Software Testing – Principles, Methodologies & Limitations

Amit Kumar
Assistant Professor
Department of Computer Science & Engineering
Maulana Azad College of Engineering, Patna, India

Abstract— Software testing is a process of executing a program or application with the intent of finding the software bugs. It can also be stated as the process of validating and verifying that a software program or application or product .Various software development and testing methodologies, tools, and techniques have developed. While it can be argued that there has been some improvement it is apparent that many of the techniques and tools are isolated to a specific lifecycle phase or functional area. The major challenge is to develop a set of test cases for the software system. The set of test cases should be minimum but assure maximum effectiveness. There are lots of testing techniques available for generating test cases.

Key words: Software Testing, Unit Testing, System Testing, Acceptance Testing

I. INTRODUCTION

Testing is a process used to help identify the correctness, completeness and quality of developed computer software. With that in mind, testing can never completely establish the correctness of computer software. 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. Software testing is not only error detection but it has following roles

1) To verify that it behaves "as specified";
2) To detect errors,
3) To validate that what has been specified is what the user actually wanted.

A. Objectives of Testing

- Executing a program with the intent of finding an error.
- To check if the system meets the requirements and be executed successfully.
- To gain confidence of users by giving quality product.

B. Features of Good Test Case

- Accurate: Exacts the purpose.
- Economical: No unnecessary steps or words.
- Traceable: Capable of being traced to requirements.
- Repeatable: Can be used to perform the test over and over.
- Reusable: Can be reused if necessary.

C. Objective of a Software Tester

- Find bugs as early as possible and make sure they get fixed.
- To understand the application well.
- Study the functionality in detail to find where the bugs are likely to occur.
- Study the code to ensure that each and every line of code is tested.
- Create test cases in such a way that testing is done to uncover the hidden bugs and also ensure that the software is usable and reliable.

II. SOFTWARE TESTING LIFECYCLE PHASES

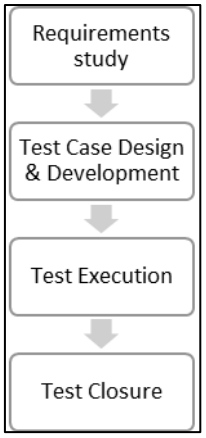

Fig. 1:

A. Requirements Study

- Testing Cycle starts with the study of client's requirements.
- Understanding of the requirements is very essential for testing the product.

B. Test Case Design & Development

- Component Identification
- Test Specification Design
- Test Specification Review

C. Test Execution

- Code Review
- Test execution and evaluation
- Performance and simulation

D. Test Closure

- Test summary report
- Project De-brief
- Project Documentation

III. LEVEL OF TESTING

A. Unit Testing

Unit Testing of software applications is done during the development (coding) of an application .The objective of Unit Testing is to isolate a section of code and verify its correctness. In procedural programming a unit may be an individual function or procedure .The goal of Unit Testing is to isolate each part of the program and show that the individual parts are correct. Unit testing is usually performed by the developer.

Objectives of Unit testing: Following are the main objectives of Unit testing:

1) To test the function of a program or unit of code such as a program or module
2) To test internal logic
3) To verify internal logic.
4) To test path and condition coverage.
5) To test exception conditions & error handling

B. Incremental Integration Testing

Continuous testing of an application as and when a new functionality is added is done. This testing is done after unit testing. Applications functionality aspects are required to be independent enough to work separately before completion of development. This testing is done by programmers or testers as shown in figure 1. Internal and external Application Design, Master Test Plan, Integration Test Plan act as input in this testing. As an output, Integration test output report is generated. It is done by white and black box testing techniques and Configuration Management techniques. Debug, Restructure, Code Analyzers can also be used to complete the testing. Integration Testing It involves building a system from its components and testing it for problems that arise from component interactions.

1) Top-down integration: Develop the skeleton of the system and populate it with components.
2) Bottom-up integration: Integrate infrastructure components then add functional components.

C. System Testing

This testing is done after integration testing. This testing is done by development teams.

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic.

1) To verify that the system components perform control functions
2) To perform inter-system test
3) To demonstrate that the system performs both functionally and operationally as specified
4) To perform appropriate types of tests relating to Transaction Flow, Installation, Reliability.

D. Acceptance Testing

It is a testing technique performed to determine whether or not the software system has met the requirement specifications. The main purpose of this test is to evaluate the system's compliance with the business requirements and verify if it is has met the required criteria for delivery to end users.

Objective of Acceptance Testing: To verify that the system meets the user requirements.

IV. SOFTWARE TESTING LIMITATIONS

Limitation is a principle that restricts the extent of any application. Software testing has also few limitations that should be considered to set realistic expectations about its benefits. Inspite of being most widely used verification technique, software testing as various following limitations:

1) Testing can be used to show the presence of errors, but never to show their absence.
2) We use testing to disclose many hidden errors but this methodology never guarantees the absence of errors. It is only used to identify the known errors. It never gives any information about those defects which remain uncovered
3) Testing cannot establish that a product functions properly under all conditions but can only establish that it does not function properly under specific conditions.
4) Software testing does not help in finding root causes which resulted in injection of defects in the first place. Locating root causes of failures can help us in preventing injection of such faults in future.

V. CONCLUSION

This article on Software testing discusses about software testing, objectives of software testing, principles. It further discusses about features of a good tester and a test case. To perform testing effectively and efficiently, person involved with testing should be familiar with basic software testing goals, principles, limitations and concepts. Further different Software testing methods i.e. unit testing, integration testing, system testing and acceptance testing are described. Implementing testing principles in real world software development, to achieve the testing objectives to maximum extent keeping in consideration the testing limitations will validate the work and will become a path for future research.

REFERENCES

[1] Ian Somerville, Software Engineering, Addison-Wesley, 2001.
[2] Myers, Glenford J., ―The art of software testing, New York: Wiley, c1979. ISBN: 0471043281
[3] Miller, William E. Howden, "Tutorial, software testing &validation techniques", IEEE Computer Society Press, 1981.
[4] B. Agarwal et al., “Software engineering and testing”. Jones & Bartlett Learning, 2010.
[5] R.W. Miller ,”Acceptance Testing”,2001, http://www.dsc.ufcg.edu.br/~jacques/cursos/map/recursos/Testing05.pdf)
[6] Roger S. Pressman, “Software engineering: A practitioner’s Approach,” fifth edition, 2001.
[7] E. F. Miller, “Introduction to Software Testing Technology”, Software Testing & Validation Techniques, IEEE, 1981, pp. 4-16.

