Analysis Research And Development Based On Software Testing Techniques And Tools

R. Sridevi, S. Rosi And.G. Sathish

Abstract -Software testing is a process of executing a program or application with the intent of finding the software bugs and we get a zero defeat on that software. It is aimed at evaluating the source of the capability to evaluate the program. It is important to find out the software testing quality of development software in real time. There a lot of advancements have been done in formal methods, validation and verification techniques. We must checkout the software before it should handle to the customer side. There are number of testing tools here in software in based on that software testing. It is important of research and a lot of development has been made in this software testing field. A test strategy is the main of testing approach to software development cycle. Analysis and research based on the development is very essential to the software testes.

Keywords: Software Testing, Techniques, Testing tools, Test strategy, Analysis

INTRODUCTION

Software Testing is an activity that is performed for evaluating software quality and also for improving it. The goal of testing is systematically and stepwise detection of different classes of errors within a minimum amount of time and also with a much less amount of effort. Software testing is also an important component of software quality assurance (SQA), and a number of software organizations are spending up to 40% of their resources on testing. There are four main objectives of testing, IBM Systems Research Institute, Lecturer in Computer Science, Polytechnic Institute of New York, The Art of Software Testing.

Detection: Various errors, defects, and deficiencies are detected. System capabilities and various limitations, quality of all components, the work products, and the overall system will be calculated.

Prevention: In this information to prevent or reduce the number of errors, to clarify system specifications and system performance is provided. Different ways to avoid risks and to tackle problems in the future are identified. It shows how the system can be used with various acceptable risks. It also demonstrates functions with special conditions and products are ready for integration or use. Improving quality by doing effective testing on software, errors can be minimized and thus quality of software is improved.

Risk analysis means the probability by which a software project can experience undesirable events, such as delays, schedule, outright cancellation and cost overruns and much more. So, a number of test cases and test plans are made in testing which means that the behavior of a program is inspected on a finite set of test cases i.e. inputs, execution preconditions, and also expected outcomes for a particular objective, such as to follow a particular program path or to verify compliance with a specific requirement, for which valued inputs are created.

Practically, the set of test cases is considered to be infinite, thus theoretically there are a lot of test cases even for the smallest and simplest program and Software Testing Techniques. In that case, testing could take a lot of time even months and months to execute. Practically, various techniques are used, and some of them are also correlated with risk analysis, while others are correlated with test engineering expertise. The basic purpose of software testing is verification, validation and error detection in order to find various errors and problems and the aim of finding those problems is to get them fixed. Software testing is more than just error detection.

Verification: To verify if system behaves as specified. It is the checking and testing of items, which includes software, for conformance and consistency of software by evaluating the results against pre-defined requirements. In verification we ask a question, are we building the product right?

Validation: In this we check the system correctness which is the process of checking that what has been specified by user and what the user actually wanted. In validation we ask a question: Are we building the right system?

Error Detection: To detect the system errors. A number of tastings should be done to make things go wrong to determine if what things should happen when they should not happen in system software.

SOFTWARE TESTING STRATEGIES

A test strategy is an outline that describes the testing approach of the software development cycle. It

R. Sridevi, Third year MCA, Priyadarshini Engineering College, Vaniyambadi, India

S. Rosi Third year MCA, Priyadarshini Engineering College, Vaniyambadi, India

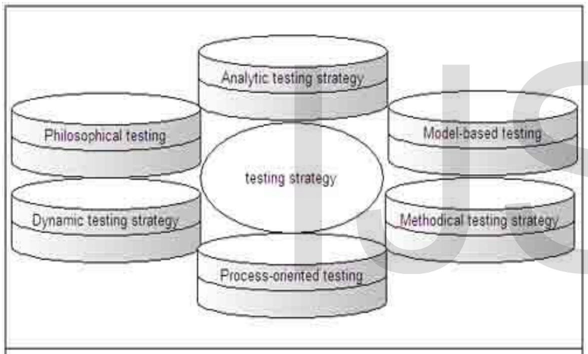
G. Sathish Assistant Professor of MCA, Priyadarshini Engineering College, Vaniyambadi, India

is created to inform project managers, testers, and developers about some key issues of the testing process.
All these strategies provide the tester a template, which is used for testing. Generally, all testing strategies have following characteristics.

- Testing proceeds in an outward manner. It starts from testing the individual units, progresses to integrating these units, and finally, moves to system testing.
- Testing techniques used during different phases of software development are different.
- Testing is conducted by the software developer and by an ITG.
- Testing and debugging should not be used synonymously. However, any testing strategy must accommodate debugging with itself.

TYPES OF SOFTWARE TESTING STRATEGIES

There are different types of software testing strategies, which are selected by the testers depending upon the nature and size of the software.



Types of Software Testing Strategy

1. Analytic testing strategy: This uses formal and informal techniques to access and prioritize risks that arise during software testing. It takes a complete overview of requirements, design, and implementation of objects to determine the motive of testing. In addition, it gathers complete information about the software, targets to be achieved, and the data required for testing the software.
2. Model-based testing strategy: This strategy tests the functionality of the software according to the real world scenario (like software functioning in an organization). It recognizes the domain of data and selects suitable test cases according to the probability of errors in that domain.
3. Methodical testing strategy: It tests the functions and status of software according to the checklist, which is based on user requirements. This strategy is also used to test the functionality, reliability, usability, and performance of the software.
4. Process-oriented testing strategy: It tests the software according to already existing standards such as the IEEE standards. In addition, it checks the functionality of the software by using automated testing tools.
5. Dynamic testing strategy: This tests the software after having a collective decision of the testing team. Along with testing, this strategy provides information about the software such as test cases used for testing the errors present in it.
6. Philosophical testing strategy: It tests the software assuming that any component of the software can stop functioning anytime. It takes help from software developers, users and systems analysts to test the software.

A testing strategy should be developed with the intent to provide the most effective and efficient way of testing the software.

THE SOFTWARE TESTING METHODOLOGIES

The testing methodology should depend on a number of factors, including time allocated to the assessment, access to internal application resources and goals of the test. Security testers should be flexible and be able to plan a test approach for any of these scenarios given the time and access to resources available for an application.

White Box Testing:

Structural testing, also known as glass box testing or white box testing is an approach where the tests are derived from the knowledge of the software's structure or internal implementation. The other names of structural testing include clear box testing, open box testing, logic driven testing or path driven testing. Security is critical when operating a Web application. White-box test design techniques include many testing phases,

- Control flow testing.
- Data flow testing.
- Branch testing.
- Statement coverage.
- Decision coverage.
- Modified condition/decision coverage.
- Prime path testing.
- Path testing.

Black Box Testing:

Black box testing refers to testing a system without having specific knowledge to the internal workings of the system, no access to the source code, and no knowledge of the architecture. Black box tests must be attempted against running instances of applications, so black box testing is typically limited to dynamic analysis such as running automated scanning tools and manual penetration testing. Typical black-box test design techniques include many testing phase,

- Decision table testing.
- All-pairs testing.
- Equivalence partitioning.
- Boundary value analysis.

- Cause–effect graph.
- Error guessing.
- State transition testing.
- Use case testing.

Gray Box Testing:
Gray Box Testing is a testing a system while having at least some knowledge of the internals of a system. This knowledge is usually constrained to detailed design documents and architecture diagrams. It is a combination of both black and white box testing, and combines aspects of each. Gray box testing allows security analysts to run automated and manual penetration tests against a target application. Gray box testing allows those analysts to focus and prioritize their efforts based on superior knowledge of the target system. This increased knowledge can result in more significant vulnerabilities being identified with a significantly lower degree of effort and can be a sensible way for analysts to better approximate certain advantages attackers have versus security professionals when assessing applications.

Black, gray and white box tests are three tests you can conduct to ensure an attacker can't get to your application. Black, white and gray box tests provide different approaches for assessing the security of Web applications. Maximizing the security value of testing approaches when you have limited time and resources requires careful test planning and a thorough understanding of the testing results

SOFTWARE TESTING PRINCIPLES
Test a program so as to make it fail: Testing is the process of executing a program with the intent of finding bugs and errors. Testing becomes more effective when failures are exposed.
Start testing early: This helps in finding and fixing a number of errors in the early stages of development, thus reduces the rework of finding the errors in the later stages.
Testing is context dependent: Testing should be appropriate and different for different context and also at different points of time.
Test Plan: Test Plan usually describes test strategy, test scope, test objectives, test environment, deliverables of the test, risks and mitigation involved, and schedule, levels of testing to be applied, techniques, methods and tools to be used.
Effective Test cases: Effective test cases must be designed so that they can be measured and clear test results are produced.
Test valid as well as invalid Conditions: In addition to valid test cases, test cases for invalid and unexpected inputs/conditions must also be checked. This form of testing is sometimes specified as regression testing.
Test at different levels: Different testing must be done at different level of testing so different people can perform testing differently using different testing techniques at all level.
End of Testing: Testing has to be stopped somewhere. It is stopped when risks are under some limit or if there is some limitation to it.
SELENIUM
Selenium is a portable software-testing framework for web applications. Selenium provides a record and playback tool for authoring tests without the need to learn a test scripting language (Selenium IDE). Selenium is composed of several components with each taking on a specific role in aiding the development of web application test automation.

Selenium IDE

Selenium IDE is a complete integrated development environment (IDE) for Selenium tests. It is implemented as a Firefox Add-On, and allows recording, editing, and debugging tests.

Selenium client API

Application Program Interface is to writing tests in Selfness, tests can also be written in various programming languages. These tests then communicate with Selenium by calling methods in the Selenium Client API. Selenium currently provides client As an alternative APIs for Java, C#, Ruby, JavaScript and Python.

Selenium Remote Control

Selenium Remote Control (RC) is a server, written in Java that accepts commands for the browser via HTTP. RC makes it possible to write automated tests for a web application in any programming language, which allows for better integration of Selenium in existing unit test frameworks. To make writing tests easier, Selenium project currently provides client drivers for PHP, Python, Ruby, .NET, Perl and Java

Selenium Web Driver

Selenium Web Driver is the successor to Selenium RC. Selenium Web Driver accepts commands (sent in Selene's, or via a Client API) and sends them to a browser. This is implemented through a browser-specific browser driver, which sends commands to a browser, and retrieves results. Most browser drivers actually launch and access a browser application (such as Firefox, Chrome or Internet Explorer); there is also an Html Unit browser driver, which simulates a browser using Html Unit.

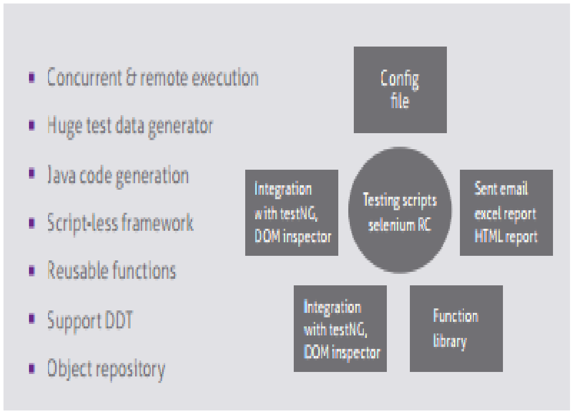
Selenium Grid

Selenium Grid is a server that allows tests to use web browser instances running on remote machines. With Selenium Grid, one server acts as the hub. Tests contact the hub to obtain access to browser instances.

- Free and open-source tool.
- Helps automate testing on AJAX/CSS applications.
- Supports testing on multiple browsers.
- Contains record and playback facility.
- Best for GUI-intelligent field selection (uses IDs, names, or Paths as needed).
- Auto-complete feature available in IDE for all common commands.

Selenium Automation Framework (SAF)

Quicker, more reliable test automation Automation technologies improve test coverage and yield higher quality products. They save thousands of manual test execution hours, significantly reducing costs. SAF is a customized framework developed using Selenium, a widely accepted web application automation tool. It shrinks test cycle times and related costs. Selenium is a portable software testing framework for web applications. The tests can be written as HTML tables or coded in a number of popular programming languages. They can be run directly in most modern web browsers. Selenium can be deployed on Windows, Linux and Macintosh. SAF framework SAF increases automation efficiency by minimizing initial coding effort. It is a script-fewer frameworks used for test automation of web applications that are developed on .Net, Java / J2EE, AJAX. The framework provides a platform to implement data driven and Hybrid – keyword + data driven – framework by spreadsheet template. It can be used in your current automation project. SAF helps enterprises speed up testing using accelerators at the test design layer while keeping the automation suite flexible to interface with commercial tools, whenever needed. The test framework provides a comprehensive reporting dashboard for managing tests.


Concurrent & remote execution
Huge test data generator
Java code generation
Script-less framework
Reusable functions
Support DDT
Object repository
Config file
Integration with testNG, DOM inspector
Testing scripts selenium RC
Sent email excel report HTML report
Integration with testNG, DOM inspector
Function library

QUALITY ASSURANCE (QA)

Quality Assurance is a way of preventing mistakes or defects in manufactured products and avoiding problems when delivering solutions or services to customers and also defines as "part of quality management focused on providing confidence that quality requirements will be fulfilled". This defect prevention in quality assurance differs subtly from defect detection and rejection in quality control, and has been referred to as a *shift left* as it focuses on quality earlier in the process. The terms "quality assurance" and "quality control" are often used interchangeably to refer to ways of ensuring the quality of a service or product.

SOFTWARE QUALITY ASSURANCE (SQA)

Software Quality Assurance is a set of activities for ensuring quality in software engineering processes (that ultimately result in quality in software products). It including some of activates,

- Process definition and implementation
- Auditing
- Training
- Processes could be:
- Software Development Methodology
- Project Management
- Configuration Management
- Requirements Development/Management
- Estimation
- Software Design
- Testing

ITERATIVE AND INCREMENTAL TESTING
Smoke Testing
This testing was done whenever a Build is received (deployed into Test environment) for Testing to make sure the major functionality are working fine, Build can be accepted and Testing can start.

System Integration Testing
This is the Testing performed on the Application under test, to verify the entire application works as per the requirements. Critical Business scenarios were tested to make sure important functionality in the application works as intended without any errors.
Regression Testing
Regression testing was performed each time a new build is deployed for testing which contains defect fixes and new enhancements. Regression Testing is being done on the entire application and not just the new functionality and Defect fixes. This testing ensures that existing functionality works fine after defect fix and new enhancements are added to the existing application. Test cases for new functionality are added to the existing test cases and executed.

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. This paper relates various types of testing technique that we can apply in measuring various quality attributes of principles. Software testing research is the driving element of development and application. In this software testing, it is important to constantly summarize new achievements, propose different ideas in order to promote the study on software testing system engineering of selenium. The quality of assurance was facilitating the development on software testing field and industry.

REFERENCES

1. Fu Bo (2009), Automatic Generation Method of Test Data Based on Ant Colony Algorithm, Computer Engineering and Applications.43(12).
2. Software Testing Techniques Guide to the Software Engineering Body of Knowledge, Swebok – A project of the IEEE Computer Society Professional Practices Committee Stacey (2014).
3. R.S. Pressman & Associates, Inc. (2014). Software Engineering: A Practitioner's Approach, 6/e; Chapter 14: Software Testing Techniques,
4. Myers, Glenford J.(2010), IBM Systems Research Institute, Lecturer in Computer Science, Polytechnic Institute of New York, The Art of Software Testing, by John Wiley & Sons, Inc.
5. Redmill, Felix (2005), Theory and Practice of Risk-based Testing, Vol. 15, No. 1.IEEE(1990), IEEE Standard Glossary of Software Engineering Terminology , Los Alamitos, CA: IEEE Computer Society Press.
6. Zhang Hongchun, Research on New Techniques and Development Trend of Software Testing a QA.
7. Nancy Bordelon, A comparison of automated software testing tools in selenium.