RESEARCH ARTICLE OPEN ACCESS

Affect of Quality Management in Finding Test Escapes by Improving Process at Different Levels in SDLC

Kranthi Kumar R [1], Kiran Kumar Reddi [2]
Research Scholar [1], Rayalaseema University, Kurnool, AP Registration No: PP COMP.SCI&ENG- 0457
Département of Computer Science [2], Krishna University
Machilipatnam - India

ABSTRACT
The complexity and size of software systems are continuously increasing. As part of the increasing growth rate and more complex and larger systems are coming into existence, new challenges and problems are being faced by the industry. Here the redefining of the SDLC is required, about changing the priorities of the activities that go into Software development. Software Testing is one such area to which little importance was attached in the early days of software development, as the emphasis was on coding and design Organizations are spending a lot of time on Software testing to make their projects successful but still the rate of software projects failure is very high.

Keywords:- Software requirement management; Requirement Analyst; Regression Testing; Retesting; Functional Testing; Test Plan; Test Strategy; Levels of Testing; Types of Testing

I. INTRODUCTION

Software industry seems to be a sector that has witnessed the highest rate of project failure in the world. Testing provides a complete software application testing solution for medium to large corporations, including Fortune 500 companies. The service is ideal for companies needing independent and/or additional testing on the software applications they develop for in-house and external clients. Testing personnel are highly trained with access to the latest technology, to ensure through testing within most industry environments.

A. What is testing?

Testing is one of the most important phases of Software Development Life Cycle (SDLC). We cannot launch any product successfully without testing. Even the product designed by best of the best developers needs to be tested and verified under various circumstances. Software testing is a process used to help identify correctness, completeness and quality of developed computer software. What is the expected behavior of the system? Testing is to check if there is no difference between actual output and expected output. Testing is an exercise to simulate a system or program operation. It helps the program to achieve the required goal. Testing is to establish a confidence that a system does what it is supposed to do and even if it "misbehaves" it should not crash, it should do it decently. In other words, testing is not a debugging or preventing process, it is only to ensure that within an application I have found/detect these many number of bugs and rest debugging/fixing of bugs is done by developers. Software Development Life Cycle (SDLC) and Software Testing Life Cycle (STLC) they are executing concurrently.

Short cut to remember the starting letter **FURRPSC MODEL:**

F → Functionality Testing
U→Usability Testing
R→ Reliability Testing
R→ Regression Testing
P→ Performance Testing
S→Scalability Testing
C→Compatibility Testing

A. Functionality Testing

To conform that all the requirements are covered. Functional requirements specify which o/p should be produced from the given input. They describe the relationship between Input and Output of the system. A major part in black box testing is called functional testing.

1. Input domain—whether taking right values of i/p or not.
2. Error handling----whether the application reporting.
3. URL's checking—for only web application, all the links are correcting working or not.

Testing Approach

1. Equivalence class.
2. Boundary value analysis.
3. Error guessing.

B. Usability Testing

To test the ease (comfort, facility) and user-friendliness of the system.

Approach: Qualitative and quantitative Heuristic Check List.

Classifications of checking:

1. Accessibility
2. Clarity of communication
3. Consistency
4. Navigation
5. Design & maintenance
6. Visual representation

Approach

1. **R**epeatedly working on the same functionality. (R)

2. **C**ritical **Q**uery **E**xecution. (CQE)
3. To emulate peak load.

Qualitative approach

1. Each and every function should be available from all the pages of the site.
2. User should be able to submit request within 4-5 actions.
3. Confirmation message should be displayed for each submits.

Quantative approach:

The average of 10 different people should be considered as the result.

C. *Reliability Testing*

Which defines how well the software meets its requirements? Objective is to find mean time between failure/time available under specific load pattern and mean for recovery.

Approach RRT (Ration Real time tool)

D. *Regression Testing*

To check the new functionalities have been incorporated correctly without failing the existing functionalities.

*Approach**:*** Automation Tool.

The bugs need to be communicated and assigned to developers that can fix it. After the problem is resolved, fixes should be re-tested, and determination mode regarding requirements for regression testing to check that fixes did not create problems elsewhere

E. *Performance Testing*

Primary objective of the performance testing is "to demonstrate the system functions to specifications with acceptable response times while processing the required transaction volume on a production sized data base.

Objectives:

1. Assessing the system capacity for growth.
2. Identifying weak points in the architecture.
3. Detect obscure bugs in the software**.**

Performance parameters;

1. Request – response time.
2. Transactions per second.
3. Turnaround time
4. Page down load time
5. Through put

Approach: Classification of performance testing.

1. Load test
2. Volume test
3. Stress test

F. *Stress testing:*

Finding break point of application. Max. No. of users that an application can handle (at the same time).

G. *Volume testing:*

Execution of our application under huge amounts of resources is called volume testing. To find out threshold point we may use this test.

Approach: Data Profile.

H. *Load testing:*

With the load that customer wants (not at the same time). Load is increasing continuously till the customer is required load. Gradually increasing the load on the application and checking the performance.

Approach: Load profile.

I. *Scalability testing:*

To find the maximum number of user system can handle. (Customer will give max. no.)

Approach: performance tools.

Classification:

1. Network scalability
2. Server scalability
3. Application scalability

J. *Compatibility testing:*

How a product will perform over a wide range of hardware, software & network configuration and to isolate the specific problems.

Approach: ET Approach.

Environment Selection:

1. Understanding the end user's application environment.
2. Importance of selecting both old browser & new browser.
3. Selection of the operating system.

II. SOFTWARE DEVELOPMENT LIFE CYCLE Vs SOFTWARE TEST LIFE CYCLE.

Software development life cycle (SDLC) and Software Testing Life cycle (STLC) they are executing concurrently.

SDLC	STLC
SDLC is Software Development Lifecycle; it is a systematic approach to develop software.	The process of testing software in a well planned and systematic way is known as software testing life cycle (STLC).
Requirements gathering	Requirements Analysis is done is this phase, software requirements are reviewed by test team.
Design	Test Planning, Test analysis and Test design is done in this phase. Test team reviews design documents and prepares the test plan
Coding or development	Test construction and verification is done in this phase, testers write test cases and finalizes test plan.
Testing	Test Execution and bug reporting, manual testing, automation testing is done, defects found are reported. Re-testing and regression testing is also done in this phase.
Deployment	Final testing and implementation is done is this phase and final test report is prepared.
Maintenance	Maintenance testing is done in this phase.

Table -1 SDLC Vs STLC

III. WHY TO Do TESTING?

As we discussed earlier, even though we have best developers developing the product still there may be flaws somewhere due to misconception, due to lack of understanding in integrating modules of too great developers and so on. Testing, if done poorly, defects are found during operation, it results in high maintenance cost and user dissatisfaction. So here are the most common software problems leading to software testing a vital role in SDLC. Test activities exist before and after test execution. These activities include planning and control, choosing test conditions, designing and executing test cases, checking results, evaluating exit criteria, reporting on the testing process and system under test, and finalizing or completing closure activities after a test phase has been completed. Testing also includes reviewing documents and conducting static analysis. Both dynamic testing and static testing can be used as a means for achieving similar objectives, and will provide information that can be used to improve both the system being tested and the development and testing processes.

IV. OBJECTIVES OF TESTING

1. Finding defects.
2. Gaining confidence about the level of quality.
3. Providing information for decision-making.
4. Preventing defects.

V. SEVEN TESTING PRINCIPLES. [5]

1. Testing shows presence of defects.
2. Exhaustive testing is impossible.
3. Early testing.
4. Defect Clustering.
5. Pesticide paradox
6. Testing is context dependent
7. Absence-of-errors fallacy.

V. WHAT IS TESTING LIFE CYCLE

Software testing life cycle identifies what test activities to carry out and when (what is the best time) to accomplish those test activities. Even though testing differs between organizations, there is a testing life cycle. It consists of six phases

1. Test Planning,
2. Test Analysis,
3. Test Design,
4. Construction and verification,
5. Testing Cycles,
6. Final Testing and Implementation and
7. Post Implementation.

Software testing has its own life cycle that intersects with every stage of the SDLC. The basic requirements in software testing life cycle is to control/deal with software testing – Manual, Automated and Performance.

Phase	Activities.	Outcome
Planning	Create high level test plan	Test plan, Refined Specification
Analysis	Create detailed test plan, Functional Validation matrix, test cases	Revised Test Plan, Functional Validation matrix, test cases.
Design	Test cases are revised; select which test cases to automate	Revised test cases, test data sets, sets, risk assessment sheet.
Construction	Scripting of test cases to automate	Test procedures/scripts, Drivers, test results, Bug reports.
Testing Cycles	Complete testing cycles	Test Results, Bug Reports
Final Testing	Execute remaining stress and performance tests, complete documentation	Test results and different metrics on test efforts
Post Implementation	Evaluate testing processes	Plan for improvement of testing process

Table - 2: Phases and activities and outcomes in Testing

STLC is a comprehensive group of testing related steps followed to deliver quality product. Testing has become an important phenomenon during and after development of any software development project. The various steps in testing process are:

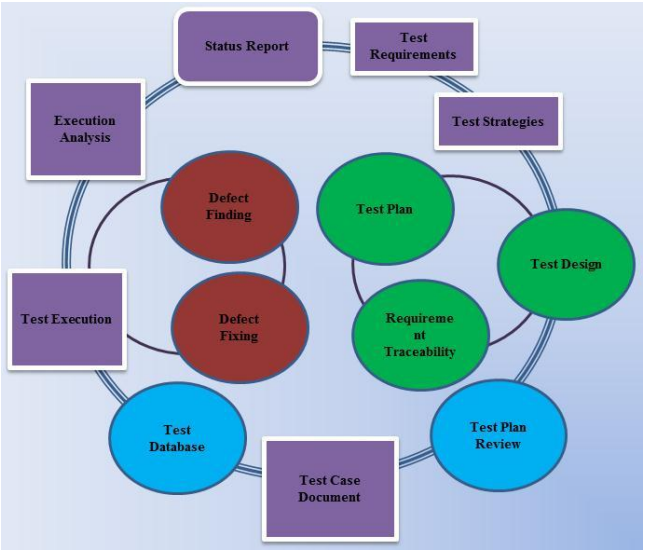

Figure 1: Software Testing life Cycle

This is the initial and key step in software testing. This step determines the overall flow of testing process starting from:

1. Requirement gathering, its analysis and specifications
2. Framing the test plan and test scenario
3. Preparing test cases and its approval
4. Tools selection and confirmation

We carry out our test planning on the basis of IEEE Standard for Software Test Documentation and other industry specifications.

A. Testing environment and harness preparation

This stage is defined by test automation framework design and development and test scripts creation. SCMS a software development company has state-of-the-art testing equipment, system software, and web middleware. This ensures creation of proper testing environment. It also helps in essentially decreasing the time for testing configurations deployment.

B. Test Execution

The manual and automated software testing, as specified in the test plan, is applied in a dynamic state of code. The approaches and methods are utilized to validate the executable code developed meeting the specifications stated at start of software Project. For each of these types of testing we have established approaches, proven testing tools, and adjusted reporting documents templates. Testing techniques applied:

1. Black Box / White Box testing
2. Ad Hoc / Exploratory testing
3. Scripted testing

C. ***Test Plan:*** Below is a formal detailed table that describes

COMPONENTS	DESCRIPTION	PURPOSE
Scope	Testing scope : what to be Tested and What NOT to be Tested	Outlines the entire process and maps specific tests
Objective	What are the main Objectives to do for Testing	To find as many bugs in the earlier stage
Risk Analysis	Critical items that will be tested	Provides focus by identifying areas that are critical for success
Test Strategy	Basic approach of Test Strategy may describe the Test Levels to be carried out.	Which describes the Organization's method of testing involved including Project and Product Risks management, the division of testing into Steps, Levels or Phases, and the high-level activities associated with the testing
Roles and Responsibility	Specific people who are and their assignments	Assigns responsibilities and keeps everyone on track and focused
Environmental Tools	The technical environment, data, work area, and interfaces used in testing	Reduces or eliminates misunderstandings and sources of potential delay

Assumptions	Code and systems status and availability	Avoids misunderstandings about schedules
Communication	Communications plan—who, what, when, how	Everyone knows what they need to know when they need to know it
Defect Reporting	How defects will be logged and documented	Tells how to document a defect, it can have reproduced, fixed, and retested

Table - 3: Test plan details

D. Test Strategy Flow:

Test Cases and Test Procedures should manifest Test Strategy.

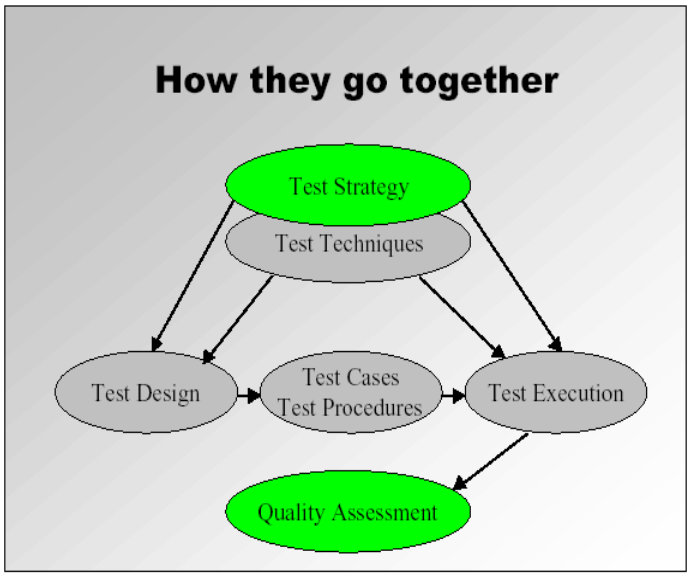

Figure 2: Test flow

E. Need for Test Strategy

The objective of testing is to reduce the risks inherent in computer systems. The strategy must address the risks and present a process that can reduce those risks. The system concerns on risks then establish the objectives for the test process. The two components of the testing strategy are the Test Factors and the Test Phase.

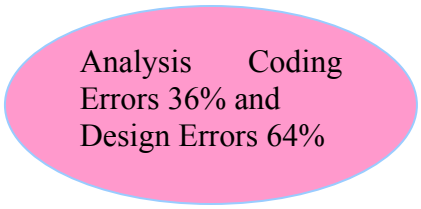

Figure 3: Test strategy

1. Test Factor – The risk or issue that needs to be addressed as part of the test strategy. The strategy will select those factors that need to be addressed in the testing of a specific application system.
2. Test Phase – The Phase of the systems development life cycle in which testing will occur

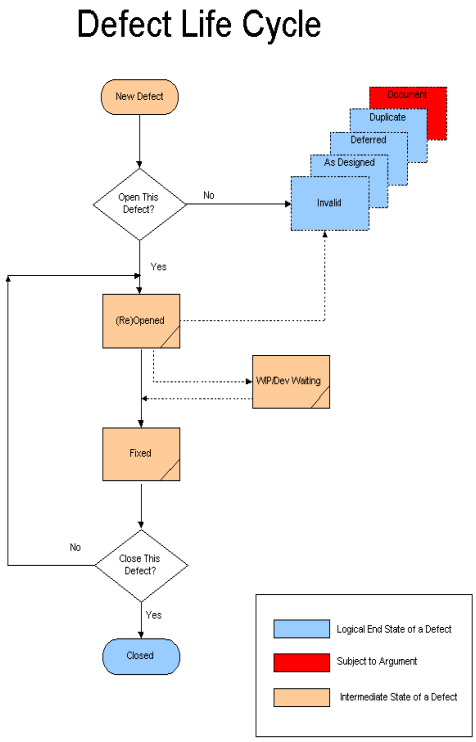

Figure 4: Defect Life Cycle

1. The Project Lead of the development team will review the defect and set it to one of the following statuses:
   a. Open – Accepts the bug and assigns it to a developer.
   b. Invalid Bug – The reported bug is not valid one as per the requirements/design
2. As Designed – This is an intended functionality as per the requirements/design
3. Deferred –This will be an enhancement.
4. Duplicate – The bug has already been reported.
5. Document – Once it is set to any of the above statuses apart from Open, and the testing team does not agree with the development team it is set to document status.
6. Once the development team has started working on the defect the status is set to WIP ((Work in Progress) or if the development team is waiting for a go ahead or some technical feedback, they will set to Dev Waiting

7. After the development team has fixed the defect, the status is set to FIXED, which means the defect is ready to re-test.
8. On re-testing the defect, and the defect still exists, the status is set to REOPENED, which will follow the same cycle as an open defect.
9. If the fixed defect satisfies the requirements/passes the test case, it is set to Closed.

Severity	Description
Critical	The problem prevents further processing and testing. The Development Team must be informed immediately and they need to take corrective action immediately.
High	The problem affects selected processing to a significant degree, making it inoperable, Cause data loss, or could cause a user to make an incorrect decision or entry. The Development Team must be informed that day, and they need to take corrective action within 0 – 24 hours.
Medium	The problem affects selected processing, but has a work-around that allows continued processing and testing. No data loss is suffered. These may be cosmetic problems that hamper usability or divulge client-specific information. The Development Team must be informed within 24 hours, and they need to take corrective action within 24 - 48 hours.
Low	The problem is cosmetic, and/or does not affect further processing and testing. The Development Team must be informed within 48 hours, and they need to take corrective action within 48 - 96 hours.

Table - 4: Defects Classification by Severity

V Model Vs Waterfall Model		
FEATURES	WATERFALL MODEL	V MODEL
REQUIREMENT SPECIFICATION	BEGINNING	BEGINNING
COST	LOW	EXPENSIVE
GUARANTEE OF SUCCESS	LOW	HIGH
SIMPLICITY	SIMPLE	INTERMEDIATE
FLEXIBILITY	RIGID	LITTLE FLEXIBLE
REUSABILITY	LIMITED	TO SOME EXTENT
USER INVOLVEMENT	ONLY AT BEGINNING	ONLY AT BEGINNING
CHANGE INCORPORATED	DIFFICULT	DIFFICULT

Table - 5: Difference between Waterfall & V Model - SDLC

VI. CONCLUSION

V- Model SDLC is almost the same as the **Waterfall model,** as both the models are of sequential type. Requirements must be very clear before the project starts, because it is usually expensive to go back and make changes.

The advantage of the V-Model - SDLC is very easy to understand and apply. The simplicity of this model also makes it easier to manage. Studies indicate that between 40% and 60% of all defects found in software projects can be traced back to errors made while gathering requirements.

The objectives of the current research are to focus on those defects (Test Escapes) to prevent those defects in the early stage of SDLC's by improving process at different levels in SDLC

The Objective of current Research is to improve and enhance the process to eliminate drawbacks of following key issues:

- ✓ High Risk and Uncertainty
- ✓ Not a good model for complex and object-oriented projects
- ✓ Not suitable for long and Ongoing projects
- ✓ Not suitable for the projects where Requirements are at a moderate to high risk of changing
- ✓ Once an application is in the testing stage, it is difficult to go back and change a functionality
- ✓ No working software is produced until late during the life cycle

REFERENCES

[1]. http://www.softwaretestingmentor.com/stlc/stlc-vs-sdlc.php
[2]. http://en.wikipedia.org/wiki/Software_testing
[3]. http://www.softwaretestingsoftware.com/defect-life-cycle/
[4]. http://www.google.co.in/search?hl=en&q=STLC+LIFE+CYCLE&bav=on.2,or.r_gc.r_pw.,cf.osb&biw=1360&bih=679&um=1&ie=UTF-8&tbm=isch&source=og&sa=N&tab=wi&ei=_zHwTtLSFoKyrAeXvC0
[5]. http://www.flamelab.de/article/7-principles-of-software-testing/

[6]. Software Project Management in Practice by Pankaj Jalote- 2010.
[7]. Software Project Management, A Unified Frame work by WALKER ROYCE -2009.
[8]. http://en.wikipedia.org/wiki/Requirements_analysis
[9]. https://www.tutorialspoint.com/sdlc/sdlc_v_model.htm