Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 6, June 2014, pg.579 – 584

RESEARCH ARTICLE

A Comparative Study of Software Testing Techniques

Anju Bansal

Computer Science & Engg., Maharshi Dayanand University, India

anju0310@yahoo.com

Abstract— Software testing is the process used to measure the quality of developed computer software. It exhibits all mistakes, errors and flaws in the developed software. In this paper, the three most prevalent and commonly used software testing techniques for detecting errors are described and compared, they are: white box testing, black box testing and grey box testing.

Keywords— Black box, Demonstration, Detection, Grey box, Prevention, Software testing, White box

I. INTRODUCTION

Software testing is important activity in Software Development Life Cycle. Software testing is the process of assessing the functionality and correctness of a program through execution or analysis.

The testing of software is an important means of assessing the software to determine its quality. Since testing typically consumes 40~50% of development efforts, and consumes more effort for systems that require higher levels of reliability, it is a significant part of the software engineering. With the development of Fourth generation languages (4GL), which speeds up the implementation process, the proportion of time devoted to testing increased.

Software testing is not a "silver bullet" that can guarantee the production of high quality software systems. While a "correct" correctness proof demonstrates that a software system (which exactly meets its specification) will always operate in a given manner, software testing that is not fully exhaustive can only suggest the presence of flaws and cannot prove their absence. Moreover it is impossible to completely test an application because (1) the domain of program inputs is too large, (2) there are too many possible input paths, and (3) design and specification issues are difficult test. The first and second points present obvious complications and the final point highlights the difficulty of determining if the specification of a problem solution and the design of its implementation are also correct.[1]

As the amount of maintenance and upgrade of existing systems grow, significant amount of testing will also be needed to verify systems after changes are made. Despite advances in formal methods and verification techniques, a system still needs to be tested before it is used. Testing remains the truly effective means to assure the quality of a software system of non-trivial complexity, as well as one of the most intricate and least understood areas in software engineering. Testing, an important research area within computer science is likely to become even more important in the future.

II. OBJECTIVE OF TESTING

The objective of testing is to find problems and fix them to improve quality (Fig.1). Software testing typically represents 40% of a software development budget.

There are four main objectives of software testing:

1) *Demonstration:* It demonstrates functions under special conditions and shows that products are ready for integration or use.

2) *Detection:* It discovers defects, errors and deficiencies. It determines system capabilities and limitations, quality of components, work products and the system.

3) *Prevention:* It provides information to prevent or reduce the number of errors clarify system specifications and performance. Identify ways to avoid risk and problems in the future.

4) *Improving Quality:* By doing effective testing, we can minimize errors and hence improve the quality of software. [4]

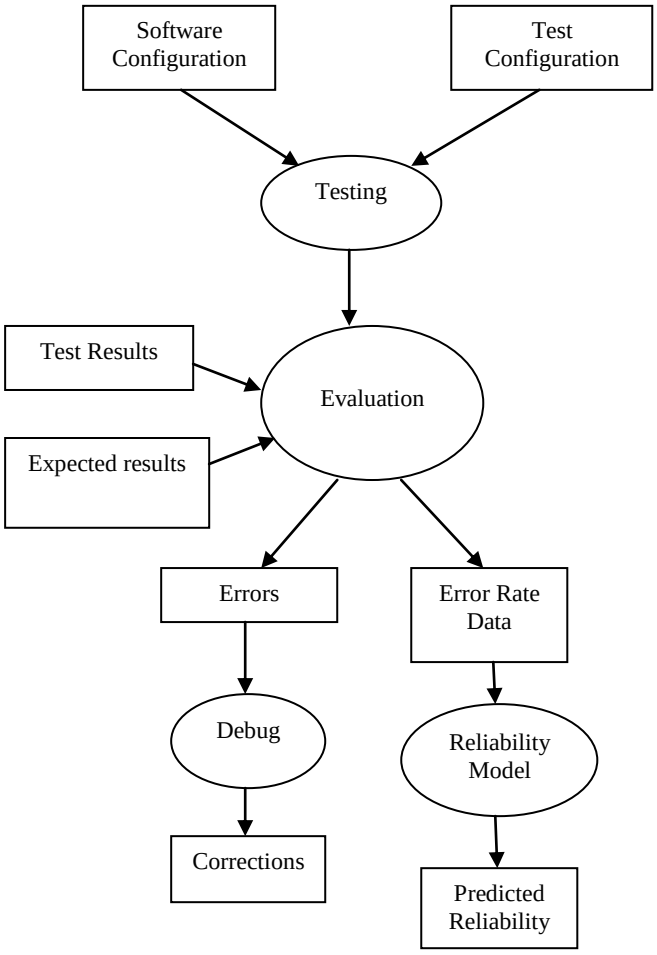

Fig.1 Test Information Flow [3]

III. DIFFERENT TESTING TECHNIQUES

A. Black Box Testing

Black Box Testing is based on the requirements specifications and there is no need to examining the code in black box testing. This is purely done based on customers view point only tester knows the set of inputs and predictable outputs.[9][5]

1) *Equivalence Partitioning:* This technique divides the input domain of a program into equivalence classes from which test cases can be derived, so it can reduce the number of test cases.

2) *Boundary Value Analysis:* It focuses on testing at boundaries, or where the extreme boundary values are chosen. It includes minimum, maximum, just inside/outside boundaries, error values and typical values.

3) *Fuzzing:* This technique feeds random input to application. It is used for finding implementation bugs, using malformed/semi-malformed data injection in an automated or semi-automated session.

4) *Cause-Effect Graph:* In this technique, testing begins by creating a graph and establishing the relation between effect and its causes.

5) *Orthogonal Array Testing:* It can be applied where input domain is very small, but too large to accommodate exhaustive testing.

6) *All Pair Testing:* In this technique, test cases are designed to execute all possible discrete combinations of each pair of input parameters. Its main objective is to have a set of test cases that covers all the pairs.

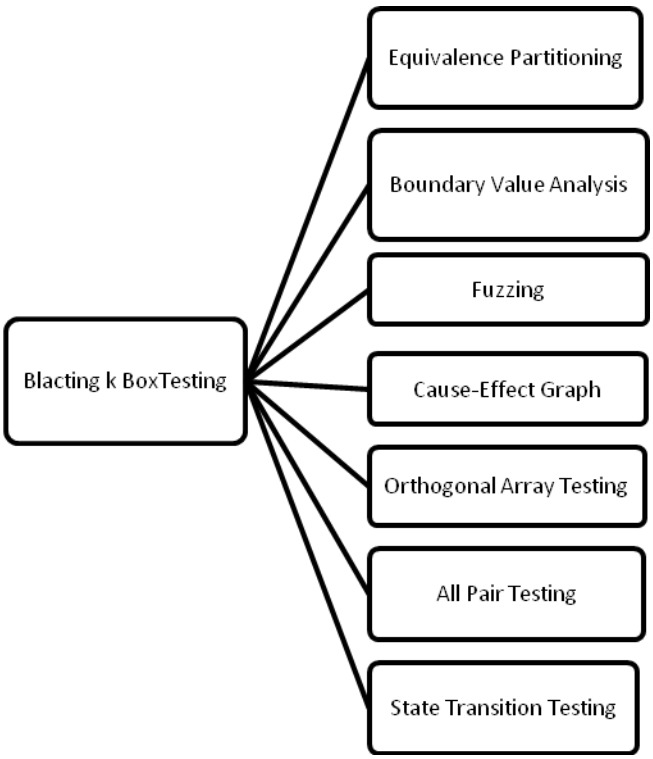

Fig.2 Represent different forms of black box testing [2]

7) *State Transition Testing:* This type of testing is useful for testing state machine and also for navigation of graphical user interface.

Advantages:
1. Testers need not to have knowledge on specific programming language.
2. Testing is done from user's point of view.
3. It helps to expose any ambiguities or inconsistencies in the requirement specifications.[6]
4. Programmer and tester both are independent of each other.

Disadvantages:
1. Test cases are hard to design without clear specifications.
2. Chances of having repetition of tests that are already done by programmer.
3. Some parts of back end are not tested at all.

B. White Box Testing

White box testing mainly focuses on internal logic and structure of the code. White-box is done when the programmer has techniques full knowledge on the program structure. With this technique it is possible to test every branch and decision in the program.[11][4]\

1) *Desk Checking:* Desk checking is the primary testing done on the code. The authors who have knowledge in the programming language very well will be involved in desk checking testing.

2) *Code Walkthrough:* In this testing process a group of technical people go through the code. This is one type of semi-formal review technique.

3) *Formal Inspections:* Inspection is a formal, efficient and economical method of finding errors in design and code. It's a formal review and aimed at detecting all faults, violations and other side effects.

4) *Control Flow Testing:* It is a structural testing strategy that uses the program control flow as a model control flow and favors more but simpler paths over fewer but complicated path.

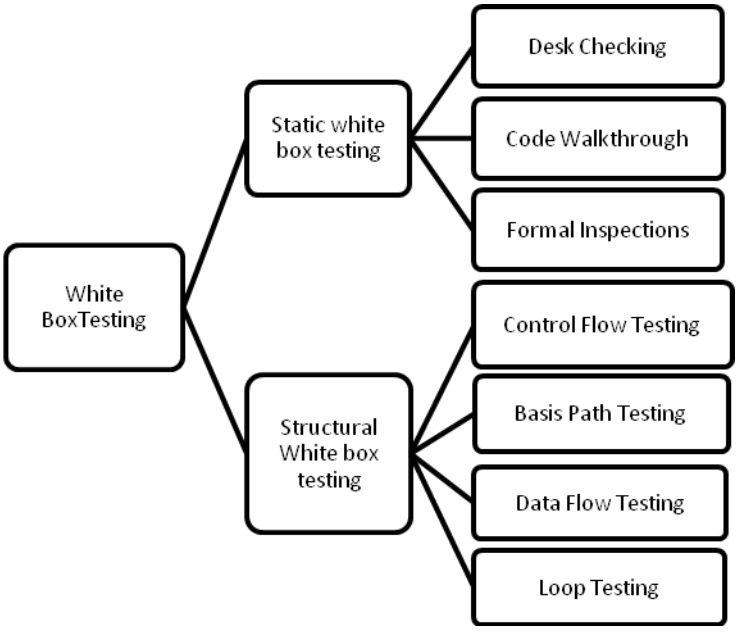

Fig. 3 Represent different forms of white box testing [6]

5) *Basis Path Testing:* Basis path testing allows the test case designer to produce a logical complexity measure of procedural design and then uses this measure as an approach for outlining a basic set of execution paths.

6) *Data Flow testing:* In this type of testing the control flow graph is annotated with the information about how the program variables are define and used.

7) *Loop Testing:* It exclusively focuses on the validity of loop construct.

Advantages:

1. It reveals error in hidden code by removing extra lines of code.
2. Maximum coverage is attained during test scenario writing.[7]
3. Developer carefully gives reasons about implementation.

Disadvantages:

1. A skilled tester is needed to carry out this testing because knowledge of internal structure is required.
2. Many paths will remain untested as it is very difficult to look into every nook and corner to find out hidden errors.

C. Grey Box Testing:

Gray-box testing attempts, and generally succeeds, to combine the benefits of both black-box and white-box testing. Gray-box testing takes the straight-forward approach of black-box testing, but also employs some limited knowledge of the inner workings of the application.
White box + Black box = Grey box,
it is a technique to test the application with limited knowledge of the internal working of an application and also has the knowledge of fundamental aspects of the system. [7] Therefore, a tester can verify both the output of the user interface and also the process that leads to that user interface output. Gray-box testing can be applied to most testing phases; however it is mostly used in integration testing.

1) Orthogonal Array Testing: This type of testing use as subset of all possible combinations.

2) Matrix Testing: In matrix testing the status report of the project is stated.

3) Regression Testing: If new changes are made in software, regression testing implies running of test cases.

4) Pattern Testing: Pattern testing verifies the good application for its architecture and design.

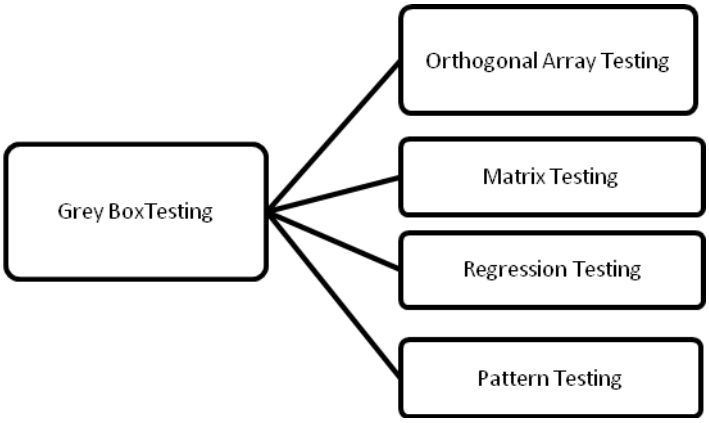

Fig. 4 Represent different form of grey box testing [8]

Advantages:
1. It provides combined benefit of black box and white box testing techniques.
2. In grey box testing, tester can design excellent test scenarios.
3. Unbiased testing
4. Create an intelligent test authoring.

Disadvantages:
1. Test coverage is limited as the access to source code is not available.
2. Many program paths remain untested.
3. The test cases can be redundant.[7]

TABLE 1

COMPARISON BETWEEN THREE FORMS OF TESTING TECHNIQUES

Sr. No.	Black Box Testing	White Box Testing	Grey Box Testing
1.	Analyses fundamental aspects only i.e.no knowledge of internal working.	Full knowledge of internal working.	Partial knowledge of internal working.
2.	It is least exhaustive and time consuming.	Potentially most exhaustive and time consuming	It is somewhere in between the two.

3.	Not suited for algorithm testing.	It is suited for algorithm testing(suited for all).	Not suited for algorithm testing.
4.	Granularity is low.	Granularity is high.	Granularity is medium.
5.	Performed by end users and also by tester and developers(user acceptance testing).	It is performed by developers and testers.	Performed by end users and also by tester and developers (user acceptance testing).

IV. CONCLUSIONS

Software testing is the activity that executes software with an intention of finding errors in it. Software testing can provide an independent view of the software to allow the business to appreciate and understand the risk of software implementation. To carry out software testing in a more effective manner, this paper provides a comparative study of three main techniques of software testing

REFERENCES

[1] Software Testing. Gregory M. Kapfhammer. The Computer Science and Engineering Handbook, CRC Press. May, 2004.
[2] Mohd. Ehmer Khan, "Different Approaches to Black Box Testing Technique for Finding Errors," IJSEA, Vol. 2, No. 4, pp 31-40, October 2011
[3] Jovanovi?, I. "Software Testing Methods and Techniques. " The IPSI BgD Transactions on Internet Research: 30
[4] F. Saglietti, N. Oster, and F. Pinte, "White and grey-box verification and validation approaches for safety- and security-critical software systems," Information Security Technical Report, vol. 13, no. 1, pp. 10–16, 2008.
[5] T. Murnane and K. Reed, "On the effectiveness of mutation analysis as a black box testing technique," in Software Engineering Conference, 2001. Proceedings. 2001 Australian, 2001, pp. 12 –20.
[6] Nidhra, Srinivas, and Jagruthi Dondeti. "Black box and white box testing techniques- A Literature." *International Journal of Embedded Systems & Applications* 2.2 (2012).
[7] Khan, Mohd Ehmer, and Farmeena Khan. "A Comparative Study of White Box, Black Box and Grey Box Testing Techniques." *International Journal of Advanced Computer Sciences and Applications* 3, no. 6 (2012): 12-15
[8] Grey Box Testing from Wikipedia available at http://en.wikipedia.org/wiki/Gray_box_testing
[9] P. Mitra, S. Chatterjee, and N. Ali, "Graphical analysis of MC/DC using automated software testing,"in Electronics Computer Technology (ICECT), 2011 3rd International Conference on, 2011, vol. 3, pp. 145 – 149.

