ISSN No. 0976-5697

Volume 1, No. 3, Sept-Oct 2010

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Software Testing

Sarbjeet Singh*, Sukhvinder singh, and Gurpreet Singh
M. Tech Student
Sri Sai College of Engineering and Technology, Pathankot, Punjab, India
sarbaish@gmail.com, sukhaish@gmail.com, chouhan87@gmail.com

Abstract: Software goes through a cycle of software development stages. A software is envisioned, created, evaluated, fixed and then put to use. To run any software consistently without any failure/bug/error, the most important step is to test the software. This paper points various types of software testing(manual and automation), various software testing techniques like black box, white box, gray box, sanity, functional testing etc. and software test life cycle models (V-model and W-model). This paper tries to solve the misconceptions of those who think that testing is to be done only after the coding phase, but in reality it is to be associated with each and every phase of software life cycle models.

Keywords: Software, manual, automation, life cycle, functional testing

I. INTRODUCTION

Any software is of high quality if it is error free, user friendly and it provides client satisfaction. And to improve qualities software testing is required. Software testing is the process of execution of a program with the intent of finding errors. There can be various types of errors in any software like specification error, design error, statement error, input error, test error, hardware error etc.. to test and remove all these types of error various types of testing like black box testing, white box testing and gray box testing can be done. Any software which is to be prepared has to go through certain life cycle model. Section 2 describes life cycle models of any software and their limitations. Section 3 describes software test life cycle models i.e. V-Model and W-model.

II. LIFE CYCLE MODELS OF ANY SOFTWARE

Software development life cycle is a time gap between conceptualization of software to the release of the software.

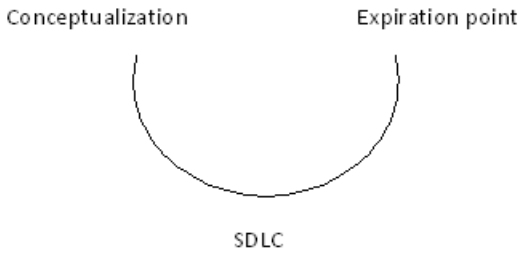

Figure 1- Software Life Cycle.

SDLC is divided into various phases. these can be (a) Feasibility study (b) Requirement analysis and specification, (c) Designing, (d) Implementation or Coding, (e) Testing, (f) Operations, (g) Maintenance. These stages can be motivated to change due to clients if there need changes and the second factor which can change it can be due to situation and also change with various SDLC models. Various SDLC models can be-

a) Waterfall model.
b) Quick fix model.
c) Iterative enhancement model.
d) Evolutionary development model.
e) Spiral model.
f) RAD model.
g) Prototype model.

A. Advantages of these Models

These are the sequential development models. So the team involved for the development is always synchronized with the process. And requirements are gathered in the early phase. After completion of one phase next phase begins. These models provide a valid document for each and every phase. Figure 2 shows various phases of software life cycle models with documents released:

B. Disadvantages of these Models

Testing is included very lately. In U.S. survey which says that if errors are found in requirement phase cost is less and if the errors are found in the operation phase then cost will be more. It is not possible to collect all requirements in the beginning of the project. It is a rigid model in which we cannot go back because requirements are freezed.

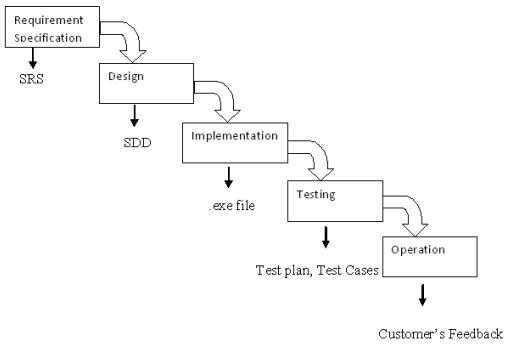

Figure 2- Phases of waterfall model with the document released at each phase.

III. V-MODEL

These are the reflected changes in waterfall model. For testing V-Model is the best model.

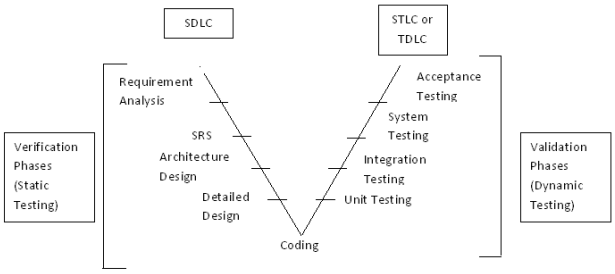

Figure-3 V-Model or TDLC or STLC

In waterfall model we enter into testing after a time span and it is to be done for a specified time span and it is to be done for a specified time span whereas in V-Model, testing starts with 1st phase of SDLC i.e requirement analysis. In V-Model testing is the largest phase of life cycle. In this model testing includes 40-50% cost. This model is both a sequential and testing life cycle model in which both the development and testing activities are executed parallel. Coding is the phase which joins development and testing on the product. The workflow of SDLC is from top to bottom whereas in STLC it is from bottom to top. When the development team is involved in requirement phase testing team will be included in acceptance test criteria. After acceptance test criteria the test will come to review the requirements and then if there is any error in requirements then the tester is going to detect the origin of all errors. Then after that test team will do system test plan then review SRS. Similarly in case of Integration test plan and unit test plan. And while coding there will be code walk through. The first four phases of SDLC will provides only documents so testing done on these documents is called as static testing. And the testing done on interfaces is called as dynamic testing.

A. Levels of testing in V- Model

The testing process is done in four levels-

a) Unit Testing.
b) Integration Testing.
c) System Testing.
d) Acceptance Testing.

B. Software Test Life Cycle (STLC)

It is a time period in which all the testing activities are accomplished.

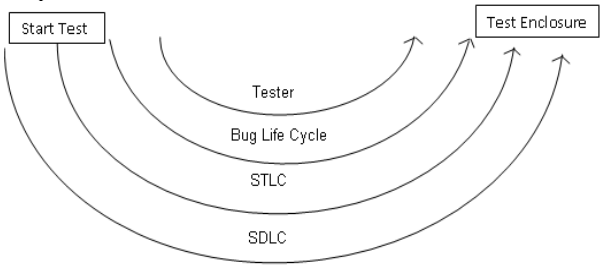

Figure 4- Software life cycle

SDLC and TDLC are to be performed parallel but TDLC is a part of SDLC. Tester is dependent on bug life cycle, Bug life cycle is dependent on test development life cycle, TDLC is dependent on SDLC.

C. Various phases of TDLC/STLC

Various phases of test development life cycle are as follows:

a) Test Specification.
b) Test environment setup.
c) Test planning.
d) Test design.
e) Test automation.
f) Test execution.
g) Test tracking and reporting.
h) Retesting and regression testing.
i) Test closure.

D. Bug Life cycle

A bug is detected by the tester and its status will be set as new by the tester and he will decide the severity of bugs(either critical, major or minor bug). Bug life cycle can be categorized in three categories:

1) If TL rejects the bug.
2) Developer says insufficient information.
3) After retesting/regression testing the result is unsatisfactory then the bug reopened then again go to TL, PM and developer.

E. Test Plan

For a test plan project plan and SRS is needed. Test plan is a document which consists of all the testing related actions. It contains scope of testing and approach used in testing as well as who will do testing, what is to be tested and how to accomplish testing all are described in test plan. For formal testing test plan must be prepared. Test plan creation is based on following factors:

- Master test plan.
- Unit test plan.
- Integration test plan
- System test plan.
- Acceptance test plan.
- Performance test plan.

If project is more risky then all the test plan criteria is necessary to be created. Responsibility of test plan creation is of test manager or team lead. Items included in test plan are: introduction of project, scope, items to be tested, items not to be tested, scheduling, roles and responsibilities, entry and exit criteria, suspension and resumption criteria, approvals bug severity, bug priority etc..

F. W-Model

This model is also known as v & v model or verification and validation model.

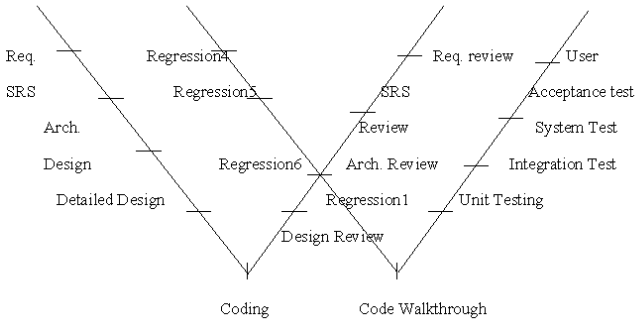

Figure5 W-MODEL

IV. CONCLUSION

With this paper we conclude that for every project a proper life cycle is to be followed which follows particular life cycle stages like unit test, integration testing, system testing and acceptance testing. And while following v-model particular test criteria is followed with every testing phase. Various life cycle models for testing are V-model and W-model (v & v model). In our further research we will study automation tools used for testing.

V. REFERENCES

[1] James Bach (2003a), "Exploratory Testing Explained", www.satisfice.com/articles/et-article.pdf (accessed March 30, 2003).

[2] Bach, James (2003b) Rapid Software Testing (course notes). www.testingeducation.org/coursenotes/bach_james/cm_200204_rapidtesting/ (accessed March 20, 2003).

[3] Berger, Bernie (2001) "The dangers of use cases employed as test cases," STAR West conference, San Jose, CA. www.testassured.com/docs/Dangers.htm. accessed March 30, 2003.

[4] Bittner, Kurt & Ian Spence (2003) Use Case Modeling, Addison-Wesley

[5] Buwalda, Hans (2000a) "The three holy grails of test development," presented at EuroSTAR conference.

[6] Buwalda, Hans (2000b) "Soap Opera Testing," presented at International Software Quality Week Europe conference, Brussels. Cockburn, Alistair (2000) Writing Effective Use Cases, Addison-Wesley.

[7] Collard, R. (1999). "Developing test cases from use cases," Software Testing & Quality Engineering, July/August, p. 31.

[8] Constantine, Larry, L & Lucy A.D. Lockwood (1999) Software for Use, ACM Press.

[9] El-Far, Ibrahim K. (1995) Automated Construction of Software Behavior Models, M.Sc. Thesis, Florida Tech, www.geocities.com/model_based_testing/elfar_thesis.pdf (accessed March 28, 2003).

[10] El-Far, Ibrahim K. (2001) "Enjoying the Perks of Model-Based Testing", STAR West conference, San Jose, CA. (Available at www.geocities.com/model_based_testing/perks_paper.pdf, accessed March 25, 2003).

[11] El-Far, Ibrahim, K; Thompson, Herbert; & Mottay, Florence (2001) "Experiences in Testing Pocket PC Applications," International Software Quality Week Europe, www.geocities.com/model_based_testing/pocketpc_paper.pdf (accessed March 30, 2003).

[12] Hendrickson, Elisabeth (2002) Bug Hunting (course, notes unpublished)

[13] Graham, Dorothy; & Fewster, Mark (1999) Software Test Automation: Effective Use of Test Execution Tools, ACM Press / Addison-Wesley.

[14] Houghtaling, Mike (2001) Presentation at the Workshop on Model-Based Testing, Melbourne, Florida, February 2001.

[15] Institute of Electrical & Electronics Engineers, Standard 610 (1990), reprinted in IEEE Standards Collection: Software Engineering 1994 Edition.

[16] Jacobson, Ivar, Grady Booch & James Rumbaugh (1999) The Unified Software Development Process, Addison-Wesley.

[17] Jorgensen, Alan (2003) "Testing With Hostile Data Streams,"

[18] http://streamer.it.fit.edu:8081/ramgen/cs/Spring03/CSE5500-Sp03-05/trainer.smi

[19] Kaner, Cem (1998), "Avoiding Shelfware: A Manager's View of Automated GUI Testing,"

[20] Keynote address, STAR '98 Conference, Orlando, FL., www.kaner.com/pdfs/shelfwar.pdf (accessed March 28, 2003).

[21] Kaner, Cem (2000), "Architectures of Test Automation," STAR West conference, San Jose, CA, www.kaner.com/testarch.html (accessed March 30, 2003).

[22] Kaner, Cem (2002) A Course in Black Box Software Testing (Professional ver), www.testingeducation.org/coursenotes/kaner_cem/cm_200204_blackboxtesting/ (accessed March 20, 2003)

[23] Kaner, Cem; Bach, James; & Pettichord, Bret (2002) Lessons Learned in Software Testing, Wiley.

[24] Kaner, Cem; Falk, Jack; & Nguyen, Hung Quoc (1993) Testing Computer Software, Wiley.

[25] Marick, Brian (1997) "Classic Testing Mistakes", STAR East conference, www.testing.com/writings/classic/mistakes.html (accessed March 17, 2003).

[26] Marick, Brian (undated), documentation for the Multi test tool, www.testing.com/tools/multi/README.html (accessed March 17, 2003).

[27] Musa, John (1998) Software Reliability Engineering, McGraw-Hill.

[28] Nyman, N. (1998), "Application Testing with Dumb Monkeys", STAR West conference, San Jose, CA.

[29] Nyman, N. (2002), "In Defense of Monkey Testing", www.softtest.org/sigs/material/nnyman2.htm (accessed March 30, 2003).

[30] Patton, Ron (2001) Software Testing, SAMS.

[31] Pettichord, Bret (2002) "Design for Testability" Pacific Northwest Software Quality Conference, October 2002, www.io.com/~wazmo/papers/design_for_testability_PNSQC.pdf (accessed March 28, 2003).

[32] Popper, Karl (1992) Conjectures and Refutations: The Growth of Scientific Knowledge. 5th Edittion. Routledge.

[33] Robinson, Harry (2001) Presentation at the Workshop on Model-Based Testing, Melbourne, Florida, February 2001; For more on Robinson's experiences with state-model based testing, see www.geocities.com/model_based_testing/ (accessed March 20, 2003.)

[34] Savoia, Alberto (2000) The Art and Science of Load Testing Internet Applications, STAR West conference, available at www.stickyminds.com.

[35] Tinkham, Andy; & Kaner, Cem (2003) "Exploring Exploratory Testing", STAR East conference, www.testingeducation.org/articles/exploring_exploratory_testing_star_east_2003_paper.pdf (accessed March 30, 2003).