ISSN 2229-5518

Importance of Testing in Software Development Life Cycle

T.Rajani Devi

Abstract— in every organization, testing is an important and valuable phase in the software development life cycle. However, the way it is carried out differs from one organization to another. Software testing has become the part of development and it is better to start testing from the initial stages, in order to avoid difficulty by correcting the bug at the last stage. And the importance of testing in software development life cycle is to improve reliability, performance and other important factors, which may define under SRS (software requirement specification). Customer can wait more for software release, but they don't like to work with defected software**.** It is advisable to carry out the testing process from the initial stages, with regard to the Software Development Life Cycle or SDLC to avoid any complications.

Index Terms— Embarking, Hazard, Mainframe, Metrics, Pitfalls

1 INTRODUCTION

SDLC stands for Software Development Lifecycle and it is the process of developing information system with proper analysis, design, implementation and maintenance. SDLC is said to be equal to layer two of open systems interconnection or OSI model of network communication. This level of protocol assures proper flow of data from one level to another. In any mainframe networks, host mainframe is considered as primary workstation and other devices are known to be secondary workstations. Therefore, SDLC uses primary station and secondary station for its mode of communication. Secondary station has its own address and they are attached to common port, which is known as multi point of multi drop arrangement. SDLC is used for point-to-point communication. This is made use for many remote communications.

There are many major things associated with SDLC. It is considered as the basis of standard data link protocol in ISO, High-level data link control. It also became one of the variations in HDLC. SDLC are efficient protocols and they are used in close network with its own private lines.

During the software development lifecycle errors occur and defects are inevitably introduced.Most organizations are aware of the importance of testing within the software development life-cycle in order to detect and remove these defects. Research has shown that the test process frequently accounts for 40% of the cost of software development. With the growing requirement for high quality and efficiency, it is becoming increasingly important for organizations to improve their software testing.

SDLC serves as a guide to the project and provides a flexible and consistent medium to accommodate changes, and perform the project to meet client's objectives. SDLC phases define key schedule and delivery points which ensure timely and correct delivery to the client within budget and other constraints and project requirements. SDLC co-operates project control and management activities as they must be introduced within each phase of SDLC.

Waterfall is a sequential and non iterative SDLC model which describes flowing of phases downwards one by one. The process does not start a phase unless the previous phase is completed once and for all completely. The waterfall model consists of the following phases:

- Requirements gathering
- Design
- Implementation
- Testing
- Maintenance

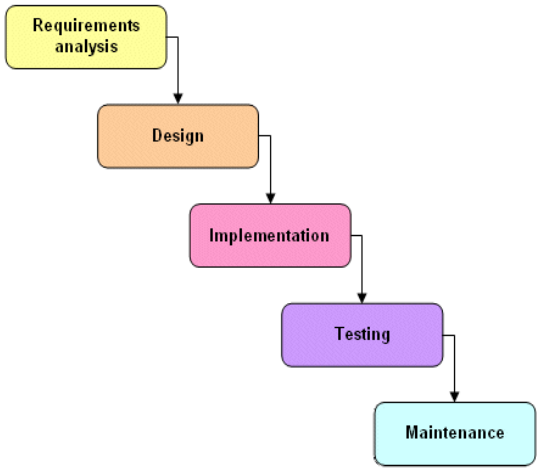

Fig: 1 Waterfall Model

2 Scope of Software Testing

The primary function of software testing is to detect bugs in order to uncover and detect it. The scope of software testing includes execution of that code in various environments and also to examine the aspects of the code - does the software do what it is supposed to do and function according to the specifications? As we move further we come across some questions such as, "When to start testing?" and "When to stop testing?" It is recommended to start testing from the initial stages of the software development. This not only helps in rectifying the errors before the last stage, but also reduces the rework of finding bugs in the initial stages every now and then. It saves time and is cost-effective. Software testing is an ongoing process, which is potentially endless but has to be stopped somewhere, due to the lack of time and budget. It is required to achieve maximum profit with good quality product, within the limitations of time and money. The tester has to follow some procedural way through which he can judge if he covered all the points required for testing or missed out any. To help testers carry out these day-to-day activities, a baseline has to be set, which is done in the form of checklists. In the current culture of software development, a testing organization may be separate from the development team. There are various roles for testing team members. Information derived from software testing may be used to correct the process by which software is developed.

3 Measuring Software Testing

There a rises a need for measuring the software, both, when the software is under development and after the system is ready for use. Though it is difficult to measure such an abstract constraint, it is essential to do so. The elements that cannot be measured need to be controlled. There are some important uses of measuring the software.

Software metrics help in avoiding pitfalls such as:

1. Cost overruns
2. In identifying where the problem has raised
3. Clarifying goals

It answers questions such as:

1. What is the estimation of each process activity?
2. What is the quality of the code that has been developed?
3. How can the underdeveloped code be improved?

It helps in judging the quality of the software, cost and effort estimation, collection of data, productivity and performance evaluation.

4 Phases in Software Development Lifecycle

Testing phase has much importance in SDLC due to a major role in debugging and error correction. The phases of SDLC is being followed in both testing and development cycle of any software application. Here are the phases of SDLC that is being followed:

4.1 Requirements Gathering and Analysis

Under this phase, proper requirements of project are gathered. All close functions are brought in to focus. All kinds of requirements and analysis of user requirement are done in this phase.

4.2 System Design

This is the next phase in SDLC where a rough system design is made. With all data and information being gathered, a system design is made.

4.3 Development

This is the next phase after system design when development of project is made. According to design, proper coding is done to gain that design. Programming language might be selected according to the project.

4.4 System Testing

Just after development phase, testing is carried out to know the outcome of application. Testing is made to know the actual result and the expected result.

4.5 Operations and Maintenance

This is the final stage of SDLC, where the software that is being developed is being distributed to end users who are responsible for maintaining and using it for proper operations. The software that is being developed must be open to any changes being made in coding.

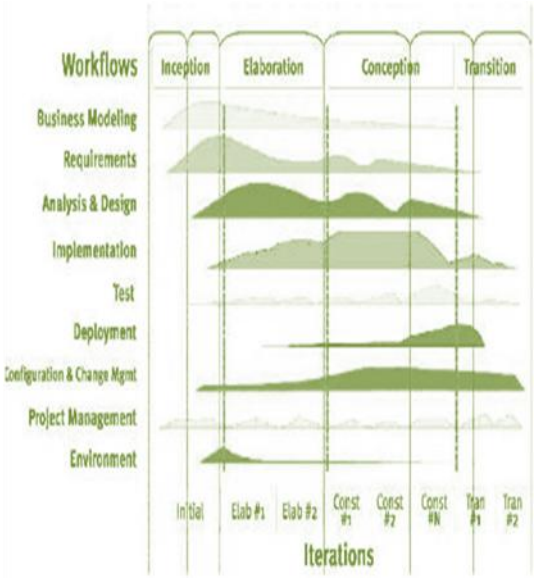

Fig: 2 Phases in Software Development Lifecycle

5 ROLE OF TESTING IN SDLC

5.1 Inception Phase

A test engineer understands the need of project in this phase.

5.2 Elaboration Phase

In this phase, any tester tries to understand how the project is being developed. Requirements are easily made.

5.3 Construction Phase

Developers play an important role and they help developing design of the software. Tester has to know that all requirements are being traced through test cases. System testing and integration testing is necessary in this phase.

5.4 Transition Phase

In this phase if any defects or bugs are found then they are re tested and it goes under regression testing phase. With regression testing, reliable products come out. With help of testing in SDLC, any basic product is transformed in to a strong and reliable product.

6 SOFTWARE DEVELOPMENT LIFE CYCLE MODEL:

There are many kinds of SDLC models and each of these models makes use of testing phase. Hence, this makes testing a very important part in any software development life cycle. With unit testing, integration tests, system testing, regression tests and user acceptance testing and major types of testing, helps any developer to come up with a reliable and trusted web application that can be useful. Testing also follows its own lifecycle like test analysis, test plan, test design and test execution

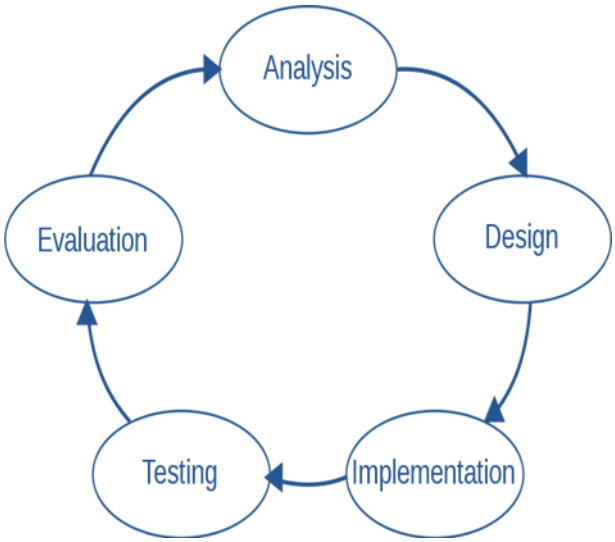

Fig: 3 SDLC Model

System with proper analysis, design, implementation and maintainance.SDLC is said to be equal to layer two of open systems interconnection or OSI model of network communication. This level of protocol assures proper flow of data from one level to another.

In any mainframe networks, host mainframe is considered as primary workstation and other devices are known to be secondary workstations. Therefore, SDLC uses primary station and secondary station for its mode of communication. Secondary station has its own address and they are attached to common port, which is known as multi point of multi drop arrangement. SDLC is used for point-to-point communication. This is made use for many remote communications.There are many major things associated with SDLC. It is considered as the basis of standard data link protocol in ISO, High-level data link control. It also became one of the variations in HDLC. SDLC are efficient protocols and they are used in close network with its own private lines.

7 MAJOR IMPORTANCE OF TESTING IN SDLC

Some major things are necessary in SDLC phase.

7.1 Identification of Bugs and Defects

Testing phase is one phase that determines the bugs and errors in the application. These bugs can be at unit level or system level. With so many testing phases, you get to keep away from any kind of bugs that may be affecting your application.

7.2 Information to stakeholders and reputation of company

With the help of testing phase, it helps you to know the state of product and service quality. The stakeholders get good in-

formation through testing phase about service quality too.

7.3 Improvement in Product Quality

As testing is a phase that helps in knowing the actual outcome and the expected outcome. Surely, it can help you to improve your product quality. With proper testing done, you come out of errors and develop perfect software application for the users.

7.4 Technical Importance

In any SDLC lifecycle, testing phase is important for technical aspect, as it has to come out with best technically sound application.

7.5 To Get ahead of any Competitive Developers

Perfect testing tools and methods help you and your application to grow up in market and keep away your competitors. With getting away in all phases of testing, you develop more sound, safe and secure software application.

7.6 Keep Away any Hazards

When you develop any software, testing plays an important part. When you give your software without any testing then it may be hazardous to the users who are using it. To keep everyone away from any hazard, it is necessary that you go under all testing phases.

7.7 Improved Quality

Proper tested software gives you more confidence of coming up with great software. Secondly, it improves quality of your software application as continuous and all kinds of testing modes have made a secure and safe application that can be used by the end users.

7.8 Verification and validation

One of the main aims of testing phase in SDLC is for verification and validation. Testing is a phase that can serve as a metrics as it is used heavily in V&V method. Based on the result, you could come out with comparison between qualities amongst different products.

7.9 Reliability Estimation

This is yet another important factor that is determined by the testing phase. If your software application has gone through all small level like unit testing and major testing like regression testing and all, then sure it is a reliable application. Hence testing determines reliability of your software application. Testing may serve as the best statistical method that defines operational testing on application resulting in a reliable software product.

7.10 Prove Usability and Operability

One very important aim of software testing is to prove the software is both usable and operable. Usability testing is where the software is released to a select group of users and their working with the product is observed. All aspects of a user's interaction with the software, like ease of use and where users are facing problems, are recoded and analyzed.

7.11 Prevent Defect Migration

The majority of errors are usually introduced in the software requirements gathering phase. If the errors are detected early, they can be prevented from migrating to the subsequent development phase.Early detection and debugging of errors leads to huge savings in software development costs.

7.12 Economic importance

A well-tested software application will have good economic impact. This is because any one would love to go with reliable and trusted application in market.

TABLE 1
Typical Software Quality Factors

Functionality (exterior quality)	Engineering (interior quality)	Adaptability (future quality)
Correctness	Efficiency	Flexibility
Reliability	Testability	Reusability
Usability	Documentation	Maintainability
Integrity	Structure	

Good testing provides measures for all relevant factors. The importance of any particular factor varies from application to application. Any system where human lives are at stake must place extreme emphasis on reliability and integrity. In the typical business system usability and maintainability are the key factors, while for a one-time scientific program neither may be significant. Our testing, to be fully effective, must be geared to measuring each relevant factor and thus forcing quality to become tangible and visible.

Tests with the purpose of validating the product works are named clean tests, or positive tests. The drawbacks are that it can only validate that the software works for the specified test cases. A finite number of tests cannot validate that the software works for all situations. On the contrary, only one failed test is sufficient enough to show that the software does not work. Dirty tests or negative tests, refers to the tests aiming at breaking the software, or showing that it does not work. A

piece of software must have sufficient exception handling capabilities to survive a significant level of dirty tests.

A testable design is a design that can be easily validated, falsified and maintained. Because testing is a rigorous effort and requires significant time and cost, design for testability is also an important design rule for software development.

What are SDLC and STLC? Explain its different phases?

SDLC is software development life cycle. The different phases of SDLC are as follows

- Requirement phase
- Designing phase(HLD, DLD (Program spec))
- Coding
- Testing
- Release
- Maintenance

STLC is software testing life cycle. The different phases of STLC are as follows:

- System Study
- Test planning
- Writing Test Case or scripts
- Review the test case
- Executing test case
- Bug tracking
- Report the defect

What is Entry and Exit Criteria in Software Testing?

The Entry Criteria is the process that must be present when a system begins, like

- SRS - Software Requirement Specifications
- FRS - Functional Reqirement specifications
- Use case
- Test Case
- Test Plan

The Exit Criteria ensures whether testing is completed and the application is ready for release, like

- Test Summary Report
- Metrics
- Defect Analysis Report.

CONCLUSION

As one can see, software testing is a very large undertaking. It is very important to have an effective process in place prior to embarking on any major software development effort. This is essential to the success of the overall effort, and delivering quality software to end-users.

REFERENCES

[1] "Systems Development Life Cycle". In: Foldoc(2000-12-24)
[2] "Systems Development Life Cycle". In: Foldoc(2000-12-24)
[3] McConnell, Steve. "7: Lifecycle Planning". *Rapid Development*. Redmond, Washington: Microsoft Press. p. 140.
[4] QuickStudy: System Development Life Cycle, By Russell Kay, May 14, 2002
[5] SELECTING A DEVELOPMENT APPROACH. Retrieved 27 October 2008.