Software Testing: A Review

Ritu Choubisa Pankaj Dalal

M.Tech (S.E.) Scholar *Associate Professor*

Shrinathji Institute of Technology & Engineering, Nathdwara-313301

Abstract

Software testing is a critical phase after specification, design & coding. Software testing is the process of the software quality assurance & represents ultimate product review. It is very important for developing error free functionality & correctness of software for client requirement. Software testing is one of the phase of SDLC to ensure free & quality Software. Here we present a review about different phases of software testing.

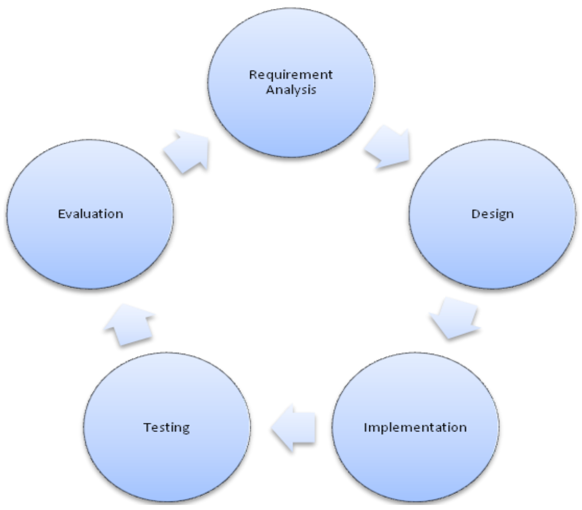
Keywords

Software Testing, SDLC (Software Development Life Cycle), STLC (Software Testing Life Cycle Stages).

I. Introduction

Software testing refers to process of evaluating the software with intention to find out error in it. Software testing is a technique aimed at evaluating an attribute or capability of a program or product and determining that it meets its quality. Software testing is also used to test the software for other software quality factors like reliability, usability, integrity, security, capability, efficiency, portability, maintainability, compatibility etc. In simple words, software testing is an activity to check whether the actual results match the expected results and to ensure that the software system is defect free. [1]

The Systems development life cycle (SDLC) is a process used by a systems analyst to develop an information system, training, and user (stakeholder) ownership or Is the model of developing and maintaining information systems. The SDLC aims to produce a high quality system that meets or exceeds customer expectations, reaches completion within times and cost estimates, works effectively and efficiently in the current and planned Information Technology infrastructure, and is inexpensive to maintain and cost-effective to enhance. [1]



"Figure1. SDLC system/software development life cycle [1]"

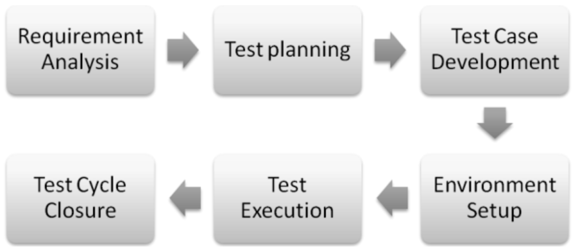
Requirement gathering and analysis is the main focus of the project managers and stake holders. These requirements are analyzed after requirement gathering for their validity and the possibility of incorporating the requirements in the system development. Finally, a Requirement Specification Artifacts is produced which is the guideline for the next phase of the model. The physical system is designed with the help of the logical design prepared by system analysts under the guideline of Requirement Specification Artifacts in Design phase which including the probable output. The Implementation/Coding is the longest phase based on system design Artifacts. The work is divided in modules/units since the code is produced; it is the main focus for the developer among all the phases of the SDLC. The next is Testing of implemented code to verify and validate to insure Quality. The Release & Maintenance is last phase of SDLC after successful testing. The product is delivered / deployed to the customer for their use. [1]

I. Software Testing Life Cycle

Testing Principle is the rule or method in action that has to be followed. The Different testing principles are as follows:

- Testing shows presence of defects.
- Exhaustive testing is impossible.
- Early Testing.
- Defect Clustering.
- Pesticide Paradox.
- Testing is context dependent.
- Absence of errors – fallacy.[1]

The process of testing software in a well-planned and systematic way is known as software testing lifecycle (STLC). STLC refers to a comprehensive group of testing related actions specifying details of every action along with the specification of the best time to perform such actions. There cannot be a standardized testing process across various organizations, however every organization involved in software development business, defines & follows some sort of testing life cycle. [1]



“Figure 2. Phases of STLC”

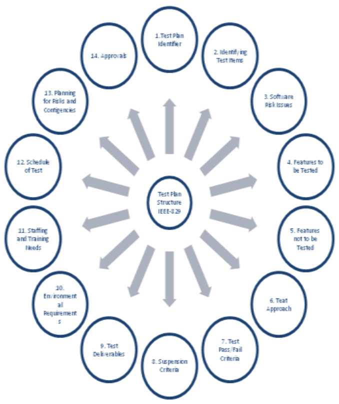
a. Requirement Analysis.

This is a very important phase in STLC. This focuses on understanding the testing requirements of the system. In this phase we finalized the testing types & techniques to be performed, feasibility for automation testing implementation, etc. [1] [2] [3]

b. Test Planning

Josh Probert says that the level of test plan defines what the test plan is being created. For example Subsection of test planning. He also said that the level of test plan should describe be able to content of the document. Josh Probert also mention test plan Structure shown in figure 3 used in this IEEE829 Standards. [2] [3][5]

Yuri chernak said that test plan complexity depends on functional Specification. The test cases cannot completely cover system functionality if functional Specification does not define functional features completely. Which reduces effectiveness of test plan. [3][7][8]


1.Test Plan Identifier
2. Identifying Test Items
3. Software Risk Issues
4. Features to be Tested
5. Features not to be Tested
6. Teat Approach
7. Test Pass/Fail Criteria
8. Suspension Criteria
9. Test Deliverables
10. Environmental Requirements
11. Staffing and Training Needs
12. Schedule of Test
13. Planning for Risks and Contigencies
14. Approvals
Test Plan Structure IEEE-829

"Figure 3. Test plan structure. [5][6]"

c. Test Case Development

Yuri has mentioned that one can use following test design techniques for identifying test cases:-

- Decision table.
- Equivalence partitioning.
- Boundary value, etc.

A test case specification should cover all the necessary test conditions like negative test cases should also be included. The test case specification could also be incorrect, if the functional specification is incorrect or unclear.
Yuri has also mention the technique for validation of test cases for improving effectiveness of test process which can be applied to any project model whether incremental model or evolutionary model.[7][8]

d. Test Environment Setup

Test Environment is the actual system/environment/setup where the testing the application. The Test environment is depends on system architecture, software & hardware requirements, etc. [7] [8]

e. Test Execution & Bug Reporting

Eduardo & Paulo has given that test execution complexity based on the use of test case specification written in a controlled natural language. This knowledge is especially useful when planning the test resources & test suites, where one important criterion is the effort to execute tests. Their special model is based on system specification. They estimate the effort required to perform more activities compare to test execution, for defining & implementing test cases. It cannot be used to estimate the execution effort of a given test case. The use of a controlled natural language reduces the ambiguity. It helps to measure the complexity by the number of possible ways to describe the same test step. It describes minimal and a small but concise in controlled language for high number of different test cases. [10][11]

The complexity evaluation of test steps is recorded. Whenever possible it is reused to evaluate complexity of other test cases, which tends to decrease the number of necessary evaluations over the period of time. [11][12]

This way they extend their work of test execution effort estimation, similar to COCOMO Model. The better precision of this new test execution estimation model depends on test environment process and team experiences. [11][12][13]

f. Test Closure. When the testing team is confident that all the reported bugs are resolved and the system is ready according to the requirement specification, test closure reports is prepared. The proper analysis and documentation is done for the major or critical bugs on closure for handling efficiently and effectively in future projects. [12][13]

III. Conclusion. The process of testing software in a well-planned and systematic order always helps for better product quality. Hence six phases of STLC are followed for consistent and reliable testing. There are many case were the planning is not done due to lake of time or cost reduction or many time proper test cases are not generated. The short of schedule to achieve target time omitting one or two phases stake the product quality. The most crucial phase is test generation where number of test cases are more than few or a few test case are taken for testing. This may lead to inferior quality of product though we have followed all the steps of STLC.

IV References.

[1] Sahil Batra and Dr. Rahul Rishi,"IMPROVING QUALITY USING TESTING STRATEGIES," Journal of Gobal Research in Computer Science, Volume 2, No.6, June 2011.

[2] Josh Probert,"Test Planning", 2009.

[3] Mustafa, K, Khan, R.A. (2007) "Software Testing Concepts and Practices." Oxford: Alpha Science International pp. 194-211.

[4] Pezzè, Mauro, Young, Michal (2008) "Software Testing and Analysis: Process, Principles and Techniques." NJ: Wiley pp. 375-389.

[5] Coley Consulting (2009) IEEE 829 Documentation. http://www.coleyconsulting.co.uk/IEEE829.htm [viewed 20/11/2009].

[6] Software Engineering Technical Committee of the IEEE Computer Society, USA (1998) IEEE 829-1988 Standard for Software Test Documentation. 0-7381-1443 X

[7] Yuri chernak, "Validating & improving test case effectiveness."

[8] Y. Chernak,"Approach to the Function Test Decomposition & Management," Proc.15 Pacific Northwest s/w Quality Conf, PNSQC/Pacific Agenda, Portland, 1997, pp 400-418

[9] IEEE Std.1012-1986, IEEE Standard for S/w Verification & Validation Plans, IEEE, Piscataway, N.J., 1986

[10] Eduardo Aranha, Paulo Borba, "Measuring Test Execution Complexity"

[11] Boehm, B., ET. All. Software Cost Estimation with "COCOMO II. Prentice Hall, 2000".

[12] Garmus, D., Herron, D." Function Point Analysis, Measurement Practices for Successful SoftwareProjects". Addison Wesley, 2001.

[13] Leitão, D., Torres, D. and Barros, F. Nlforspec: "Translating natural language descriptions into formal test case specifications". 2006