International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 3 Issue: 12 6674 - 6678

Study and Analysis of Software Testing

Meenu
Department of Computer Science And Applications
RGGCW, Bhiwani
India
meenu.dhull202@gmail.com

Navita
Department of Computer Science And Applications
RGGCW, Bhiwani
India
navita.gill@gmail.com

***Abstract*—**Software testing is a process conducted by the tester to give proper information about the quality of the software. It is a method which provides aids to reduce bugs and maintenance costs of the software .It provides a conceptual view of the software to allow the business to acknowledge and determine the risks of the software implementation. It includes different types of techniques and tools to recognize the bugs or errors in the software. These techniques were evolving from the last decade which generates possible test cases and hence improve the quality of the software. The main aim of the testing process is to produce reliable software by removing defects of the software.

Keywords-*Software testing, STLC, Integration testing, Black Box testing and White Box testing.*

*****

I. INTRODUCTION

The testing of the software is an essential aid of checking the software to verify its quality. Testing includes half percentage of the total development efforts. It can be done at any instant of the software development life cycle of software. Modern software systems must have reliable and accurate results. For the assurance of reliable software, static as well as dynamic techniques are used. Static techniques includes the model or software checking or static analysis while dynamic techniques includes testing which generates different test cases. All the techniques have their own pros and cons. Software testing is a critical element of software quality assurance. It encourages the level of confidence of customers and hence increases the business economics. Testing is the process of executing program with the intention of finding errors.

II. IMPORTANCE OF SOFTWARE TESTING

Software testing is important because everyone make mistakes while developing the software. There occur different mistakes, defects or bugs in the software. Some defects may produce hazardous results and some may not .It depends on the importance or the level of the defects. A tester has to check the software to produce the reliable software which satisfies all the requirements of the customers. No software is 100% free from the defects. There are different reasons which clearly represent the importance of the software testing.

Software testing is very important because of the following reasons:

- Software testing is really necessary to figure out the errors or bugs that occur during the software development phases.
- It is required for the assurance of the fulfillment of the customer's requirements.
- It is essential since it makes sure of the Customer's reliability and their satisfaction in the application.
- It is used to enhance the Quality of the product. Quality product delivered to the customers helps in getting their confidence.
- Testing is essential in order to help the customers by providing different facilities 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.
- Testing is necessary for an acceptable performance of software application or product.
- It's necessary to ensure that the application should not result into any defect because it can be very costly in the future or in the later stages of the development.
- It ensures that the output meet the customer and business requirements

III. PRINCIPLES OF SOFTWARE

Software testing is a challenging task, when software testing follows some principles then it helps in the test designing and evaluation of the software development life cycle tasks.

Principles of testing:

1) Testing shows presence of defects:

Testing is the process with the intent of finding defects. No software is 100% defect free. Every software have some defects. These defects may or may not be hazardous .Testing helps in discovering and removing of those defects.

2) Exhaustive Testing is impossible:

To test every single element or every single combination in the software is impossible. Tester selects only selected or most appropriate test cases to check the software. Because if tester checks every combination then no software can be delivered on time as every combination takes a lot of time for checking.

3) Early testing:

The sooner we start the testing activities the better we can utilize the available time. Testing should be done at every phase of the SDLC. Early testing helps in removing the defects early. If the defects are found earlier in SDLC then it is very easy to fix.

4) Pesticide paradox:

In testing process different types of test cases should be generated so that reliable software is made. If same type of test cases is repeated again and again then new defects can't be discovered. To overcome this pesticide paradox different test cases should be written and checked.

5) Defect clustering:

During the process of testing it should be observed that the most defects are related to small modules means 80%of the defects of software are found in 20% of modules i.e. small number of modules contain more defects.

6) Testing is context depending:

Different testing techniques are applied to the nature and type of application . For instance medical diagnosis application requires more testing as compare to a game software.

7) Absence of error is fallacy:

Complete absence of defects doesn't mean that software will produce accurate result.Sometimes it produce bad results after delivery .So testing after delivery should also be done.

IV. SOFTWARE TESTING LIFE CYCLE(STLC)

Software testing life cycle(STLC) is a process with specific number of steps. These steps are executed in a systematic and planned way . By the help of this cycle the quality of the software or product is ensure . Every step of this cycle have their own importance. Different organisation have different phases in the STLC but the basis remains same.

1) Requirement phase:

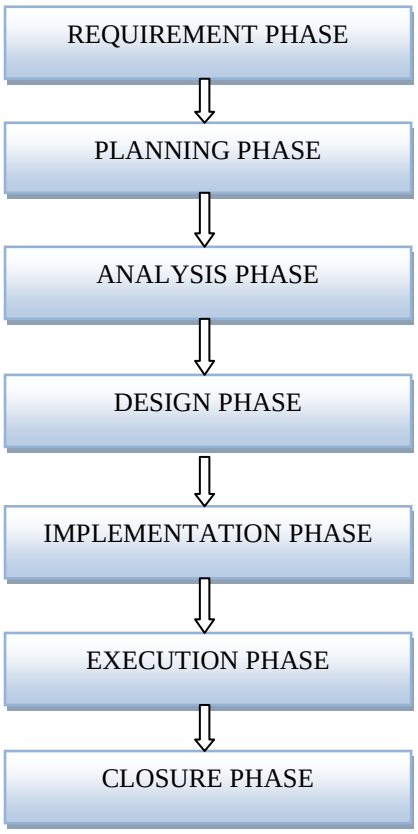
In the requirement phase, quality assurance team analyse and study the requirements of the software. After analysing all the requirements, they fiqure out the list of testable and non testable requirements. If any conflict occurs on any requirement then brainstorming sessions with orther team takes place . This phase collects necessary requirements for futher steps or phases.

2) Planning phase:

It plays vital role in STLC. All the testing approach is defined in this phase. This phase involved the determination of the effort and cost estimation of the entire software or project. It includes the selection of the tools which determine the costs. Test plan document is prepared in this phase. Resource planning and training requirement is also determine in this phase.

3) Analysis phase:

The analysis phase define "what" to be tested not how. Team analyze the test conditions from the requirement document which is prepared in the requirement phase. It defines those requirements which are to be tested in the software. Various factors are determined or analyze for the testing of the requirements.



4) Design Phase:

This phase defines "how" to test. Test data is defined. Test coverage is determined in it. All the conditions are specified for the test cases, which condition is applied on which test case is defined in it. It also identifies the test environment and requirement traceability metrics.

5) Implementation Phase:

Creation of the detailed teat cases taken place in this phase. Regression test suite includes the priority based test cases. Before execution of the test case, the correctness of the test case is checked or verified.

6) Execution Phase:

Test cases are executed in this phase. If the test case is successfully executed then it is said to be passed otherwise it is reported as defect and this defect is send to the developer team for the correctness. This phase actually gives the proper information about the test plans. Test execution report is prepared and circulates to the developer. After fixing the bug the developer team, tester does retesting to ensure that the defect has been fixed.

7) Closure Phase:

Testing team will meet, examine and evaluate artifacts to determine approach that have to be implemented in future, taking lessons from current STLC. It helps in improvement of drawbacks of current STLC in near future. This phase includes different activities like it examine cycle completion criteria based on time, cost, test coverage etc. Closure report is also prepared.

V. LEVEL OF SOFTWARE TESTING

Basically there are 3 levels of testing named as unit, integration and system testing. Tests are combined whereby they are added in the development process.

1) Unit testing:

In the unit testing in the software development process, each unit of program is isolated and it is checked independently that these individual parts are correct with respect to requirements and functionality. Unit testing is often automated but it can also do manually. The size of a unit is so small to test as compare to entire product. When all the units are tested then it become easy to test dependency among modules.The output is ready for integration testing.

2) Integration Testing:

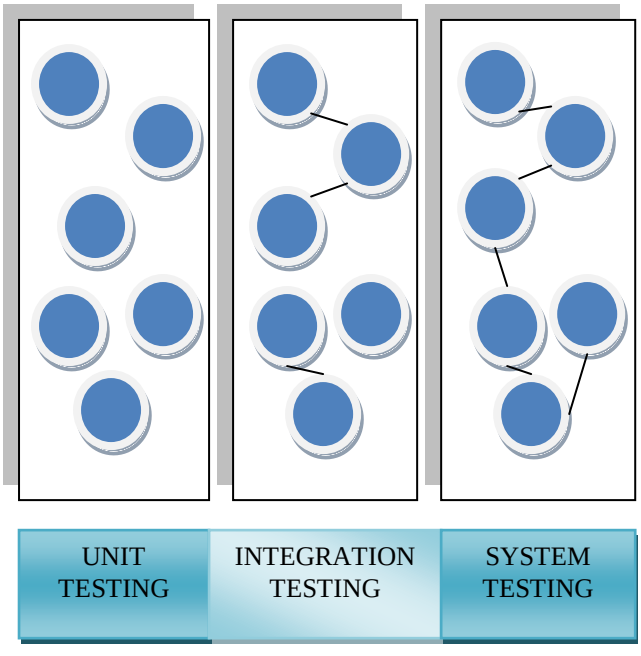
During this phase, units tested during unit testing are grouped in large aggregates called modules and then these modules are tested according to integration test plan defined to those aggregates. Modules are tested according to different classical strategies like top down integration , bottom up integration and sandwich integration. The output is ready for the system testing. Integration testing can unveil the problems with the interfaces among the program components before the problem occurs in real world program execution

.

3) System testing:

In the system testing, the whole integrated system /software is tested. The need of this testing is to find or evaluate the system according to specified conditions. It may include tests based on :

- Requirement specification
- Business process
- Use cases



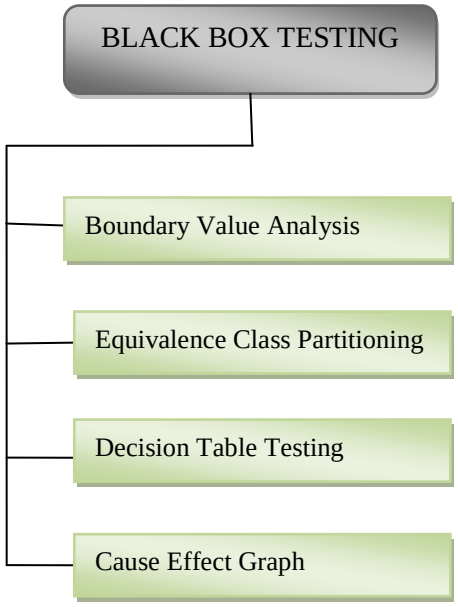
VI. METHODS OF SOFTWARE TESTING

There are different types of software testing . but it is broadly divided into 2 types i.e. black box and white box testing .

Black Box testing:

Black box testing is also called functional or specification based testing. It involves observation of the output of certain input. It doesn't consider any details of the software. Only the output values are checked corresponding to their respective input values. There is no attempt to study or check the code by which the output is produced .It is based on external specifications. It only checks the functionality of the software. It observes the basic aspects of the software, and to check that whether it is according to the user requirements.

There are number of strategies involved in the black box testing that can be used for designing the test cases which are useful for finding the defects.



1) Boundary value analysis:

In this technique input values may be on boundary or just above and below the boundary.Experiences shows that boundary level test cases have higher chances of detecting errors.

2) Equivalence class partitioning

It can minimize the number of test cases by partition the input domain of a program in a finite number of equivalence classes. Input domain includes both valid and invalid states. If any detect occurs in a class then it is expected that test cases belong to that class have similar type of defect. If error is not detected in the class then it is expected that no other test cases would find any defect.

3) Decision table testing:

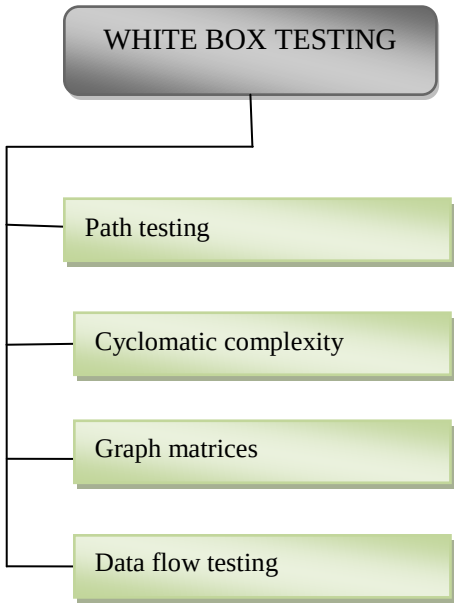
This testing includes a number of combination of conditions to take the decision. It includes 4 parts i.e. condition stub, condion entry, action stub and action entry. Decision tables have been used to define and study the complex logical relationships.

4) Cause effect Graph:

This technique overcome the drawback of boundary value analysis .This testing creates a graph and establish the relation between effect and its causes. It is a technique that helps in selecting, in a sequential manner, a more number of set of test cases. It has beneficial effect in figuring out the incompleteness and ambiguities in the specification.

White Box Testing

White box testing is also called structural or program based testing .It involves the complete observation of the internal structure. Actually it is a complementary approach of black box testing .The internal logic and structure of the software of system is tested in it. For performing white box testing it is necessary to have complete knowledge about the source code. There are number of strategies involved in white box testing which helps in testing the source code of the program.



1) Path testing:

Path testing selects a set of test paths through the program. If the test path is properly chosen, then it means that we have achieved some measure of test thoroughness. It consists flow graph and DD path(decision to decision path) which cover every branch in the program.

2) Cyclomatic complexity

It plays vital role in measuring the complexity metric in Software Engineering. It is defined by Thomas Mc Cabe in 1976. It gives the internal view of code. It is easy to understand and calculate. It can be calculated in one of three ways:

- The number of regions belongs to cyclomatic complexity
- Cyclomatic complexity: E-N+2(where E is no. of edges and N is no. nodes)
- Cyclomatic complexity: P+1(where P is the number of predicate nodes)

3) Graph matrices:

In the testing, flow graph is converted into a square matrix with one row and one column for every node in graph. The main aim of the graph matrices is to traverse all links of graph at least one.

4) Data flow testing:

This testing represents the movement of data in the program . The main concentration is on the usage of variables. The main points include

- Statements where these variables received values
- Statements where these values are used or referenced.

VII. FUNCTIONAL AND NON –FUNCTIONAL TESTING

Functional Testing:

It is form of testing done antagonistic the business demands of applications. It is a black box kind of testing. It takes into account the entire integration system to assess the system's accordance with its described essentials. Based on the utilitarian requirement document this sort of testing is to be accomplished. In real testing, testers demand to check a particular action or utility of the code. For functional testing either manual testing or automation tools can be handled but functionality testing would be easily done using manual testing only. Prior to non Functional testing the Functional testing would be carried first.

Five steps must be hold in mind in the Functional testing:

- Formation of test data set up on the specifications of functions.
- Business demands are the inputs to functional testing.
- Based on functional specifications detect output of the functions.
- The execution of test cases.
- Examine the real and apprehended outputs.

Non Functional Testing:

The non Functional Testing is the form of testing done antagonistic the non functional requirements. Most of the standards are not look upon in functional testing so it is applied to analyze the readiness of a system. Non-functional requirements tend to be those that reflect the quality of the product, especially in the reference of the suitability perspective of its users. It can be started after the completion of Functional Testing. The non functional tests can be productive by utilizing testing tools. The testing of software characteristics which are not associated to any definitive utility or user action like performance, scalability, protection or behavior of application under certain constraints. On functional testing has a extreme impact on customer and user satisfaction with the result. Non functional testing should be indicated in a deductive way, not like "the system should be speedy or "the system should be not difficult to operate" which is not testable. Basically in the non functional test is applied to major non-functional characteristics of software systems. Let's assume non functional requirements examples; in how much period does the software will utilize to accomplish a task? Or how quick the response is.

VIII. CONCLUSION

Software testing is generally less directed and rigorous than it should. A major reason behind it is that we have endeavor to assign finest practices, methodologies, principles, standards for excellent software testing. Testing is an action implement for computing the quality of software and for developing it. Hence, the purpose of testing is sequential disclosure of different classes of faults or defects. It is a process of admitting the actions and accuracy of a software by deep study. The main aim of testing can be as quality assurance, estimation of reliability, validation and verification. Software testing is an basic element of software quality assurance and defines a review of specification, design and coding. The major purpose of software testing is to verify the quality of software system by systematically testing the software in carefully controlled conditions and situations; another aim is to recognize the completeness and correctness of the software.

REFERENCES

[1] Myers, Glenford J. (1979). The Art of Software Testing. John Wiley and Sons. pp. 145–146. ISBN 0-471-04328-1.

[2] Microsoft Development Network Discussion on exactly this topicArchived April 2, 2015 at the Wayback Machine.

[3] Section 1.1.2, Certified Tester Foundation Level Syllabus, International Software Testing Qualifications Board.

[4] "SOA Testing Tools for Black, White and Gray Box SOA Testing Techniques". Crosschecknet.com. Retrieved 2012-12-10.

[5] The Relationship between Risk and Continuous Testing: An Interview with Wayne Ariola, by Cameron Philipp-Edmonds, Stickyminds December 2015.

[6] Ayman Madi, O. K. Z. a. S. K., 2013. On the Improvement of Cyclomatic Complexity Metric. International Journal of Software Engineering and Its Applications, 7(2)

[7] Paul Krill (22 August 2014). "Software testers balk at ISO 29119 standards proposal". InfoWorld.

[8] Gold, R., 2010. Control flowgraphs and code coverage. Int. J. Appl. Math. Comput. Sci., 20(4), pp. 739-749.

[9] Harsh Bhasin, E. K., 2014. Black Box Testing based on Requirement Analysis and Design Specifications. International Journal of Computer Applications, 87(18), pp. 0975-8887

[10] IEEE (1990). IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York: IEEE. ISBN 1-55937-079-3.

[11] Trivedi, S. H., 2012. Software Testing Techniques. International Journal of Advanced Research in Computer Science and Software Engineering, 2(10)

[12] Pan, Jiantao (Spring 1999). "Software Testing (18-849b Dependable Embedded Systems)". Topics in Dependable Embedded Systems. Electrical and Computer Engineering Department, Carnegie Mellon University.

[13] Savenkov, Roman (2008). How to Become a Software Tester. Roman Savenkov Consulting. p. 159. ISBN 978-0-615-23372-7.

[14] EtestingHub-Online Free Software Testing Tutorial. "e)Testing Phase in Software Testing:". Etestinghub.com. Retrieved 2012-01-13.

.