Software Testing Techniques in Software Development Life Cycle

D. Meenakshi, J. Swami Naik, M. Raghavendra Reddy

Computer Science Department,
G. Pulla Reddy Engineering College
Kurnool, Andhra Pradesh, India

Abstract: Software testing provides a means to minimize errors, cut maintenance and decrease overall software costs. Different software development and testing methodologies, tools, and techniques have emerged to enhance software quality. At each and every phase of software development life cycle testing is performed with different tools and techniques. One of the major problems within software testing area is how to get a suitable set of cases to test a software system. The software should assure maximum effectiveness with the least possible number of test cases. At present there are many testing techniques available for generating test cases.

Keywords: Software Testing, Level of Testing, Testing Technique, Testing Process

I. INTRODUCTION

Software testing is as getting on as the hills in the past digital computers. Testing the software is the only means for assessing the quality of software. In view of the fact that testing typically consumes 40 - 50% of effort in development, and requires more effort for systems that require more reliability, it is an important part of the software engineering. Contemporary software systems must be more reliable and correct. Automatic methods are used for ensuring software correctness which range from static techniques, such as (software) model checking or static analysis, to dynamic techniques, such as testing. All these testing techniques have strengths and weaknesses: model checking (with abstraction) is automatic, exhaustive, but may suffer from scalability issues. Static analysis, on the other hand, scales to very large programs but may give too many fake warnings, while testing alone may miss significant errors, since it is intrinsically incomplete.

Software Testing:

Software testing is not only error detection; Testing software means operating the programs under controlled conditions, to (1) check whether it behaves “as specified”; (2) to detect errors, and (3) to validate that what has been specified is what the user actually wants.

Activities in software testing

1. Verification is defined as checking or testing of items, including software, for conformance and reliability by evaluating the results against requirements specified by the user. [Verification: Are we building the product right?]
2. Error Detection: Testing makes an attempt to go things wrong and to determine if things happen when they shouldn’t or things don’t happen when they should.
3. Validation looks at the system correctness – i.e. is defined as checking that what has been specified is what the user actually wanted. [Validation: Are we building the right product?]

The purpose of testing is verification, validation and error detection in order to detect problems – and the purpose of finding those problems is to get them set. Most generic Software problems are: Inadequate software performance, Data searches that yields incorrect results, Incorrect data entry & unsuccessful data edits, wrong coding / implementation of business rules, Incorrect calculation, ineffective data edits, Incorrect processing of data relationship, Incorrect or inadequate interfaces with other programs, Inadequate performance and security controls, Incorrect file handling, insufficient support of business needs, Unreliable results or performance, Confusing or misleading data, Software usability by end users & Obsolete Software, Inconsistent processing.

Terminology:

- Mistake – A manual action done which produce an incorrect result.
- Fault (or Defect) – An incorrect step, process, or data definition in a program.
- Failure – The lack of ability of a system or component to execute its Required function within the precise Performance requirement.
- Error – The variation between a computed, observed, or Metric or condition and the true, specified, or theoretically correct value or condition.
- Specification – A document that specifies in a complete, accurate, confirmable manner, the requirements, design, performance, or other feature of a system or component, and often the Procedures for determining whether these provisions have been satisfied. We observe errors, which can often be connected with failures. But the decisive cause of the fault is often very hard to find.
II. OBJECTIVES:

- A good test case is one that has a probability of detecting an error.
- A superior test is not redundant.
- A successful test is one that uncovers a yet undiscovered error.
- A good test supposed to be “best of breed”.
- To check if the method does what it is predictable to do.
- To verify if the system is “Fit for function”.
- To confirm if the system meets the requirements and be executed successfully in the Intended environment.
- Executing a program with the intention of finding an error.

Testing type | Specification | Scope | Opacity       | Who do it?       \\
-------------|---------------|-------|---------------|-----------------\
Unit         | Low level design Actual code | Classes | White box | programmer |
Integration   | Low level and high level design | Multiple classes | White and Black box | programmer |
Function      | High level design | Whole product | Black box | Independent test group |
System        | Requirement analysis | Whole product in environment | Black box | Independent test group |
Acceptance    | Requirement analysis | Whole product in environment | Black box | Customer |
Beta          | Adhoc | Whole product in environment | Black box | Customer |
Regression    | Changed software | Whole product in environment | White and Black box | Independent test group programmer |

III. SOFTWARE TESTING LIFECYCLE – PHASES

- Testing Cycle starts with the study of user’s requirements.
- Understanding of the requirements is very essential for testing the product.
- Component Identification
- Test Specification Design
- Test Specification Review
- Code Review, Test execution and evaluation
- Performance and simulation Testing

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

Test Process Analysis
- Analysis done on the reports and improving the application’s performance by implementing new technology and additional

IV LEVELS OF TESTING:

Unit testing:
- The initial level of testing.
- Tests done on particular methods or components.
- Requires information about internal program design and code.
- Test will be done by Programmers (not by testers).
Objectives

To test the function of a program or unit of code such as a program or module, To test internal logic, To verify internal design, To test path & conditions coverage, To test exception conditions & error handling.

When

After modules are coded

Who

Developer

Input

Internal Application Design, Master Test Plan, Unit Test Plan

Output

Unit Test Report

Methods

White Box testing techniques, Test Coverage techniques

Tools

Debug, Re-structure, Code Analyzers, Path/statement coverage tools

Incremental Integration Testing

- It is done based on design and Continuous testing of an application as and when a new functionality is added.
- Application’s functionality aspects are required to be independent enough to work separately before completion of development.
- Done by programmers or testers.

Top-down integration:

- Develop the skeleton of the system and integrates it with components.

Bottom-up integration:

- Integrate infrastructure components then add functional components.
- To simplify error localization, systems should be incrementally integrated

Objectives

To technically verify proper interfaces between modules, and within sub-systems

When

After modules are unit tested

Who

Developer

Input

Internal & External Application Design, Master Test Plan, Integration Test Plan

Output

Integration Test report

Methods

White and Black Box techniques, Problem / Configuration Management

Tools

Debug, Re-structure, Code Analyzers

System testing:

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

Acceptance testing:

To check whether the software meets customers requirements or not.

Beta testing:

Testing the software under the customers environment.

Regression testing:

Testing done to the software after making the changes or enhancements

V. TEST PLAN

Purpose of preparing a Test Plan

- Validate the acceptability of a software product.
- Help the people outside the test group to understand „why” and „how” of product validation.
- A Test Plan should be thorough enough (Overall coverage of test to be conducted)

Scope

- The areas to be tested by the QA team.
- Specify the areas which are out of scope (screens, database, mainframe processes etc).

Test Approach

- Details on how the testing is to be performed.
- Any specific strategy is to be followed for testing (including configuration management).

VI. CONCLUSION:

Testing can show the presence of faults in a system; it cannot prove that there are no remaining faults. Component developers are dependable for component testing; system testing is the responsibility of a separate team. Integration testing is testing increments of the system; release testing involves testing a system to be released to a customer. Use experience and guidelines to design test cases in defect testing. Interface testing is designed to discover defects in the interfaces of composite components. Equivalence partitioning is a way of discovering test cases - all cases in a partition should behave in the same way. Structural analysis relies on analyzing a program and deriving tests from this analysis. Test automation reduces testing costs by supporting the test process with a range of software tools.

REFERENCES

[3]. Effective Software Testing, by E. Dustin
[4]. Software testing, by Ron Patton
[5]. Software engineering, by Roger Pressman
[6]. http://people.engr.ncsu.edu/txie/testingresearchsurvey.htm
[7]. http://www.engpaper.com
[8]. http://www.people.engr.ncsu.edu/txie/testingresearchsurvey.htm
[10]. www.findwhitepapers.com