IJCSN - International Journal of Computer Science and Network, Volume 6, Issue 5, October 2017
ISSN (Online) : 2277-5420
www.IJCSN.org
Impact Factor: 1.5

A Novel Testing Techniques and Tools on Software Development

[1] N.Sudheer; [2] Vidushi Sarma; [3] Nesar Ahmad

[1] CSE, Siddharth Institute of Engineering and Technology
Puttur, Andhra Prdesh, India

[2] Computer Engineering, Aligarh Muslim University
Aligarh,India

[3] Computer Engineering, Aligarh Muslim University
Aligarh,India

Abstract - Software Testing is a process of finding errors while executing a program so that we get a zero defect software. It is aimed at evaluating the capability or usability of a program. Software testing is an important means of accessing quality of software. Though a lot of advancements have been done in formal methods and verification techniques, still we need software to be fully tested before it could be handled to the customer side. Thus there are a number of testing techniques and tools made to accomplish the task. Software testing is an important area of research and a lot of development has been made in this field. In this paper, testing techniques and tools have been described. Some typical latest researches have been summarized. Software testing is gaining more and more importance in the future.

Keywords: *Software testing, Software testing strategies, Testing tools, Test plans, Software testing principles, Research orientation.*

1. Introduction

Software Testing is an activity that is performed for evaluating software quality and also for improving it (Guide to the Software Engineering Body of Knowledge, Swebok – A project of the IEEE Computer Society Professional Practices Committee, 2004). Thus, 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 (Myers, Glenford J.(1979), IBM Systems Research Institute, Lecturer in Computer Science, Polytechnic Institute of New York, The Art of Software Testing, by John Wiley & Sons, Inc.):

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

Demonstration: It shows how the system can be used with various acceptable risk. It also demonstrates functions with special conditions and shows how 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. For life-critical software like flight control, testing can be much expensive as risk analysis is also involved. 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 (Stacey, D. A., Software Testing Techniques). In that case, testing could take a lot of time even months and months to execute. So, how to choose a proper set of test cases? 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. Software testing is done under controlled conditions for:

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?

IJCSN
www.IJCSN.org

Copyright (c) 2017 International Journal of Computer Science and Network. All Rights Reserved.

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 errors. A number of testings should be done to make things go wrong to determine if what things should happen when they should not.

2. Software Testing Strategies

A software testing strategy integrates various software test case design methods into a well planned series of steps that result in successful testing of software. Software testing strategies are thus important for testing. Software testing strategy is generally developed by testing specialist, project managers and software engineer. 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.

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

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

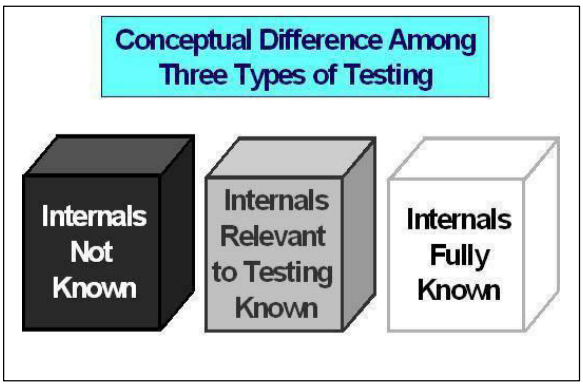
3. Software Testing Methodologies

There are following methodologies for software testing:

White Box Testing

In this testing, internal details and structure of system is made visible. Thus, it is highly efficient in detecting and resolving problems, because bugs can often be found before they cause trouble. We can thus define this method as testing software with the knowledge of its internal structure and coding. White box testing is also called clear box testing, white box analysis or clear box analysis. It is a strategy for finding errors in which the tester has complete knowledge of how the program components interact. This method is rarely used practically for debugging in large systems and networks, thus used for Web services applications. Different types of white box testing techniques are as follows:-

Basis Path Testing
Loop Testing
Control Structure Testing



Software testing methodologies

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.

Black box testing

A black box is any device whose internal details and workings are not understood by or accessible to its user. It is testing of software based on specifications and output requirements and without any knowledge of the coding or internal structure in the program. The main aim is to test how well the system conforms to the specified requirements for the system. Black box testing have little or no knowledge to the internal logical structure of the system. Thus, it only examines the fundamental aspect of the system. It makes sure that all inputs are properly accepted and outputs are correctly produced (Software Engineering: A Practitioner's Approach, 6/e; Chapter 14: Software Testing Techniques, R.S. Pressman & Associates, Inc., 2005). Different types of Black box testing techniques are as follows:-

Equivalent Partitioning Boundary
value Analysis Cause-Effect
Graphing Techniques

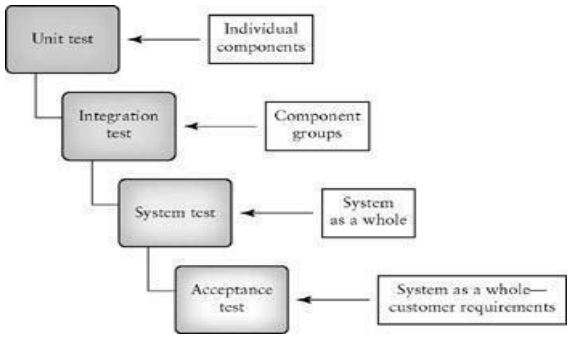
It tends to test the end-to-end quality of the entire system. System test is often based on the functional and requirement specifications of the system. Non-functional quality attributes, such as security, reliability, and maintainability, are also checked.

Acceptance Testing

IJCSN
www.IJCSN.org

Copyright (c) 2017 International Journal of Computer Science and Network. All Rights Reserved.

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



Software testing strategies

Comparison Testing
Fuzz Testing
Model-based testing

Grey Box Testing

In recent years, a third testing method has been also considered i.e. grey box testing. It is defined as testing software and also having some knowledge of its internal logic and underlying code. It uses internal data structures and algorithms for designing the test cases more than black box testing but much less than white box testing. This method holds important when conducting integration testing between two or more modules of code written by different developers, where only their interfaces are exposed for testing (Redmill, Felix (2005), Theory and Practice of Risk-based Testing, Vol. 15, No. 1). This method includes reverse engineering to determine boundary values. Grey box testing is unbiased and non-intrusive because it doesn't require that the tester have access to internal source code.

4. Software Testing Principles

Different software testing principles are as follows:
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, schedule, levels of testing to be applied, techniques, methods and tools to be used. Test plan should accurately meet the needs of an organization and customer as well (IEEE(1990), IEEE Standard Glossary of Software Engineering Terminology, Los Alamitos, CA: IEEE Computer Society Press).

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.

5. Software Testing Tools

There are a number of tools available in market for software testing. Some have been used from a very long time and some new tools have also been developed with a lot of new functionalities. Here, we are going to discuss few tools that are used for automated testing (Nancy Bordelon A comparison of automated software testing tools).

Ranorex

This is a simple, comprehensive and cost effective tool used for automatic testing. It is a better alternative to other testing tools because it tests applications from a user's perspective, using standard language and common programming techniques like C# and VB.net. It does not require understanding a scripting language, because it is coded in pure .net code. Any one of the three languages, VB.net, C# and Iron Python can be used. It is used by a lot of commercial software companies and enterprises around the globe. These simulation tools such can have same problems to the same record and playback methods, as the test plan and test cases are often tightly coupled to the code, and both methods still depend highly on experts to create the correct these tests to ensure full coverage. Future work for ranorex involves creating an easily accessible, open and highly documented interface for the clients to write their own plug-ins, which provides the maximum of recognition for their own applications. Some of the features of this tool are:

- The test automation modules can be created with a standard .NET compiler.
- It provides the ability to do test automation in client's own environment
- It uses standard and modern programming techniques
- It allows testers with little programming knowledge to create professional test plans and cases and modules with Ranorex Recorder
- It does image-based recognition
- It contains Record-Replay functionality which is called Ranorex Recorder
- It provides easy integration for 32 and 64 bit operating systems
- It is built on the .NET Framework

IJCSN
www.IJCSN.org

Copyright (c) 2017 International Journal of Computer Science and Network. All Rights Reserved.

- It offers a standard and flexible test automation interface
- The Ranorex Recorder provides user code actions, which allows developers to provide special validation or automation methods for their testers with less experience in programming
- It targets to get everything flexible and automated
- It supports all the technologies via Ranorex Plug-Ins
- It allows user interface for managing test cases, plans and configurations
- It supports the use of data variables

Rational Functional Tester (RFT)

IBM developed this product in 1999. It is an object-oriented programming based automated testing tool. It includes regression and functional testing tools which note down the results of black box tests in a well scripted format. Once captured, these scripts can be executed against future script builds of any application to verify that new functionalities have not disabled any previous functionality. With the help of this tool, black box tests can be run as well as white box tests for code bottlenecks, memory leaks or measuring code coverage. In 2006, IBM made a major transition to its software development platform to better help companies build complex software and applications. The Baltic or IBM Rational 7 was developed in 2006. Some of the advantages of this tool are:

- It enables regression testing
- It frees up Quality Assurance departments from maintaining and executing basic tests plan and cases, and encourages the creation of additional, thorough tests
- It automates other non testing activities such as functional and test lab machine preparation.
- It reduces the probability of human error that can occur during activities such as test step execution and also test result recording

It works with Web based, Java, and Microsoft Visual Studio, .NET, SAP, terminal-based, Siebel and Web 2.0 applications. This product also uses a Object Code Insertion (OCI) technology where no source code is used. This technology looks at the executable files in an application. These tools when built into the software, including Pure Coverage and Purify Quantify, perform white box testing on a third party code. Some of the advantages of these tools are:

- It provides memory leak detection and run-time error
- It records the exact amount of time an application spends in a given block of code for the purpose of finding all inefficient code bottlenecks
- It pinpoints areas of application that have been and have not been executed
- When performing regression tests on a product, if the application changes, like, images in different locations, tests will not fail because the product uses robust

Janova

This tool is much similar to others as it enables some users to automate software testing solutions and with the help of this tool it is done in a cloud too. This tool does not require any scripts to be written i.e. only simple English-based tools are used that simplify the task of software implementation with efficient and easy to use tools. Other advantage of this tool is that its cost is very less i.e. $10 per month. There is no such software to download and thus no infrastructural investment is required. Since it is used in the cloud, it has a very quick and easy setup that includes no install. This cloud based software has an easy navigation to home page.

6. Latest Research and Development in Software Testing

With the advancement in research and development on component system testing, analysis techniques and form-modeling in embedded software and the software credibility, new results and important issues on software testing keep emerging in last recent years. Some have been introduced below:

The latest results of software testing

Test-driven development

The programming is guided by testing. Before writing the code, we should write the related test cases and plans first, and program that test code, then test that develop code by testing the program and thus the cycle continues, until the development has been completed. The recent popular XP i.e. Extreme Programming mode has strongly advocated this idea of testing (Zhang Hongchun Research on New Techniques and Development Trend of Software Testing).

Iterative and incremental testing

It has evolved from the iterative model. After the iteration is done, the system will automatically integrate some new functions until the entire system function has completed. It mainly focuses on the cumulative functions used in the regression test and each iterative test is completed in two parts: incremental test on current iterative product and the regression test on the wholly completed function of the former iterative cycle. This is one of IBM most widely used test methods.

GUI automation test

This is an automated testing framework which is based on object-oriented capture technology for GUI. About the method of generation of testing case, a type of automatic generation method of test data which is based on ant colony algorithm. By using bit coding, a model for input domain of the software under test to ant paths of the ant colony algorithm was established. It also improved the

IJCSN
www.IJCSN.org

Copyright (c) 2017 International Journal of Computer Science and Network. All Rights Reserved.

variety of ant paths and decreased the degree of the stagnation and precocity. The idea of automatic generation of model-driven software code in MDA improved the automation degree of the software testing.

Testability of component software

On the selection of a test case, a metadata selection method to select a test case is applied. It embeds the information and case to component in order to achieve the generation of a test case, and also used the method of UML to test case meta-model, show any relevant use case meta- model, **,** mapping between them, and the elements of the component metadata. IGA and its advantages on the generation of component software testing case were introduced. Also, an advanced method that is the IIGA, which brought the migration, parallel, self-adoption and immune operator into traditional genetic algorithm also proofed its convergence.

Embedded Software Simulation Test

With the advancement in research and development on component system testing, analysis techniques and form-modeling in embedded software and the software credibility, new results and important issues on software

Various embedded software without any significant modification by simulation of the ARM embedded system on PC were tested. A test development environment for MVC-based embedded software was designed, which not only ensured the successful development of ESTDE but also improved the adaptability and repeatability of the system.

International and domestic research hotspots

The research about the use of software testing techniques and methods directing to the features of software was done. Some were those researches about software testing techniques that aim at different types of software features like embedded systems and the real-time systems, etc (Fu Bo (2007), Automatic Generation Method of Test Data Based on Ant Colony Algorithm, Computer Engineering and Applications.43(12)).

The research about some software testing techniques that direct to other new software development techniques, including the researches about software testing techniques that aim at Internet structure, Object-oriented technology, Java language and software component.

The research about automatic testing technique. It helps to improve the degree of automation in all steps of testing and thus ease the burden of test analysts, such as automatically generating test cases, automatic performance of regression tests, much more.

The research about tools and environment used in testing. Testing environment and testing tool should be developed with the techniques and methods of software testing like testing designing tool, testing planning tool, structure testing tool, testing managing tool, static analysis tool, performance and load testing tool, regression testing tool, and the improvement of interoperability and effectiveness of testing tools.

7. 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. Software testing research is the driving element of development and application. In this era of new and higher demand of software testing, it is important to constantly summarize new achievements, fresh hotspots and propose different ideas in order to promote the study on software testing system engineering, to facilitate the rapid development on software testing field and industry.

References

[1] Fu Bo " Automatic Generation Method of Test Data Based on Ant Colony Algorithm", Computer Engineering and Applications.,Phil. Trans. Roy. Soc. London, vol. A247, pp. 529-551, April 2007.

[2] R.S. Pressman & Associates" Engineering: A Practitioner's Approach" 2005.

[3] Redmill, Felix "Theory and Practice of Risk-based Testing",vol.15No.1,May2005.

[4] Swain,S.K.Mohapatra,D.P.Mall,"Test case generation based on use case and sequence diagram",International journal of Software Engineering(IJSE),vol-7,Issue-5,pg-289-321,2010.

[5] D,Verndon. G.McGraw.2004,"Risk Analysis in software Design. IEEE Security and Privacy.2,4",vol-3,Issue-5, pg 32-37, 2004.

Authors

N.Sudheer completed his B.E from University college of Engineering, Annamalai University and M.Tech from JNTUK Computer science and Engineering. He completed his Ph.D. in the area of Software Engineering from Aligarh Muslim University. During his teaching experience, he worked as Assistant Professor, and Currently working as Associate Professor, in the faculty of CSE at Siddharth Institute of Engineering and Technology [Autonomous], Puttur. His interested areas are Software Engineering. He guided many UG and PG students. He is a profilic researcher and published reputed national and international journals as well as conferences. He is the life member of IAENGE & Involved in Various Editorial Board Member in Reputed Journals.

Dr.V.Sharma born in Aligarh, UP. Worked as a Professor (CSE), in Aligarh Muslim University,Aligarh, Uttar- Pradesh, India. He Guided Number of Ph.D Scholars in Aligarh Muslim University & Published number of Journals. Research interest includes Software Engineering, Networks and Testing Methodologies & Published number of Journals.

Dr. Nesar Ahmad Presently Working as a Professor (CSE), in Aligarh Muslim University,Aligarh, Uttar- Pradesh, India. He Guided Number of Ph.D Scholars in Aligarh Muslim University

IJCSN
www.IJCSN.org

Copyright (c) 2017 International Journal of Computer Science and Network. All Rights Reserved.

& Published number of Journals. Research interest includes Software Engineering, Networks and Testing Methodologies & Published number of Journals.

IJCSN
www.IJCSN.org

Copyright (c) 2017 International Journal of Computer Science and Network. All Rights Reserved.