International Journal of Research
Available at https://pen2print.org/index.php/ijr/
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 21
October 2018

Framework to Software Testing and Types

Ayesha Nuzha[1], H Meenal[2]
[1]Student, Stanley College of Engineering and Technology for Women
[2]Assistant Professor, Stanley College of Engineering and Technology for Women

Abstract

In this paper we discuss the different aspects, and type of testing as the SRE and Testing run along side by side either by static testing and reviews or running the system/SW to confirm the compliance of requirements/functionalities. Testing types including different common major types of testing Black Box, White Box and there relative advantages, and disadvantages and techniques used in the software/system testing.

Keywords: Software Testing, White Box Testing, Black Box Testing.

I.CONTRACTIONS AND ABBREVIATIONS:

1. ISTQB: International Software Testing Qualification Board
2. CAT: Certified Agile Tester
3. ROI : Return on investment
4. SWEBOK :Software Engineering Body of Knowledge standard
5. SDLC: Software Development Life Cycle
6. SLA: Service Level Agreement
7. SW: Software
8. BVR : Business Value Earned

II. INTRODUCTION OF SOFTWARE TESTING:

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test[1]. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

Testing can be performed either by static review (that is reviewing the documents like requirements, user stories, test Cases/scripts, reviews or dynamically (that is running the code/application and explore the application for the defects and issues)

III. WHAT ARE THE OBJECTIVES OF THE SOFTWARE TESTING?

The major objectives of the software testing is responds correctly to all kinds of inputs (handle properly valid and invalid input), its Performs its functions within an acceptable time based on the SLA (service level agreement) or generally acceptance criteria set, Software is sufficiently usable Software can be installed and run in its intended environment for which it was designed, and Achieved the expected result desired/required it means it serve the functional purpose and fit for the end user use[2].

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 21
October 2018

IV. WHY TESTING IS OBLIGATORY:

Software Testing is obligatory because we all make mistakes. Some of those mistakes are unimportant, but some of them are expensive or even dangerous and catastrophic. We need to check everything and anything we produce because things can always go wrong and as a humans we all make mistakes most of the times. Since we assume that our work may have mistaken, hence we all need to check our own work[2].

However some mistakes come from bad assumptions and blind spots, so we might make the same mistakes when we check our own work as we made when we did it[3]. So we may not notice the flaws in what we have done. Ideally, we should get someone else to check and our work because another person is more likely to spot the flaws.

Sometimes defects may be caused be biased judgment and over confidence which is also a major source of errors and defects in one's works[3]. Ideally, we should get someone else to check our work whom has no emotional attachment to our work and has an unbiased opinion is more likely to spot the flaws. These are several reasons which clearly tells us as why Software Testing is important and what are the major things that we should consider while testing of any product or application[4].

V. WHY TESTING IS VITAL:

Software testing is really required to point out the defects and errors that were made during the different phases of the SDLC. It is essential since testing makes sure of the Customer's reliability and their satisfaction in the application. It is very important to ensure the Quality of the product[5]. Quality product delivered to the customers helps in gaining their confidence by knowing more about received product Testing is necessary in order to provide the facilities to the customers like the delivery of high quality product or software application which requires lower maintenance cost and hence results into more accurate, consistent and reliable results and less down time/maintenance time which in turn increases the BVR (Business Value Earned) by customer. Testing is required for an effective performance of software application or product. It's important to ensure that the application should not result into any failures because it can be very expensive in the future or in the later stages of the development and last but not least it is important to stay in the business and grow the business!

VI. CLASSIFICATIONS OF TESTING:

There are two major methods of testing Black box testing, and white box testing.

a. **Black Box Testing:-** Black Box Testing, also known as Behavioral Testing, is a software testing method in which the internal structure/ design/ implementation of the system being tested is not known to the tester. These tests can be functional or non-functional, most of the time are usually functional. This method is named "Black Box" because the software program, in the eyes of the tester, is like a black box; inside which one cannot see[6]. This method attempts to find errors in the categories of Incorrect or missing functions, Interface errors, errors in data structures or external database access, behaviour or performance errors, Initialization and termination errors.

International Journal of Research
Available at https://pen2print.org/index.php/ijr/
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 21
October 2018

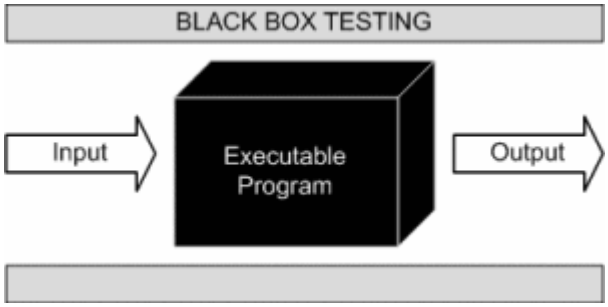

Figure a) Black Box Testing Overview.

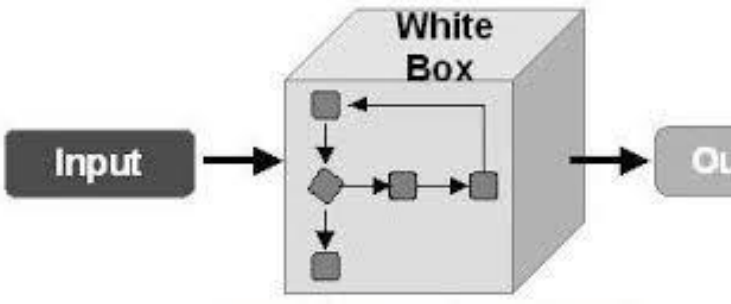

Figure b) White Box Testing Approach

b. **White-box Testing** :-White Box Testing (also synonyms as Clear Box Testing, Open Box Testing, Glass Box Testing, Transparent Box Testing, Code-Based Testing or Structural Testing) is a software testing method in which the internal structure/ design/ implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs. Programming know-how and the implementation knowledge is essential. White box testing is testing beyond the user interface and into the nitty-gritty of a system. This method is named so because the software program, in the eyes of the tester, is like a white/ transparent box; inside which one clearly sees White box test design technique: Procedure to derive and/or select test cases based on an analysis of the internal structure of a component or system, Tests internal structure White box test design technique: Procedure to derive and/or select test cases based on an analysis of the internal structure of a component or system, Tests internal structure Inputs chosen to test specific paths of the code.

VII. LEVEL OF TESTING:

Tests grouped by where they are added in the development process and/or by level of specificity Software Engineering Body of Knowledge (SWEBOK) standard divided and categorized them as below[7].

Unit testing:

Unit tests are basically written and executed by software developers to make sure that code meets its design and requirements and behaves as expected. The goal of unit testing is to segregate each part of the program and test that the individual parts are working correctly.

Integration testing:

Integration testing tests integration or interfaces between components, interactions to different parts of the system such as an operating system, file system and hardware or interfaces between systems.

System testing:

In system testing the behaviour of the whole system/product is tested as defined by the scope of the development project or product.

User Acceptance Testing (UAT):

The User Acceptance test: focuses mainly on the functionality thereby validating the fitness-for-use of the system by the business user. The user acceptance test is performed by the users and application managers to

determine the system is serving the purpose and fit for the user of end user/business.

VIII. TEST TECHNIQUES:

a. Black Box testing techniques
b. White box testing techniques

a. Black Box Testing Technique

Below are the major and famous in industry black box testing techniques including the basic definitions and explanations.

Decision table testing: A decision table is a good way to deal with combinations of things (e.g. inputs). This technique is sometimes also referred to as a cause-effect' table. The reason for this is that there is an associated logic diagramming technique called cause-effect graphing' which was sometimes used to help derive the decision table this as a combinatorial logic network, however, most people find it more useful just to use the table

All-pairs testing: All-pairs testing or pairwise testing is a combinatorial method of software testing that, for each pair of input parameters to a system (typically, a software algorithm), tests all possible discrete combinations of those parameters. Using carefully chosen test vectors, this can be done much faster than an exhaustive search of all combinations of all parameters, by "parallelizing" the tests of parameter pairs Equivalence partitioning.

Equivalence/Class Partitioning: Equivalence partitioning or equivalence class partitioning (ECP) is a software testing technique that divides the input data of a software unit into partitions of equivalent data from which test cases can be derived. In principle, test cases are designed to cover each partition at least once. This technique tries to define test cases that uncover classes of errors, thereby reducing the total number of test cases that must be developed. An advantage of this approach is reduction in the time required for testing a software due to lesser number of test cases. Equivalence partitioning is typically applied to the inputs of a tested component, but may be applied to the outputs in rare cases. The equivalence partitions are usually derived from the requirements specification for input attributes that influence the processing of the test object.

Boundary Value Analysis: Boundary value analysis (BVA) is based on testing at the boundaries between partitions. Here we have both valid boundaries (in the valid partitions) and invalid boundaries (in the invalid partitions).

Cause–effect Graph: Cause–effect graph is a directed graph that maps a set of causes to a set of effects. The causes may be thought of as the input to the program, and the effects may be thought of as the output. Usually the graph shows the nodes representing the causes on the left side and the nodes representing the effects on the right side. There may be intermediate nodes in between that combine inputs using logical operators such as AND and OR.

Error Guessing: Technique that makes use of a tester's skill, intuition and experience in testing similar applications to identify defects that may not be easy to capture by the more formal techniques. It is usually done after more formal techniques are completed.

State Transition Testing: State transition Testing is outputs are triggered by changes to the input conditions or changes to 'state' of the system. In other words, tests are designed to execute valid and invalid state transitions[8].

Use case testing: Use case testing is to identify test scenarios that exercise the whole system on each transaction basis from start to finish. User story testing: It is a part of agile delivery method in which a user story/requirements are transformed to test scenario and further to test case and

then the test case is executed and user story is tested.

Exploratory Testing: Exploratory software testing is a powerful and fun approach to testing. In some situations, it can be orders of magnitude more productive than scripted testing. Exploratory testing is perform in session with define time slot mostly by experienced tester or user for each session around 30 to 60 minutes.

Domain analysis: Domain analysis is the process by which a software engineer learns background information. He or she has to learn sufficient information so as to be able to understand the problem and make good decisions during requirements analysis and other stages of the software engineering process.

Combining technique: Testing two or more variables in combination with each other. We discuss this in the Addendum on Techniques later in this chapter. Combination testing is important, but many testers don't study enough of it. Most benefits provided by the program are based on the interaction of many variables. If we don't vary them jointly in our tests to not to miss errors that are triggered by difficult combinations, rather than difficult individual values.

b. White Box Testing Techniques

White-box test design techniques include the following code coverage criteria.

Control Flow Testing: Control-flow testing techniques are based on judiciously selecting a set of test paths through the program. The set of paths chosen is used to achieve a certain measure of testing thoroughness e.g., pick enough paths to assure that every source statement is executed at least once.

Data Flow Testing: Data flow testing is a family of test strategies based on selecting paths through the program's control flow in order to explore sequences of events related to the status of variables or data objects. Dataflow Testing focuses on the points at which variables receive values and the points at which these values are used.

Branch testing: Branch testing is used to ensure that each one of the possible branch from each decision point is executed at least once and thereby ensuring that all reachable code is executed

Statement coverage: Statement coverage is a white box testing technique, which involves the execution of all the statements at least once in the source code. It is a metric, which is used to calculate and measure the number of statements in the source code which have been executed. Using this technique we can check what the source code is expected to do and what it should not. It can also be used to check the quality of the code and the flow of different paths in the program. The main drawback of this technique is that we cannot test the false condition in it as given by below formula. (Statement coverage = No of statements Executed/Total no of statements in the source code * 100)

Decision/Branch coverage: Decision coverage or Branch coverage is a testing method, which aims to ensure that each one of the possible branch from each decision point is executed at least once and thereby ensuring that all reachable

code is executed. That is, every decision is taken each way, true and false. It helps in validating all the branches in the code making sure that no branch leads to abnormal behaviour of the application.

Modified Condition/Decision Coverage: In software testing, the modified condition/decision coverage (MC/DC) is a code coverage criterion that requires all of the below during testing. Each entry and exit point is invoked, each decision takes every possible outcome, each condition in a decision takes every possible outcome each condition in a decision is shown to independently affect the outcome of the decision[9].

Path /Prime Path testing: The method analyzes the control flow graph of a program to find a set of linearly independent paths of execution as shown by below formula. Branch Testing = (Number of decisions outcomes tested / Total Number of decision Outcomes) x 100 %

IX. MERITS AND DEMERITS OF THE BLACK BOX TESTING

1) Merits of Black Box Testing

Black box is more efficient when applied on large systems, the tester is does not inevitably required to be a more technical person. Since the tester and developer are independent of each other's in black box testing so it is balanced and unprejudiced.

The testers are not necessarily required to have the deep functional knowledge of the system.

The test shall be done from the end user's and business owners' point of view, as at the end user should accept and use the system/SW.

Black box testing additionally contribute to identify vagueness and contradictions in functional design and specifications as the black box testing is proceeded, test scenarios and test case is designed as soon as the functional design and specification are completed and tester do not need to wait to start testing activity test can work in parallel while development team is designing the system and tester are preparing the acceptance criteria/test cases.

2) Demerits of Black Box Testing

Test cases may be challenging to be designed without having clear functional specifications. Also it is difficult to identify tricky and invalid inputs if the test cases are not developed based on specifications.

Black box testing has a charter and time boxing because of this constrain and it is difficult to identify all possible inputs in limited time chartered this may cause a delay in writing test cases which in turn may results some undefined scenarios and negative scenarios are not been designed and tested at all, which may pose a risk to the system/SW.

X. ADVANTAGES AND TRADEOF THE WHITE BOX TESTING

1) Advantages of White Box Testing

White-box testing is one of the two biggest testing methodologies used today. White box testing has several major advantages which are summarized as below. Side effects of having the knowledge of the source code is beneficial to thorough testing. Optimization of code by revealing hidden errors and being able to remove these possible defects. Gives the programmer introspection because developers carefully describe any new implementation provides traceability of tests from the source, allowing future changes to the software to be easily captured in changes to the tests. White box tests are easy to automate, white box testing give clear, engineering-based, rules for when to stop testing.

2) Disadvantages of White Box Testing

Although white-box testing has great advantages, it is not perfect and contains some disadvantages which are summarized as below. White-box testing brings complexity to testing because the tester must have knowledge of the program in which the code is written and the tester most be well versed with the standards and should know the refactoring in that language, including being a programmer, white-box testing requires a programmer with a high level of knowledge due to the complexity of the level of testing that needs to be done.

On some occasions, it is not realistic to be able to test every single existing condition of the application and some conditions will be untested. The tests focus on the software as it exists, and missing functionality may not be discovered as we also aware that 100 % testing is not possible so due to time resection the sometime the critical area of the code/functionality is not thoroughly tested while the allocated time for testing is wasted in trivial part of the code.

XI. CONCLUSION:

Both white and black-box testing are necessary for the successful software/system delivery. In many cases black box testing is done by dedicated testers/testing team while white-box testing is performed by developers or the one whom has good knowledge of the development.

Both types of testing has its advantages, and disadvantages including trade of whoever black box testing is extensively used in industry as compare to white box testing. Most of the organization also prefer Black box testing in order to protect their system/SW to minimize the internal code, design and architecture of their system/SW to as few people as possible.

XII. EPILOGUE, AND REVOLUTIONARY RESEARCH:

It is clear that we can use different approaches for different projects based on the testing scoop and the complexity of the system under test, however it is recommended to use the black box testing techniques for the bigger system which has a lot of integration point as white box testing is implied and suitable for mostly the small project additionally the ROI on white box for bigger project is not suitable as it requires a lots of efforts and drill down in the code, which may be another issues as many organization do not want to open up the code to anyone other than developer.

The anticipated research approach is to use black box with the applicable combination of white box to have a hybrid which sometimes may referred as gray box which is indeed a hybrid of White box and black box. Selecting gray box testing after completing the recommended comparative study and analysis of both the white and black box testing and to get the both possible unique advantages of black box and white box with higher ROI and greater products/systems/SW quality with market competitive cost as the defects detected in last phase or when product is released to the customers are 10X expensive to fix if detected in initial phase of SDLC and mitigated.

XIII. LIST OF FIGURES:

XIV. REFERENCES:

1. Black-box Testing", John Wiley & Sons, 1995, ISBN 0-471-120944
2. The Art of Software Testing - Glenford J. Myers, Corey Sandler, Tom Badgett
3. Software Engineering 8e by Ian Sommerville
4. Exploratory Testing by Kaner, Cem
5. The Mythical man month
6.istqb_foundation_level_syllabus_2011
7.Advanced_syllabus_2012_technical_test_analyst_ga_release_20121019
8.Advanced_syllabus_2012_test_analyst_ga_release_20121019
9. Software Testing, a Craftsman's