problems.

**Keyword:** Software testing, bugzilla, selenium integrated development environment, quality assurance, and software quality

## 1. Introduction

### Software Quality

Here is definition of software quality according to IEEE
[1]

1)The degree to which a system, component, or process meets specified requirements
2)The degree to which a system, component, or process meets customer or user needs or expectation.

This definition gives us two alternative ways how we can understand software quality.

First defines software quality based on specification prepared during development of software or even before it. This specification is formed by requirements based on customer needs. We can find in some glossary what exactly requirement means in this context. Here is
definition from ISTQB Glossary of Testing Terms [2]

Requirement is a condition or capability needed by a user to solve a problem or achieve an objective that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.

First approach means that errors included both in requirements and software specifications do not reduce software quality.

Second definition is very customer oriented; it is focused on achieving customer satisfaction. As Daniel Galin wrote in his book Software Quality Assurance, from theory to implementation [3] adopting the second approach demands that the developer invests significant professional efforts in examining and in correcting, if necessary, the customer's requirements specifications. As a result, difficulties are expected to arise during the development process of the projects, especially when attempting to prove how well the program fulfils the user's needs.

### Software Quality Assurance

**Definition from IEEE [1]**

1. A planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to establish technical requirements.
2. A set of activities designed to evaluate the process by which the products are developed or manufactured. Contrast with quality control.

This IEEE definition is limited to the development process of software product and also to the technical aspects of the functional requirements.

Second definition comes from book Software Quality Assurance, from theory to implementation [3]. Author extends IEEE definition and adds SQA more space in
development process. Definition by Galim[3]

**Software Quality Assurance Is**:

A systematic, planned set of actions necessary to provide adequate confidence that the software development process or the maintenance process of a software system product conforms to establish functional technical requirements as well as with the managerial requirements of keeping the schedule and operating within the budgetary confines.

According to this expanded definition SQA should be extended to cover the long years of service subsequent to product delivery. It also should include activities that deal with scheduling and the budge.

Software Quality Assurance encompasses the entire software development life cycle and the goal is to ensure that the development and/or maintenance processes are continuously improved to produce products that meet specifications/requirements.

The process of Software Quality Control (SQC) is also governed by Software Quality Assurance

## 2. Quality Assurance versus Quality Control

| *Quality Assurance* | *Quality Control* |
|---|---|
| 1. Quality Assurance helps us to build processes. | 1. Quality Control helps us to implements the build processes. |
| 2. It is the Duty of the complete team. | 2. It is only the Duty of the Testing team. |
| 3. QA comes under the category of Verification. | 3. QC comes under the category of Validation. |
| 4. Quality Assurance is considered as the process oriented exercise. | 4. Quality Control is considered as the product oriented exercise. |
| 5. It prevents the occurrence of issues, bugs or defects in the application. | 5. It always detects, corrects and reports the bugs or defects in the application. |
| 6. It does not involve executing the program or code. | 6. It always involves executing the program or code. |
| 7. It is done before Quality Control. | 7. It is done only after Quality Assurance activity is completed. |
| 8. It can catch an error and mistakes that Quality Control cannot catch, that is why considered as Low Level Activity. | 8. It can catch an error that Quality Assurance cannot catch, that is why considered as High Level Activity. |
| 9. It is human based checking of documents or files. | 9. It is computer based execution of program or code. |

| | |
|---|---|
| 10. Quality Assurance means Planning done for doing a process. | 10. Quality Control Means Action has taken on the process by execute them. |
| 11. Its main focuses on preventing Defects or Bugs in the system. | 11. Its main focuses on identifying Defects or Bugs in the system. |
| 12. It is not considered as a time consuming activity. | 12. It is always considered as a time consuming activity. |
| 13. Quality Assurance makes sure that you are doing the right things in the right way that is the reason it is always comes under the category of verification activity. | 13. Quality Control makes sure that whatever we have done is as per the requirement means it is as per what we have expected, that is the reason it is comes under the category of validation activity. |
| 14. QA is Pro-active means it identifies weaknesses in the processes. | 14. QC is Reactive means it identifies the defects and also corrects the defects or bugs also. |

## 3. Problem Statement

The problem is that how to attain a quality of a software. It can achieve by software testing but software testing has various bottlenecks like process, planning, technology and process related. The main problem of quality arises due to lack of communication between tester and developer. Sometimes, the developers see the testers as their adversaries. Problem arises when the existing tester left the job and when new tester arrives. If everything is verbally, not on paper how can a new tester understand test environment? Another problem is when to use regression testing and how much regression testing is enough?

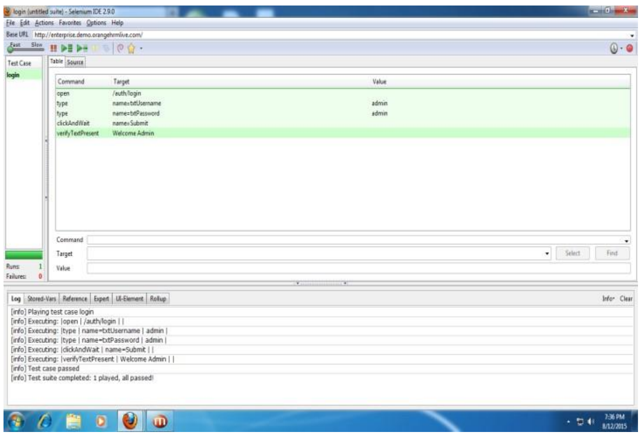
## 4. Proposed Solution

Software testing is first step towards quality. This paper describes how to selenium IDE tool helps in achieving quality within short time span. Communication between tester and developer can improve by the use of bugzilla tool, which is a defect tracking tool. In my thesis i will describe how communication takes place between tester and developer through use of Bugzilla?

This thesis work will also discuss about when to automate the testing process and when to use regression testing and how much regression testing is enough?

### Selenium IDE

Selenium integrated development environment, acronym as Selenium IDE is an automated testing tool that is released as a Firefox plug-in. It is one of the simplest and easiest tools to install, learn and to go ahead with the creation of test scripts. The tool is laid on a record and playback fundamental and also allows editing of the recorded scripts.


login (untitled suite) - Selenium IDE 2.9.0
File Edit Actions Favorites Options Help
Base URL http://enterprise.demo.orangehrmlive.com/
Fast Slow
Test Case
login
Table Source
Command
Target
Value
Select
Find
Runs: 1
Failures: 0
Log Stored-Vars Reference Expert UI-Element Rollup
Info Clear
7:36 PM
8/12/2015

| Command | Target | Value |
|---|---|---|
| open | /auth/login | |
| type | name=txtUsername | admin |
| type | name=txtPassword | admin |
| clickAndWait | name=Submit | |
| verifyTextPresent | Welcome Admin | |

```
[info] Playing test case login
[info] Executing: |open | /auth/login | |
[info] Executing: |type | name=txtUsername | admin |
[info] Executing: |type | name=txtPassword | admin |
[info] Executing: |clickAndWait | name=Submit | |
[info] Executing: |verifyTextPresent | Welcome Admin | |
[info] Test case passed
[info] Test suite completed: 1 played, all passed!
```

**Figure 1:** Snapshots of selenium IDE to record and run the test suit with an application of an ORANGEHRM

ISSN: 1934- 9955www.ijise.net

Vol-06 Issue-01 Nov 2011

## BUGZILLA

Bugzilla is web-based project management software that is being published as **Open source software**. Bugzilla is used to manage software development and help you get a handle on the software development process. Bugzilla is powerful & commanding tool that will allow your team to get organized and communicate effectively. It is allow tracking the bugs & code changes efficiently. This is developed by the Mozilla foundation. This Bug Tracking Tool is used many of top rated organizations like Mozilla, Facebook, NASA, Open Office, RedHat etc.

Bugzilla can Successful projects often are the result of successful organization and communication.

### Benefits of Bugzilla:

Help to increase product quality, get better communication with team members, using bugzilla helps to improve customer satisfaction; Bugzilla can increase the productivity of software development process.

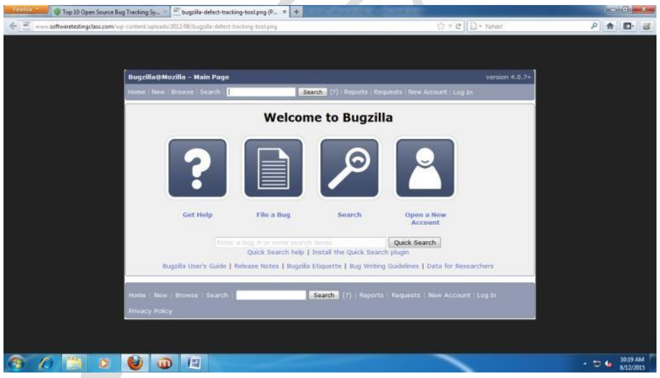

Bugzilla@Mozilla – Main Page
version 4.0.7+
Home | New | Browse | Search | Search | [?] | Reports | Requests | New Account | Log In
Welcome to Bugzilla
Get Help
File a Bug
Search
Open a New Account
Quick Search
Quick Search help | Install the Quick Search plugin
Bugzilla User's Guide | Release Notes | Bugzilla Etiquette | Bug Writing Guidelines | Data for Researchers
Privacy Policy

**Figure 2:** Snapshot of Bugzilla tool

### Software Testing:

Ron Patton in his book Software Testing [4] defines testing as

The goal of a software tester is to find bugs, find them as early as possible, and make sure they get fixed.

A software quality assurance person's main responsibility is to create and enforce standards and methods to improve the development process and to prevent bugs from ever occurring. In real organizations, there the activities of testers**.**

### Definition by IEEE [1]:

1. The process of operating a system or component under specified conditions, observing or recording the results, and making an evaluation of some aspect of the system or component.
2. The process of analyzing a software item to detect the differences between existing and required conditions (that is bugs), and to evaluate the features of the software item.

Software testing is area that is a quite huge. It is not easy to describe it in a few sentences.

**4.1** There are four software testing strategies:

### Unit testing

It is done at the lowest level. It tests the basic unit of software, which can be a module or component. Unit is the smallest module i.e. smallest set of lines of code which can be tested. Unit testing is just one of the levels of testing which contribute to make the big picture of testing a whole system. Unit testing is generally considered as a white box test class.

ISSN: 1934- 9955www.ijise.net

Vol-06 Issue-01 Nov 2011

**Integration Testing**

It is done when two or more tested units are combined into a larger structure. This testing is often done on the interfaces that are between the components and the larger structure that is being constructed, if its quality property cannot be properly assessed from its components.

**System Testing**

It tends to test the end-to-end quality of the entire system. System test is often based on the functional and requirement specifications of the system. Non-functional quality attributes, such as security, reliability, and maintainability, are also checked.

**Acceptance Testing**

It is done when the complete system is handed over to the customers or users from developer side.

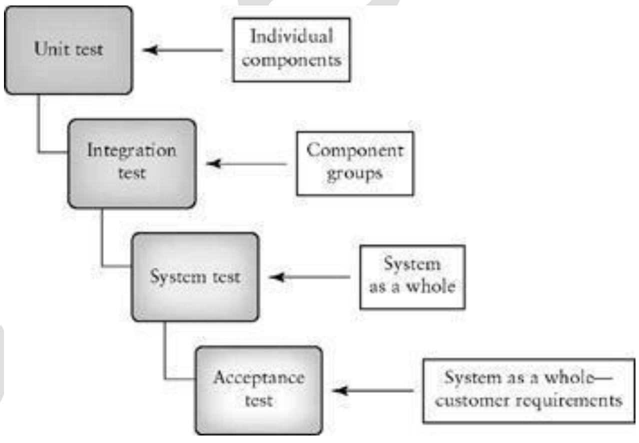
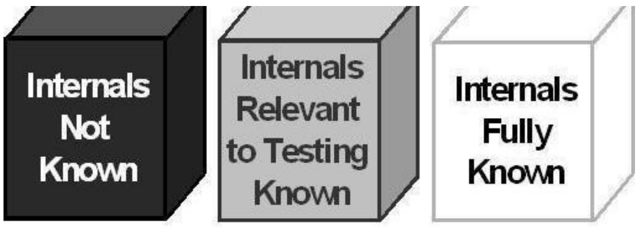


**Figure 3:** Software testing strategies

The aim of acceptance testing is to give assure that the system is working rather than to find errors.

**4.2 Testing Methodology:**



**4.3 Testing Principles:**

A principle is an accepted rule or method for application in action that has to be, or can be desirably followed. Testing Principles offer general guidelines common for all testing which assists us in performing testing effectively and efficiently. Principles for software testing are:

**1)** **Test a Program to Try to make it Fail:** Testing is the process of executing a program with the intent of finding errors [5]. Our objective should be to demonstrate that a program has errors, and then only true value of testing can be accomplished. We should expose failures (as many as possible) to make testing process more effective.

**2) Start Testing Early:** If you want to find errors, start as early as possible. This helps in fixing enormous errors in early stages of development, reduces the rework of finding the errors in the initial stages. Fixing errors at early phases cost less as compared to later phases. For example, if a problem in the requirements is found after releasing the product, then it would cost 10–100 times more to correct than if it had already been found by the requirements review. Figure 1 depicts the increase in cost of fixing bugs detected/fixed in later phases.

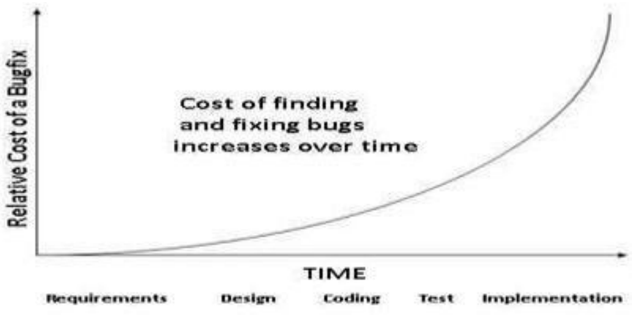


**Figure 4:** Depicts the increase in cost of fixing bugs detected/fixed in later phases

**3) Testing is Context Dependant:** Testing is done differently in different contexts. Testing should be appropriate and different for different points of time. For example, safety-critical software is tested differently from an e-commerce site. Even a system developed using the waterfall approach is tested significantly differently than those systems developed using agile development approach. Even the objectives of testing differ at different point in software development cycle. For example, the objective of unit and integration testing is to ensure that code implemented the design properly. In system testing the objective is to ensure the system does what customer wants it to do [6]. Type of testing approach that will be used depends on a number of factors, including the type of system, regulatory standards, user requirements, level and type of risk, test objective, documentation available, knowledge of the testers, time and budget, development life cycle.

**4) Define Test Plan:** Test Plan usually describes test scope, test objectives, tes strategy, test environment, deliverables of the test, risks and mitigation, schedule, levels of testing to be applied, methods, techniques and tools to be used. Test plan should efficiently meet the needs of an organization and clients as well. The testing is conducted in view of a specific purpose (test objective) which should be stated in measurable terms, for example test effectiveness, coverage criteria. Although the prime objective of testing is to find errors, a good testing strategy also assesses other quality characteristics such as portability, maintainability and usability.

**5) Design Effective Test Cases:** Complete and precise requirements are crucial for effective testing. User Requirements should be well known before test case design. Testing should be performed against those user requirements. The test case scenarios shall be written and scripted before testing begins. If you do not understand the user requirements and architecture of the product you are testing, then you will not be able to design test cases which will reveal more errors in short amount of time. A test case must consist of a description of the input data to the program and a precise description to the correct output of the program for that set of input data. A necessary part of test documentation is the specification of expected results, even if providing such results is impractical [5]. These must be specified in a way that is measurable so that testing results are unambiguous.

**6) Test for Valid as Well As Invalid Conditions:** In addition to valid inputs, we should also test system for invalid and unexpected inputs/conditions. Many errors are discovered when a program under test is used in some new and unexpected way and invalid input conditions seem to have higher error detection yield than do test cases for valid input conditions [5]. Choose test inputs that possibly will uncover maximum faults by triggering failures.

**7) Review Test Cases Regularly:** Repeating same test cases over and over again eventually will no longer find any new errors. Therefore the test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to potentially find more defects. We should target and test susceptible areas. Exploratory Testing can prove very useful. Exploratory testing is any testing to the extent that the tester actively controls the design of the tests as those tests are performed and uses information gained while testing to design new and better tests[7].

**8) Testing must be done by different persons at different levels:** Different purposes are addressed at the different levels of testing. Factors which decide who will perform testing include the size and context of the system, the risks, the development methodology used, the skill and experience of the developers. Testing of individual program components is usually the responsibility of the component developer (except sometimes for critical systems); Tests at this level are derived from the

International journal of imaging science and engineering

ISSN: 1934- 9955www.ijise.net

Vol-06 Issue-01 Nov 2011

developer's experience. Testing at system/sub-system level should be performed by the independent persons/team. Tests at this level are based on a system specification [8]. Development staff shall be available to assist testers. Acceptance Testing is usually performed by end user or customer. Release Testing is performed by Quality Manager.

**9) Test a Program Innovatively:** Testing everything (all combinations of inputs and preconditions) is not feasible except for trivial cases. It is impossible to test a program sufficiently to guarantee the absence of all errors [5]. Instead of exhaustive testing, we use risks and priorities to focus testing efforts more on suspected components as compared to less suspected and infrequently encountered components.

**10) Use both Static and Dynamic testing:** Static testing is good at depth; it reveals developers understanding of the problem domain and data structure. Dynamic testing is good at breadth; it tries many values, including extremes that humans might miss. To eliminate as many errors as possible, both static and dynamic testing should be used [9].

**11) Defect Clustering** Errors tend to come in clusters. The probability of the existence of more errors in a section of a program is proportional to the number of errors already found in that section [7], so additional testing efforts should be more for used on more error-prone sections until it is subjected to more rigorous testing.

**12) Test Evaluation** We should have some criterion to decide whether a test is successful or not. If limited test cases are executed, the test oracle (human or mechanical agent which decides whether program behaved correctly on a given test [10]) can be tester himself/herself who inspects and decides the conditions that makes test run successful. When test cases are quite high in number, automated oracles must be implemented to determine the success or failure of tests without manual intervention. One good criterion for test case evaluation is test effectiveness (number of errors it uncovers in given amount of time)

**13) Error Absence Myth:** System that does not fulfill user requirements will not be usable even if it does not have any errors. Finding and fixing defects does not help if the system built does not fulfill the users' needs and expectations. In addition to positive software testing (which verify that system does what it should do), we should also perform negative software testing (which verify that system does not do what it should not do).

**14) End of Testing:** Software testing is an ongoing process, which is potentially endless but has to be stopped somewhere. Realistically, testing is a trade-off between budget, time and quality [11]. The effort spent on testing should be correlated with the consequences of possible program errors [12]. The possible factors for stopping testing are:

1. The risk in the software is under acceptable limit.
2. Coverage of code/functionality/requirements reaches a specified point.

## 5. Conclusion

Quality is the main focus of any software engineering project. Without measuring, we cannot be sure of the level of quality in software. So the methods of measuring the quality are software testing techniques. In this paper i have discuss about how SELENIUM IDE tool helps to achieve quality and also describe how to communication take place between tester an developer? My thesis work mainly focuses on functional quality.

## Reference

The original source is the IEEE Computer Society. Standard Glossary of Software Engineering Terms, 610.12-1990, IEEE (1990). This resource is accessible online at: http://www.ieee.org/portal/site.
(ISTQB) [online] (2009) is the source for information on software testing qualifications. referenced on 2009-11-30.Obtainable over the Internet at: http://www.istqb.org/index.htm
Michael Galin (2004). Ensuring the quality of software, from theory to practice. Pearson, the publishing house.
the work of Ron Patton in 2001.Testing software.A publication by Sams
According to Myers et al. (1979), "The Art of Software Testing" was published by Wiley in New York.

(ISBN: 0471043281)
Shari Lawrence Pfleeger claimed this in 2001.Theory and Practice of Software Engineeringϟ, Pearson Education
It was 4/16/03 when James Bach died."Explanation of Exploratory Testing," version 1.3
Somerville, Ian (2001, p. 8). Subject: "Software Engineering," Wesley Publishing
(Static and Dynamic Testing Compared) by Programming Research Ltd. [9].
Antoinetta Bertolina (2003) on page 10. "Proceedings of the abstract state machines," "Software Testing Research and Practice," volume 10, issue 1, pages 1–21, of the Advances in Theory and Practice conference.
[11] "Software Testing" by Rajat Kumar Bal.
according to Peter Sestoft (2008-02-25). Version 2 of "Systematic Software Testing"
[1]