An Evolution of Testing In Software Engineering

K. Rajasri, T. Godwin Selva Raja

Abstract— **Software Testing is the process of finding bugs in the software & makes the software bug free. In the software development life cycle (SDLC) the Testing plays an important role, which helps to improve the quality, reliability & performance of the system with all checks what all functions software supposed to do & also check that Software is not doing what he not supposed to do. There is the major importance of testing in the part of the SDLC and it is better to introduce testing in the early stage of the SDLC phases so it helps to identify the defects in the early stage & try to avoid the bugs finding & get resolved in the last critical stage. So the development of the software will be very quick and it fulfills the customer satisfaction.**

Index Terms—**Testing, SDLC, STLC, Bug, Software.**

I. Introduction

The internet is quickly growing in today's world. The customer of the internet wants to get products to be implemented and updated faster than their competitors. The customer wants more for software releases with new features to be implemented in a short time frame, but they don't like to work with defects software In the next few days the new version of the product will be released, & gets only a couple of days of testing before it is shipped [1] [8]. So due to this short time frame or continuous releases the more bugs get piled up into the products and which gets fixed in the next release. So there arises the problem. Releasing such software with so many bugs into it may affect the user experience which makes a bad impact on quality impression of your company brand. They will remember about the delivered bad quality product, so there will be Importance of testing, which makes the vital role in the software development life cycle (SDLC) [3]. Usually software testing is considered as one phase of the software development life cycle. There's something to be said for including testing in all phases. We can take other simpler examples to clear why testing is important. In the Bank software think if showing zero instead of thousand in the balance amount field due to bugs in the banking software or in the student mark sheet student got good marks but system showing the incorrect results due to bugs in the student result software [2][4]. If the software will show some error message or notification instead of wrong result in case of a system error that could be a better option to use. So, in the testing, we try to make the software defect free [6]. In the testing process may not fix the entire defect present in the software application or it we cannot say it as software application is 100% error free but taking one step ahead to doing this & provide user friendliness.

***K. Rajasri**, Senior Assistant Professor, CSE Department, Christ College of engineering and technology, Pondicherry University, Pondicherry, India,*
***T. Godwin Selva Raja**, Student M.Tech, CSE Department, Christ College of engineering and technology, Pondicherry University, Pondicherry, India,*

II. Importance Of Testing

Software testing is a process of Verification and Validation to check whether a software application under test is working as expected. To test the application, we need to give some input and check if getting results as per mentioned in the requirements or not. This testing activity is carried out to find the defects in the code & improve the quality of a software application. Testing of application can be carried out in two different ways, Positive testing and Negative testing.

- Testing should be introduced in the early stage of the SDLC, The cost of fixing the bug is larger if testing is not done in early stage & bugs found in later stages.
- In the today's competitive market only the quality product stays longtime firmly, so to make sure they produce the good quality product the testing of application is a key factor in SDLC.
- As it's not possible makes it software application is defect free, but testing will be necessary.
- Most important things about testing are the development environment is different than the Testing environment and the testing done on testing environment is similar to the Production environment.

After all, for the growth of any business the most important user satisfaction & testing plays a key role in to make this happen. But to make this happen, you have to plan it properly before executing it. In the "Test Planning" we can cover how making plans to test the software application effectively & more efficiently. Stay tuned for more updates

A. Positive Testing

Positive Testing is testing process where the system validated against the valid input data. In this testing, tester always check for only valid set of values and check if a application behaves as expected with its expected inputs. The main intention of this testing is to check whether software application not showing error when not supposed to & showing error when supposed to. Such testing is to be carried out keeping positive point of view & only execute the positive scenario. *Positive Testing* always tries to prove that a given product and project always meets the requirements and specifications. Under Positive testing is testing the normal day to day life scenarios and checks the expected behavior of the application.

B. Negative Testing

Negative Testing is testing process where the system validated against the invalid input data. A negative test checks if an application behaves as expected with its negative inputs. The main intention of this testing is to check whether software application not showing error when supposed to & showing error when not supposed to. Such testing is to be carried out keeping negative point of view & only execute the test cases for an only invalid set of input data. Negative testing is a testing process to identify the inputs where the system is not designed or un-handled inputs by providing different invalid. The main reason behind Negative testing is to check the stability of the software application against the influences of different variety of incorrect validation data set.

III. Bug Clearance

The main principal of Software testing of an application is to find important bugs in the software application & try to make software application bug free. In this Software testing classes article I am making things to simpler to testers.

Once a bug is found, this should be communicated to develop. Before reporting the bug make sure that the bug is well documented with steps to repair, conditions under which this bug is occurring, how many time it occurs & the expected result of the bug. The bug report should accurate & complete, so that developers can get the exact failure reason. Based on this developer can get an exact idea of problems faced by user & it helps to resolve the problem accurately. To facilitate this task tester should report the bug & verify that this is a bug & add same repair steps with example & attach screenshots which helps to prove that a bug has been encountered. Also attach the related logs that provide the activities about the time of bug occurrence. While reporting the bug, it should be divided into various categories like GUI, Functional or Business, Navigational, Validation error, etc. which will again help to categorize the bugs in the bug management.
The approach of tester would help a lot to get bug fixed bug quickly & correctly. The main thumb rule is that you (tester) should be confident enough while reporting the bug. Before adding a bug makes sure that you are not adding duplicate bug (which is already logged). Many of the bug tracking systems are helping to identify or prevent to add duplicate bug, which is restricted to adding unnecessary bugs which will help to reduce rework in case of bug management. Along with bug report, adding little extra information would definitely help developers to get exact scenario or steps to understand the problem like types of hardware & software, environment configuration, setup, versions (like Browser version & name) etc.

A. Report the problem as early as possible

While testing if you observed any bug, add this bug in the bug tracking tool immediately, don't wait to write bug in details afterwards. If you are thinking of reporting bugs later than it might be possible that miss few important reproduce steps. Reporting bug immediately will help to write good bug report, which help developers to get an exact idea of problems faced by user.

B. Double crosses check bug before reporting the bug

The thumb rule is the bug should be reproducible using added "Steps to reproduce" in the bug report. If you are thinking that the bug is not reproducing all the time, then it should be reported in the bug by using the field "Reproduces: Sometimes" field in the bug report.

C. Check if the same bug is occurring in some other related module

Much of the time the same issue might be occurring in the other module of the project as well. So might be possible that occurrence of the same issue in another module. Check this before filing a bug & if it is happening, then this should be added in the bug report.

D. Write a good bug summary

Based on the summary of the bug the developers are getting nature of the bug. If the bug summary is not that good enough, then because of such poor quality bugs will unnecessarily increase the bug cycle. The bug summary should be well communicated & good enough to understand developer about exact problem. Also keep in mind that this bug summary will be used to search the bug afterwards for all the bugs reported.

E. Don't make use of offensive language in the bag

It is very nice that you found a bug but it doesn't mean that you use offensive language in bug against the developer or any individual or should not all blame to develop.

F. Before clicking Submit button, please review the bug report

This is good practice to read bug before reporting it. You should check for Title, Summary, and Steps to repro in the bug report. As mentioned in the above point check for if any offensive language used in bug report, which might create a misunderstanding or false impression. Such sentences or words should be kept away from the bug report. Also add Screenshots which helps prove that a bug has been encountered.

IV. Software Testing Life Cycle (STLC)

Software Testing Life Cycle (STLC) is the testing process which is executed in a systematic and planned manner. In STLC process, different activities are carried out to improve the quality of the product. Let's quickly see what all stages are involved in typical Software Testing Life Cycle (STLC). Following steps are involved in Software Testing Life Cycle (STLC). Each step has its own Entry Criteria and deliverable.

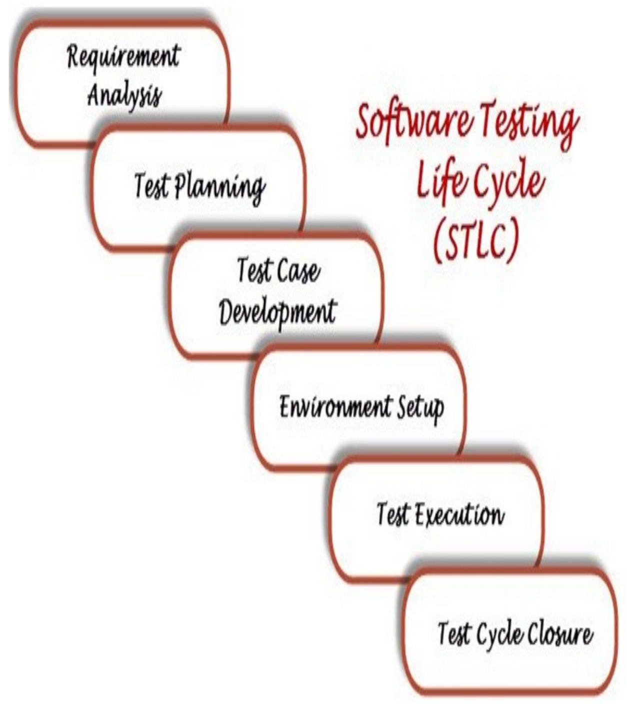

Fig.1 : Software Testing Life Cycle (STLC)

A. Requirement Analysis

Requirement Analysis is the very first step in Software Testing Life Cycle (STLC). In this step Quality Assurance (QA) team understands the requirement in terms of what we will test & figure out the testable requirements. If any conflict, missing or not understood any requirement, then QA team follows up with the various stakeholders like Business Analyst, System Architecture, Client, Technical Manager/Lead etc. to better understand the detail knowledge of requirement.

From very first step QA involved in the where STLC which helps to prevent the introducing defects into the software under test. The requirements can be either Functional or Non-Functional like Performance, Security testing. Also requirement and Automation feasibility of the project can be done at this stage.

B. Test Planning

Test Planning is most important phase of the Software testing life cycle where all testing strategy is defined. This phase also called as a Test Strategy phase. In this phase typically Test Manager (or Test Lead, based on company to company) involved to determine the effort and cost estimates for the entire project. This phase will be kicked off once the requirement gathering phase is completed & based on the requirement analysis, start preparing the Test Plan. The Result of Test Planning phase will be Test Plan or Test strategy & Testing Effort estimation documents. Once the test planning phase is completed the QA team can start with test case development activity.

C. Test Case Development

The test case development activity is started once the test planning activity is finished. This is the phase of STLC where the testing team writes down the detailed test cases. Along with test cases testing team also prepares the test data, if any required for testing. Once the test cases are ready, then these test cases are reviewed by peer members or QA lead.

Also the Requirement Traceability Matrix (RTM) is prepared. The Requirement Traceability Matrix is an industry-accepted format for tracking requirements where each test case is mapped with the requirement. Using this RTM we can track backward & forward traceability.

D. Test Environment Setup

Setting up the test environment is a vital part of the STLC. Basically test environment decides on which condition software is tested. This is independent activity and can be started parallel with Test Case Development. In the process of setting up testing environment test team is not involved in it. Based on company to company may be a developer or customer creates the testing environment. Meanwhile testing team should prepare the smoke test cases to check the readiness of the test environment set up.

E. Test Execution

Once the preparation of Test Case Development and Test Environment setup is completed, then test execution phase can be kicked off. In this phase testing team start executing test cases based on prepared test planning & prepared test cases in the prior step.

Once the test case is passed, then the same can be marked as Passed. If any test case is failed, then corresponding defect can be reported to developer team via the bug tracking system & bug can be linked to the corresponding test case for further analysis. Ideally, every failed test case should be associated with at least single bug. Using this linking we can get the failed test case with bug associated with it. Once the bug fixed by development team then the same test case can be executed based on your test planning.

If any of the test cases are blocked due to any defect, then such test cases can be marked as Blocked, so we can get the report based on how many test cases passed, failed, blocked or not run, etc. Once the defects are fixed, same Failed or Blocked test cases can be executed again to retest the functionality.

F. Test Cycle Closure

Call out the testing team member meeting & evaluate cycle completion criteria based on Test coverage, Quality, Cost, Time, Critical Business Objectives, and Software. Discuss what all went well, which area needs to be improved & taking the lessons from current STLC as input to upcoming test cycles, which will help to improve bottleneck in the STLC process. Test case & bug report will analyze to find out the defect distribution by type and severity. Once complete the test cycle, then test closure report & Test metrics will be prepared. Test result analysis to find out the defect distribution by type and severity.

V. CONCLUSION

The world of the software tester is full of information. Information flows from the application we are testing, the platform and environment the application runs on, and from the development history of the application itself. The way testers consume and leverage this information will ultimately determine how well applications are tested and subsequently the quality levels of the applications that make up the software ecosystem. The only successful future for software testers is one where we master this information and use it in the ways previously prescribed failing to do so will mean the same levels of low quality our industry has historically achieved. This paper addressed all major points of Software Testing Life Cycle (STLC) and the testing importance & Bug Report. So it the quality of the bug report document should be high. Good bug report helps to save time of developer & tester. Take time before reporting the bug, but make sure that you should stick to bonus tips added above. Taking such efforts of reporting good quality bug report strengthen then the relationship between tester & developer.

REFERENCES

[1] Kolawa, Adam; Huizinga, Dorota (2007). *Automated Defect Prevention: Best Practices in Software Management*. Wiley-IEEE Computer Society Press. pp. 41–43. ISBN 0-470-04212-5.

[2] Kolawa, Adam; Huizinga, Dorota (2007). *Automated Defect Prevention: Best Practices in Software Management*. Wiley-IEEE Computer Society Press. p. 426. ISBN 0-470-04212-5.

[3] Parag C. Pendharkara, Corresponding author contact information, E-mail the corresponding author, James A. Rodgerb, Girish H. Subramanian (November 2008). "Information and Software Technology". *Volume 50, Issue 12* (Science Direct): Pages 1181–1188. doi:10.1016/j.infsof.2007.10.019.

[4] "Proceedings from the 5th International Conference on Software Testing and Validation (ICST). Software Competence Center Hagenberg. "Test Design: Lessons Learned and Practical Implications."

[5] "IEEE article on Exploratory vs. Non Exploratory testing". Ieeexplore.ieee.org. Retrieved 2012-01-13.

[6] "Globalization Step-by-Step: The World-Ready Approach to Testing. Microsoft Developer Network". Msdn.microsoft.com. Retrieved 2012-01-13.

[7] Myers, Glenford J. (1979). *The Art of Software Testing*. John Wiley and Sons. pp. 145–146. ISBN 0-471-04328-1.

[8] Gelperin, D.; B. Hetzel (1988). "The Growth of Software Testing". CACM **31** (6). ISSN 0001-0782.