October 2013

TESTING PROCESS OVERVIEW EXPLORATION FOR A SOFTWARE PRODUCT IN A LARGE SOFTWARE DEVELOPMENT PROJECT

SRIHARSHA G

M.S.Ramaiah Institute of Technology, Bangalore, India, sriharsha.guttal@gmail.com

NARESH E

M.S.Ramaiah Institute of Technology, Bangalore, India, nareshkumar.e@gmail.com

Follow this and additional works at: <https://www.interscience.in/ijcsi>

Part of the [Computer Engineering Commons](#), [Information Security Commons](#), and the [Systems and Communications Commons](#)

Recommended Citation

G, SRIHARSHA and E, NARESH (2013) "TESTING PROCESS OVERVIEW EXPLORATION FOR A SOFTWARE PRODUCT IN A LARGE SOFTWARE DEVELOPMENT PROJECT," *International Journal of Computer Science and Informatics*: Vol. 3 : Iss. 2 , Article 6.

DOI: 10.47893/IJCSI.2013.1126

Available at: <https://www.interscience.in/ijcsi/vol3/iss2/6>

This Article is brought to you for free and open access by the Interscience Journals at Interscience Research Network. It has been accepted for inclusion in International Journal of Computer Science and Informatics by an authorized editor of Interscience Research Network. For more information, please contact sritampatnaik@gmail.com.

TESTING PROCESS OVERVIEW EXPLORATION FOR A SOFTWARE PRODUCT IN A LARGE SOFTWARE DEVELOPMENT PROJECT

SRIHARSHA¹ G & NARESH E²

^{1,2}M.S.Ramaiah Institute of Technology, Bangalore, India
E-mail : sriharsha.guttal@gmail.com, nareshkumar.e@gmail.com

Abstract - Software testing is a process where you have the stated requirements, pre-requisite software that is to be installed and the configuration of the machine that need to be done as per the test data for the test case and for a specific platform which when verified works perfectly. This paper describes about Manual test case process overview in the testing life cycle for a large software development project and how you track the effort for the same. A Tester need to know about the fundamentals of testing concepts. Documentation of the testing concepts and the process overview of the testing life cycle for a software product need to be done initially so that a new tester when recruited can understand the basic flow as to how it is carried out in real time in an organization practically. Initially a tester need to execute the manual test case for few platforms and some tougher manual test cases need to be learnt from the tester who is having some experience. The tougher manual test cases need to be identified initially from the experienced tester who will be giving a session for the execution of such test cases. So when the foundation for a new tester is powerful the plan assigned for the execution of test cases is met. The effort tracking for the testing can be captured for a tester and should be matched with planned effort for the execution of test cases. Measurement of the testing effort plays a very important role as to how comfortable a new tester is and further root cause analysis is done if required when the new tester does not meet the planned effort.

Keywords - software testing, system acceptance testing, confidence testing, problem list entry, exploratory testing

I. INTRODUCTION

Software testing is part of system development life cycle wherein quality is determined by estimating the software product. Miller's goal for Testing is described as follows [12] :

“The general aim of testing is to affirm the quality of software system by systematically exercising the software in carefully controlled circumstances”.

Edsger Dijkstra, a Dutch computer scientist and a major contributor in the development of software engineering community stated “Testing can only show the presence of errors, not their absence”.

Miller's and Dijkstra quotations on testing suggest that a tester should always be looking for errors. The objective of testing done at every stage of software life cycle will be different in nature. However understanding of the requirements and thorough knowledge of the domain is very important and without that the functionality of the program as designed cannot be captured. Software testing process in general need to be addressed with a plan. This plan indicates the various testing techniques that is to be carried out for the product under different platforms. Therefore different testing techniques under theoretical knowledge need to be well-defined and the practical testing which is carried out need to be demonstrated. Testing is a process of verifying that a system works as specified in its stated requirements [1]. In the book “The Art of Software Testing” written by Myers in 1979 transferred information about the design of the testing techniques. So the manual test case when written was given important when there is presence of error in the release of the

software product. In another book Software Testing Techniques [13] which consist of different testing techniques briefs about the definition and theoretical plan which is to be carried out and addressed so that in Industrial Software Development the practicality in implementing those theoretical techniques can be carried out easily. Test case management system is based on the guidelines provided by the IEEE 829 standard. Enhancements can be easily done to Test case management system because of its object-oriented design [2, 3]. Test case management system encourages a framework for the administration of tests by providing uniformity in Test Case Formats as well as Test Results Formats which are based on the IEEE 829 standard [4].

Fig. 1[9] - Hierarchy Of Test Storage

A. Hierarchy of Test Storage [9]

In this Hierarchy of Test Storage Test Case Management System stores test cases. This Hierarchy gives a three level structure for Test Case Management System to store. The structure consists of Product Release Version, Platform and Sub Platform. Product Release Version refers to the release version in which the tester is working. Platform refers to the functional area that is under test for the given system and Sub Platform refers to the grouping of tests within a Platform. Figure 1 below demonstrates separate sections based on Product Release Version, Platform and Sub Platform [9]. Test Set represents a directory where all the Product Release Version of the test cases are stored. A test case in Test Set is defined as a test case description file that follows the test case format rules based on the IEEE 829 standard and other related files that are referred to from within the test case [9]. Thus a test case consists of many files. A test suite or test set can be formed by a logical grouping of tests [9]. This enables the tests to be referred to as a group [9].

II. DESCRIPTION

This paper describes about the process overview of the software product for the testing team in industrial software development. Detailed process overview ensures the team to follow the steps and acknowledge to the team leads after every sub-process is successfully completed. Documentation of the processes and procedures must be done and it should be updated as and when required. The flow of process overview of a product for the testing team is as follows:

A. System Acceptance Testing

System Acceptance Testing is a testing method where the team is ready with the machine setup for which the build team provides the release of the software product to be tested on during which the testers execute the tests and report bugs for which a report is sent to the stakeholders and developers on a daily basis with the percentage of tests executed, issues etc.

System Acceptance Testing deals with both functional and non functional requirements wherein requirements for each test case in the test set need to be traced and well-documented. These test cases most probably deal with User interface screen by making an data entry with proper messages of successful entry or error messages in the status bar as the expected behaviour.

There are three types of log files created during System Acceptance Testing.

- 1) Regression Log file : This Log file is maintained and is created when some functionality that the tester has been using till yesterday has disappeared recently. The log file consists of the information under which platform, which step is

not working, who is the tester and the behaviour of the test case with screen shot attached if any.

- 2) Non-Regression Log file : This Log file is maintained and is created when some functionality that the tester has been using previously and recently has disappeared.
- 3) Showstopper Log file : This log file is created when the manual test case is created and added to test set of the software product and is failed when executed for the first time.

The log file consists of the information under which platform, which step is not working, who is the tester and the behaviour of the test case with screen shot attached if any.

These Log files are reported to the developer. A screen shot of the behaviour which is not the expected behaviour must also be attached to the log file so the specific development team will analyse the issue and resolve it as early as possible.

As mentioned in System Acceptance Testing there are three log files. These log files must be reported to the developer with proper priority. Regression log file must be given High priority compared to other log files.

The Testing Life Cycle for execution of manual test cases under System Acceptance Testing is shown in the Figure 2 and Figure 3.

Fig. 2 : Machine Setup for System Acceptance Testing

Fig. 3 : Manual Test Case Execution

The effort tracking for the system acceptance testing is done on a daily basis by making an entry in the excel sheet created for System Acceptance Testing. This sheet gives tester a plan to execute within the scheduled time for every test set under different platforms.

B. Confidence Testing

Confidence testing is another form of System Acceptance Testing wherein the bugs reported to the developer during acceptance testing are verified and need to be checked in the log file whether the status is open or fixed. If the status of the issue is open in the log file confidence testing is not carried out because the issue logged is not fixed yet. If the status of the issue is fixed confidence testing is mandatory and

those test cases which have to be executed under confidence testing are grouped.

In confidence testing team may include some tests that might have impact or are critical are verified once again and any bug if found is documented in the log file and report is sent to the developer.

C. Exploratory Testing

Exploratory testing is defined as simultaneous learning, test design and test execution; that is the tests are not defined in advance in an established test plan but are dynamically designed executed and modified. The effectiveness of exploratory testing relies on the software engineer's knowledge which can be derived from various sources [6]. James Bach defines Exploratory Testing as simultaneous learning, test design and test execution [5]. Tinkham and Kaner's definition of exploratory testing as "Any testing to the extent that the tester actively controls the design of the tests as those tests are performed and uses information gained while testing to design new and better tests" [10]. But according to Kaner, Bach and Pettichord exploring means ".....purposeful wandering: navigating through a space with a general mission but without a prescribed route. Exploration involves continuous learning and experimenting"[7]. Exploratory Testing is also useful when test scripts become "tired", that is they are not detecting many defects anymore [8]. Vaga and Amland propose that exploratory testing should be planned as part of the testing approach in most of the software development projects[11]. There are some pre-requisite documents of the software product for which a tester need to go through and understand the requirements and platform in order to undergo exploratory testing. A tester might focus on for example User interface screen and message display after making a data entry or the priority of logic for the radio buttons over the default check might be done in exploratory testing. Initially in exploratory testing the tester might have low priority to work towards the logic and identify the defects than high priority towards the user interface screen. The tester always has the benefit of identifying the defects in the user interface screen and when tester makes data entry with a Number, String, and Decimal etc. The developer has the advantage of both logic and the user interface in exploratory testing.

Effort tracking for Exploratory Testing is captured in an excel sheet.

The attributes included are as follows:

- 1) Employee ID or Serial Number
- 2) Resource
- 3) Issue reported or Missing Test case
- 4) Log files creation Identifier
- 5) Defect created (Yes/No)
- 6) Defect Number
- 7) Test case Added
- 8) Number of tests

- 9) Comments
- 10) Brain storming sessions

This sheet gives a tester a clear framework as to who is doing Exploratory Testing and under which platform a tester carried out exploration of the software product.

Exploratory Testing gives a tester an in depth knowledge of the software product and a tester will be experimenting with existing functionality in the software product. The expected behaviour must be discussed with the specific development team. So to identify the development teams for analysing the expected behaviour in exploration is an important task. The communication between the developer and tester plays an important role in Exploratory Testing. Exploratory Testing should be planned accordingly by the team leads but there must be a plan of carrying Exploratory Testing after system acceptance testing for every release of the software product. The plan for a tester is to discover at least 3 defects or discuss issues regarding the existing test case or add another scenario for the existing test case during Exploratory Testing. The tester will get resolved from the developer or any experienced tester within the team and get it verified with the developer so that the tester can add a test case for the test set for it after the log file is created and sent to the developer. The development team will have to fix the issue and give confirmation that it has been resolved in the release of the product version. The tester when the release of the product version mentioned will try to reproduce the steps as the fix has been done and a new test case is written for it by the tester. However this type of testing will give the tester to enhance further and get more knowledge and do more exploration on the software product for every release. Exploratory Testing is embedded into the System Acceptance Testing where you start the initial stage of which functionality can be explored. A Strategy needs to be carried out at this stage. This kind of activity for individual testers will have some competition among testers and the testing team will come to know about the defects and flexibility to analyse the expected behaviour of the software product.

D. Root Cause Analysis

Root cause analysis is defined as the purpose of determining the root cause is to fix the problem at its most basic source so that it does not occur again even in other software products. The Root cause is the most basic causal factor or factors that if corrected or removed will prevent the recurrence of the situation.

Root cause analysis is done after completing the system acceptance testing. Root cause analysis template is prepared and is filled by the tester. The tester does not meet the test set plan for system acceptance testing will have to be answered in Root cause analysis excel sheet. The template in general includes what, how and why.

The purpose of conducting a session on Root cause analysis is to avoid the delay and just to get prepared with the solution under such circumstances when it happens again in the future. The Root cause analysis factors included in the sheet are as follows:

- 1) The version or release of the product
- 2) Component on which the tester is working
- 3) Log file generated if any
- 4) Description of the Problem faced during system acceptance testing
- 5) Status of the log file.
- 6) Maintenance of the test case
- 7) Defect ID updating
- 8) Reasons behind the problem
- 9) Solution to the problem

E. Problem List Entry

The Problem List Entry are made by the development team of the Software product and it is sent to the tester in order to reproduce the steps mentioned in the file.

These entries are stored in a separate file with Problem List Entry ID, Platform for the execution, Status (reproducible or not), developer's name or customer's name, customer's model if any and Headline of the Problem List Entry.

Under the Headline of the Problem list entry the developer will create the detailed description of the problem which is to be analysed and communicated with specific Development team about the problem's perception. The tester initially needs to describe in brief about the reproducible steps when the problem statement is analysed.

The steps to be reproduced will help the developer whether the tester has understood the requirements, objectives of the problem. The tester will have reproducible steps and will make it to write a manual test case for it and then the test case is sent for review to developer or to an experienced tester within the testing team who has worked under that platform for the software product. When the manual steps are approved by the developer the tester will include it for the System Acceptance Testing or confidence testing or further can be explored in exploratory testing. If the problem list entry made is matching with any one of existing manual test case or is a continuation of the test scenario then another test case is added or if it is completely same then it is rejected. The test case when approved is added to the test set of the software product with test data, test objectives, requirements, Assumptions, test cases, platform, history of changes, step number, step name, expected behaviour and a screen shot of the expected behaviour if any. Whenever a test case is written for the problem list entry a new defect is created with the defect number and when reviewed the defect is closed. The effort is also tracked for the problem list analysed and test case written for it. This effort helps in future estimations and collating accurate metrics. It

also helps to analyse the strengths of different resources.

III. RESULT ANALYSIS

The effort information for the system acceptance testing in executing the manual test cases is calculated for analysing the tester whether he is capable to meet the plan with the specification of working hours for each test set. The total number of tests in each test set assigned to tester with complexity level (low, medium and high) and initially planned working hours for the execution of test cases in every test set under different platforms must be matched when you make an entry in effort information excel sheet. The entry made in the effort information for the planned hours in execution of test cases in every test set might have increased due to some reasons is discussed in the Root Cause Analysis(RCA). Effort is calculated by the formula as follows:

$$\text{Total Effort} = \text{Number of days worked in a week} * 7. \quad (1)$$

Assumption: In a day 7 hours for the execution of test cases.

$$\text{Variance} = (\text{Actual Planned Effort} - \text{Total Planned Effort}) * 100 / \text{Actual Planned Effort} \quad (2)$$

The Effort information and root cause analysis for system acceptance testing will be documented for future references.

IV. CONCLUSIONS AND FUTURE WORK

Software testing an evergreen domain in Software engineering addresses the presence of errors and provides satisfaction to the client after the release of the software product and is ready for operational use for the clients as per requirements.

However there are many upgrades in the testing techniques and the plan of choice for conducting a testing must be well-defined and documented. Software testing in industrial software development is carried out through a test plan wherein every tester will go through it and any concerns will be addressed through meetings so that when System acceptance testing starts the tester is good enough to understand the platform, requirements and effort to be met for execution of manual test cases. Tester should also know about different Development teams for that software product being tested so that when the behaviour of the functionality is not as expected it must be addressed to the specific development team and get it resolved. So Communication between tester and developer plays a vital role in the software testing process. Further some selected manual test cases can be automated in the future inorder to reduce the time. The measurement of effort tracked for testing is done wherein the initial plan of testing activity when done is matched with the effort. Effort plan for the

execution of test cases for the product plays a vital role in the release of the software product.

The process overview of the testing activity done will help in building high quality software plays an important procedure for Researchers and enhancement can be done so that tester will do the testing activity more tester friendly in the future. Researchers should demonstrate the process overview of the Testing activity, plan, measurement for the software product and the execution of test cases given for tester under different platforms should be planned in such a way that tester is comfortable in all the platforms with initial understanding of the requirements and execution. Tester should not always be testing on specific platform. However the goal and plan to the tester for the execution of test cases should have been undergoing testing activity on all platforms for specific time periods and well-documentation of the procedures and Configuration should have been done. There should be proper knowledge transfer done initially for the execution of test cases with proper configuration from the experienced tester whenever tester is assigned for the execution under different environments. Whenever the tester starts execution of the test cases under different platforms after proper knowledge transfer being done will find challenging in executing test cases. Problem list entry for a developer and Tester acts as a exploratory testing activity. Researchers should focus on the foundation being given for testers and making it more challenging with the well-documentation activity being carried for process overview of the testing activity and any updating should be done for future references.

V. ACKNOWLEDGMENT

I am happy to present this paper to my Lecturer, Naresh E (Software Engineering, Department of Information Science and Engineering). I am grateful that Naresh E gave us such an interesting idea to do, and I have tried my best to succeed at it. I would also like to thank my family members for their support. A special thanks to the authors mentioned in the references page.

REFERENCES

- [1] Barnard, J. H. Collicott R. B. COMPAS: a development process Support system, AT&T Technical Journal, Volume 69, No.2, pp 52-64, MarcW April 1990.
- [2] Coad and Yourdon. Object Oriented Design, Prentice Hall, Englewood Cliffs, NJ, 1991.
- [3] Lorenz, Mark : Object Oriented Software Development – A Practical guide, Prentice Hall, Englewood Cliffs, NJ, 1993.
- [4] Royer, Thomas C., Software Testing Management - Life on The critical path, Prentice Hall, Englewood Cliffs, NJ, 1993.

-
[5] J. Bach, "Exploratory Testing", in the Testing Practitioner, Second ed., E. van Veenendaal Ed. Den Bosch: UTN Publishers, 2004, pp. 253-265. [6] IEEE, "Guide to the Software Engineering Body of Knowledge", IEEE, Tech. Rep. IEEE - 2004 Version, 2004. [7] C. Kaner, J. Bach and B. Pettichord, Lessons Learned in Software Testing, New York: John Wiley & Sons, Inc., 2002. [8] L. Copeland, A Practitioner's Guide to Software Test Design, Boston: Artech House Publishers, 2004. [9] Hiren D. Desai, Research Engineer, Science and Technology, BellSouth Telecommunications Inc, Atlanta, GA, Test Case Management System, IEEE, 1994.	[10] A. Tinkham and C. Kaner, "Exploring Exploratory Testing", 2003, Accessed 2005 04/25, http://kaner.com/pdfs/Exploring ExploratoryTesting.pdf. [11] J. Vaga and S. Amland, "Managing High-Speed Web Testing", in Software Quality and Software Testing in Internet Times, D. Meyerhoff, B. Laibarra, van der Pouw Kraan, Rob and A. Wallet Eds., Berlin: Springer-Verlag, 2002, pp. 23-30. [12] E. F. Miller, "Introduction to Software Testing Technology," Tutorial: Software Testing & Validation Techniques, Second Edition, IEEE Catalog No. EHO 180-0, pp. 4-16. [13] B. Beizer, "Software Testing Techniques," Second Edition, Van Nostrand Reinhold Company Limited, 1990, ISBN 0-442-20672-0.