International Journal of Information Technology and Knowledge Management
January-June 2011, Volume 4, No. 1, pp. 297-299

EXPLORING TESTING STRATEGIES

Ajay Jangra[1], Gurbaj Singh[2], Jasbir Singh[3] & Rajesh Verma[4]

Software testing refers to the inspection of the software with an intention to find out the errors in it. It plays an important role in all SDLC stages. Testing performs to build a quality software, to build confidence in user/developer and to examine the performance of software in all possible circumstances. In this paper we briefly explore all strategies with a single diagram; we draw a testing hierarchy diagram to explain all kinds of testing strategies and their interrelations as a complete in one picture.

Intended Audience: This work provides fundamental knowledge about software testing for researchers, software developers, testers and clients/user's also.

Keywords: Testing, Software Failure, White Box Testing, Black Box Testing, Testing Objectives

1. INTRODUCTION

Testing simply refers to the validation and verification specially to built a good quality software. Testing also defined as "Software testing is technique of evaluating the attributes (i.e.correctness, completeness, security, consistency, unambiguousness, quality etc.) of software and determining that whether it meets its required functionality or not". Purpose of testing is to find out Defects and causes and fixed them as early as possible. Testing requires more project effort and time than any other software development activity; therefore it needs a suitable strategy to make testing successful. Testing is an activity to find the bugs in software that may perform by tester or by applying strategies like white box or black box. So, the activities involved in the testing should be in planned way. [1, 3, 6, 8, 11]

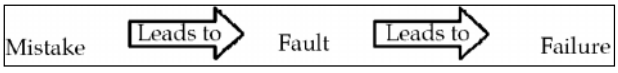

Fig. 1

Some interesting facts about Testing:

- If testing is conducted successfully it will uncover errors in the software. Also it doesn't guarantees that after testing, software is free of defects.
- Testing cannot show the absence of defects, it can show only the defects are present.
- 100% testing is not feasible.

[1, 2, 3] CSE Department U.I.E.T. Kurukshetra University, Kurukshetra, India

[4]Prof. & Head CSE department, K.I.T.M. Kurukshetra, India

Email: [1]er_jangra@yahoo.co.in, [2]gurbaj1986@yahoo.com, [3]jasbir.singh02@gmail.com, [4]vermar.rajesh1974@gmail.com

Testing But Why?

Simple answer is "To find out the faults". Testing performs for following reasons: i) Defect Removal. ii) Correctness. iii) Quality. iv) Consistent v) Completeness vi) Performance. vii) Reduce cost & time...etc. Testing can also perform to add the functionality in software such as: i) Reliability ii) Portability. iii) Reusability. iv) Interoperability. v) Maintainability. vi) Flexibility vii) Testability. viii) Integrity. ix)Usability. x) Efficiency. [1, 3, 6, 8, 12]

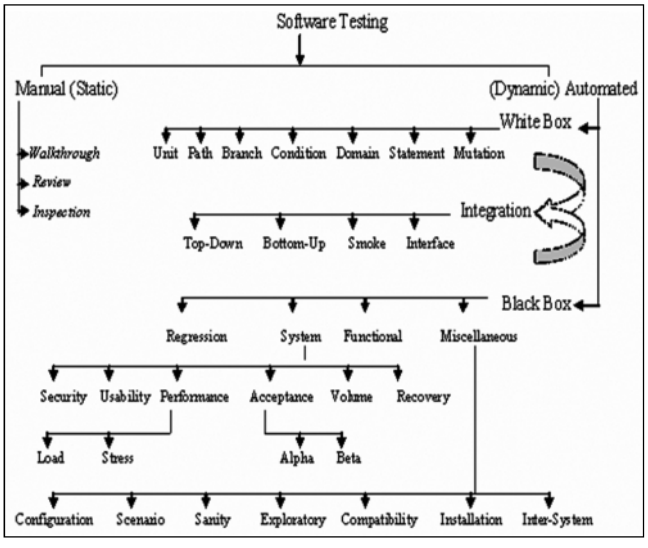

Fig. 2: Testing Strategies in Brief

The above diagram (fig-2) shows the interrelations between various testing Strategies. This diagram mainly divide software testing into two categories (manual/static & automated/dynamic). Both White box and Black box testing drives common strategy known as Integration Testing Strategy. From this testing hierarchy (fig.2) it is easy to understand types of testing and there interrelations, which are explain in the following sections [3, 11].

A. Manual Testing: What, When & How?

It's labor-intensive & slow process where tester tests the software statically. When the software is in early phase of software development life cycle like in analysis phase Manual Testing [8] is to be done, because at that time everything is on paper nothing is for execution therefore Manual Testing is done at this stage. It is done by the analyst, developer and the testing team. Manual testing can be done by (i) Walkthrough: Walkthrough team (project manager & few concerned people) adapt informal method to review the code. (ii) Review: In this, the product documentation is analyzed in details & the conflicts/error is detected, then suggestion and solution are proposed in review report. (iii) Inspection: It is a formal method, minutely checks all the procedures standards and maturity of working environment of the organization and generates an inspection report, on this basis the approval is given.[1, 7, 8]

B. Automated Testing: What, When & How?

Here tester runs the script on any testing tool for testing automatically (to save time), repetitively, and through a number of iterations. Automated testing is done when different modules are ready for execution. Individual/ integrated modules are tested thoroughly using different scripts that are run on testing tool. [2, 4, 7] Automated testing is done by: a) White box testing b) Black box testing

B.1. White Box Testing: What & How?

Also known as glass, structural, open box or clear box testing. Where tester test the internal structure of code. It is done by using following strategies/techniques: (i) Unit testing: It focuses verification effort on the smallest unit of software design. Unit Testing comes at the very basic level as it is carried out as and when a particular functionality is built. It insures that the component being tested conforms to its specifications and is ready to be integrated with other components. (ii) Path Testing: Each and every independent path within the code is executed at least once to find out any error. (iii) Branch Testing: Branch testing helps in validating of all the branches in the code and making sure that no branching leads to abnormal behavior of the application. (iv) Condition Testing: Each and every condition is executed by making it true and false in test cases, in each of the ways at least once. (v) Statement Testing: In this the code is executed at least once in such a way that every statement of the module is executed by the test cases. (vi) Mutation Testing: In this testing, the application is tested for the code that was modified after fixing a particular defect. (vii) Domain Testing: In this testing, the module is tested for all possible inputs. But it is not possible to test all input data because it is infinite so we use Equivalence partitioning and boundary value analysis techniques. [1,2,10]

A-B. Integration Testing: What & How?

Different modules are integrated with each other to form a system and Integration testing use the systematic combination and execution of product components to insure that the interfaces between the components are correct and the product components combine to execute the software's functionality correctly. Integration Testing is done by using following strategies/techniques: (i) Top-Down: In this testing Modules are integrated by moving downward through the control hierarchy, beginning with the main control module. Modules subordinate to the main control module are incorporated into the structure in either a depth-first or breadth-first manner. (ii) Bottom-Up: In this testing, top level modules are tested and the branch of the module are tested step by step using 'Stubs' until the related module comes to an end. (iii) Smoke Testing: Smoke testing is used to check the testability of the application. It is also called 'Build Verification testing or Link testing', it checks whether the application is ready for further major testing and working, without dealing with the finer details. (iv) Interface Testing: It is performed to check, how user-friendly the application is. The user should be able to use the application, without any assistance by the system personnel. [1, 2, 3, 5, 8]

B.2. Black Box Testing : What & How ?

This testing ignores the inspection of internal mechanism of a system and focuses solely on the outputs generated in response to inputs and execution conditions. Tester does not have to deal with the code. It is also called as Behavioral Testing [9]. It is done by using following strategies/ techniques: (i) Regression Testing: In this, tester checks whether a small change in any component of the application does not affect the unchanged components. Testing is done by re-executing the previous versions of the application [13]. (ii) Functional Testing: In this, the software is tested for the functional requirements. It checks whether the module is behaving according to the specification. (iii) System Testing: System testing validates that the software meets its functional and non-functional requirements and is also intended to test beyond the bounds defined in the software/ hardware requirement specifications. System testing is actually a series of different tests whose main purpose is to fully exercise the computer-based system. Following are different types of system testing that are important for software systems [14]. a)Recovery Testing: It check's how fast the software is able to recover from any hardware failure, catastrophic problems or any type of system crash. b) Volume Testing: This testing process huge amount of data is to check the extreme limitations of the system. c) Security Testing: It confirms how well software protects itself against unauthorized internal or external damage of code. d) Usability Testing: This testing checks the ease of use of an application. If user interface is an important issue & needs to be specific for specific type of user then usability testing

is used. It is also called as 'Testing for User Friendliness'. e) Performance Testing : This testing checks whether the system is performing properly, according to the user's requirements. Performance testing is of 2 types: e^1) Load testing: In this testing, the software is raised beyond the limits, to check the performance. e^2) Stress Testing: In this testing, the software is tested beyond the normal expectations or operational capacity. f) Acceptance Testing: This testing verifies that whether the software is acceptable to the customer and it's fulfilling the specified requirements. Acceptance testing is of 2 types: f^1) Alpha Testing: It is performed at the developer's site by the customer in a closed environment. f^2) Beta Testing: This type of Acceptance testing is done at the customer's site by the customer in the open environment. The presence of the developer, while performing these tests, is not compulsory.[1, 4,9]

(iv) Miscellaneous Testing Strategies: a) Configuration Testing: This testing is done to test for compatibility issues. It checks minimum and optimal configuration of hardware and software, and determines the effect of adding or modifying resources. b) Scenario Testing: This testing provides a more realistic and meaningful combination of functions, rather than artificial combinations that are obtained through domain or combinatorial test design. c) Sanity Testing: It uses to check the behavior of the system. It is also called as Narrow Regression Testing. d) Exploratory Testing: It is performed to explore the software features. e) Compatibility Testing: This testing determines if software under supported configurations perform as expected, with various combinations of hardware and software packages. f) Installation Testing: This testing identifies the ways in which installation procedure leads to incorrect results. g) Inter-System Testing: It checks the interface between two or more application systems. [1, 5, 11]

2. CONCLUSION

Although testing is crucial and important to build a good quality software. This paper elaborates all testing strategies, their interrelations, when to perform and how. We design a interrelation diagram of various testing strategies. In this paper we provide answers to the questions that what software testing is?, when and how to perform ? But still there are some queries which are still waiting for the answer like testing: but how much? Good alternatives to testing?

REFERENCE

[1] G.K.Saha "Understanding Software Testing Concepts" Published in ACM Ubiquity, 9, Issue 6, 2008.

[2] Robert Nilsson and Jeff Offutt "Automated Testing of Timeliness : A Case Study" Published in IEEE 2007.

[3] Cem Kaner, J.D,Ph.D. "The Ongoing Revolution in Software Testing" Published in 2004.

[4] B.Baudry, F.Fleurey, J.M Jezequel and Y.L.Traon "Automatic Test Cases Optimization using a Bacterological Adaption Model: Application to .NET Components", Published in IEEE 2002, pp.253-256.

[5] W.K.Chan, T.Y.Chen, T.H.Tse "An Overview of Integration Testing Techniques for Object-Oriented Programs", Published in 2002.

[6] Cem Kaner, Ph.D., J.D, "Inefficiency and Ineffectiveness of Software Testing: A Key Problem in Software Engineering", Published in 2000.

[7] H.Yin, Z.L-Dengel and Y.K.Malaiya "Automatic Test Generation using Checkpoint Encoding and Antirandom Testing", Published in IEEE 1997, pp.84-95.

[8] J.E.Bentlet, W.Bank,C.NC "Software Testing Fundamentals-Concepts, Roles, and Terminology" pp.1-12.

[9] L.Williams "Testing Overview and Black-Box Testing Techniques", Published in 2004, pp. 33-57.

[10] L.Williams "White-Box Testing" Published in 2006, pp.60-74.

[11] D.Graham, E.V.Veenendaal, I.Evans, R.Black, "Foundations of Software Testing", Published in 2008.

[12] I.Sommerville "Software Engineering" Eighth Edition Published in 2007.

[13] William E. Perry, "Effective Methods for Software Testing" Third Edition Published in 2006.

[14] R.S.Pressman "Software Engineering A Practitioner's Approach" Fifth Edition Published in 2001.