Seven Fundamental Principles of Testing

Vaishali Borse

K.K.W.I.E.E.R, Nasik

Abstract: *Software testing is very essential in today's scenario, for this the seven fundamental principles are required which has been observed since last 40 years. This paper describes the principles and examples of for each principle along with their requirement. It also describes how to spot these principles and how to take advantage of them. This will make a better tester. In this paper there is explanation for why testing is required, what are the different steps for testing, what is V- model?*

Keywords: Defects, Clustering, Pesticide, Paradox, SDLC, STLC

1. Introduction

In today's life software is very important for all of us. In any organization have more contribution to developed software, android apps. For developing any software, first of all organization needs client requirement that is called requirement gathering. Based on the client's requirements, developer develops the software. Such development also goes through SDLC cycle. i.e. Software Development Life Cycle. It is having six steps –
1) Requirement gathering and analysis
2) Design
3) Implementation or Coding
4) Testing
5) Deployment
6) Maintenance

The problem arises is why does software fail after going through above steps. Because humans are important beings so, software is important because humans are important. The chain is like this – A Programmer makes a mistake or error, for getting giving long string into a field on a screen. The programmer puts a defect into the program, such as for getting to check input strings for length when the program is executed, if the right conditions exist. The defect will result in unexpected behavior,

The system exhibits a failure, failure being the result of 'bugs in the code'. The percentages of no. of defects are in requirement specification, design and implementation from research report are as follows:
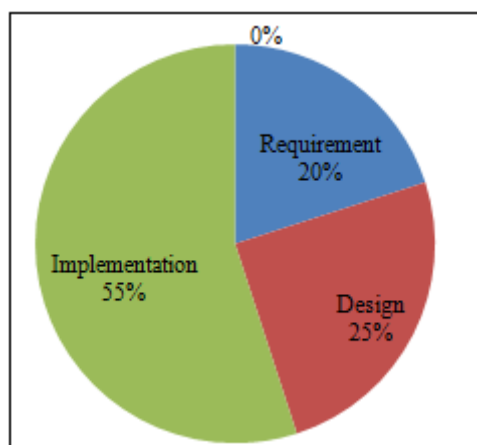

Figure 1.1: Percentages of Defects

Generally all work products will always have defects or bugs, so there are defects and there are risks of system failure. If defects are increased then risks of failure factor is also increased. For this testing is required .testing is part of how their risks of failure factor can be reduced.

Software testing is an activity to check actual output matches with expected output, and to ensure that the software is defect free. Same as SDLC, software testing has a life cycle that is STLC i.e. Software Testing Life Cycle. It has five steps which are as follows:
1) Planning and control
2) Analysis and Design
3) Implementation and Execution
4) Evaluating exit criteria and Reporting
5) Test closure activities

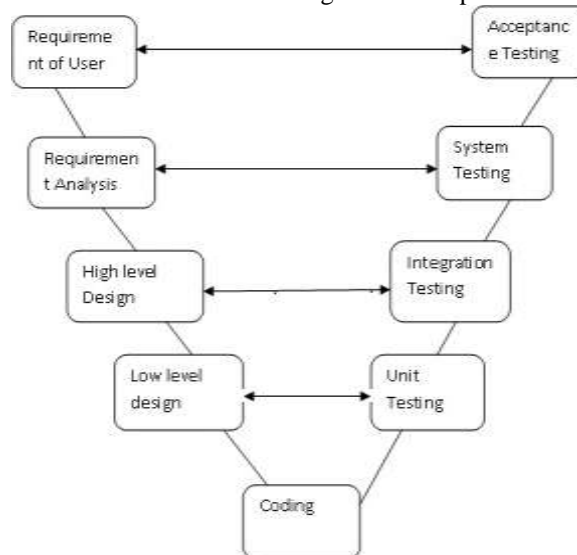For every phase in SDLC is a corresponding with testing phase then the V-model of testing was developed.

Figure 1.2: V-model

Here in Fig 1.1the left side is SDLC and the right side is STLC. In which it shows how each and every phase is co-related with each other. It clearly shows that each step of SDLC has testing phase, so testing is very essential part of system and for this seven principles of testing are important. These seven principles are the pillars of testing.

2. Understanding

Basically, the Principles are nothing but the rules or laws has to be followed. Principles are essential characteristics to developing and testing any system or software. These are important to obtained test result positive or we can say well. But for this we required right strategy for testing, so these right strategies are obtained from seven principles of testing.
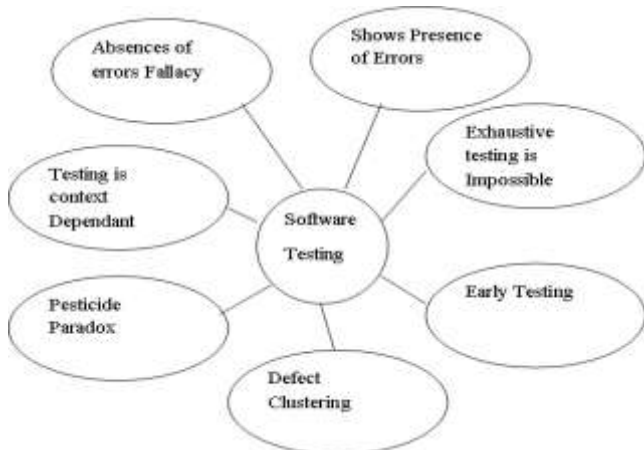
Figure 2.1: Seven Principles

2.1. Testing shows the presence of defects

In the first principle after doing testing activities, testing shows the defect are present in software .Defects are identified by different software testing execution techniques. But at the same time, testing doesn't prove that there are no defects present in the software. Testing shows the presence of defects, it will be 99.5% may be, but should not give a proof for defect free software. Testers are unable to find defects 100% after repeated regression testing, doesn't mean that the software is bug –free.

2.2. Exhaustive Testing is impossible

Exhaustive testing is not practically possible that is it is not possible to test complete software. Test with all inputs combinations is not possible i.e. with all scenario is impossible. Question arrives that how we will test the complete software; it is performed by risk-based testing. In which identifying the impact, this can help us to identify the module which are on high risk.

Another way to test impact combination is to test requirements based testing. Here testers identify the user's requirement specification for identifying test conditions. Test efforts are based on priority to number of test cases for each requirement.

2.3. Early Testing

Testing activities should start early in software or system development life cycle, because early test requirement or early test design and reviews activities find defects early when they are cheap to find and fix.

Defect (requirement+Design)=1/2 Defects(Total)
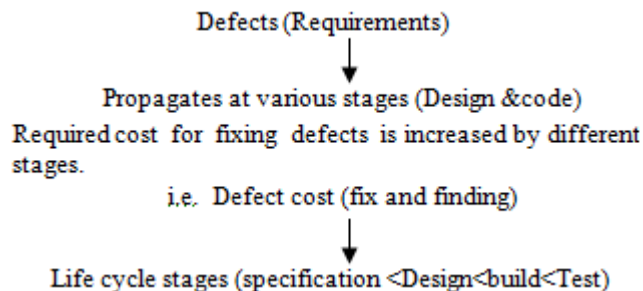
So reduced cost of defects fixing and finding, it means reduced duration of the project because we should minimize the cost of remaining defects and also required time.
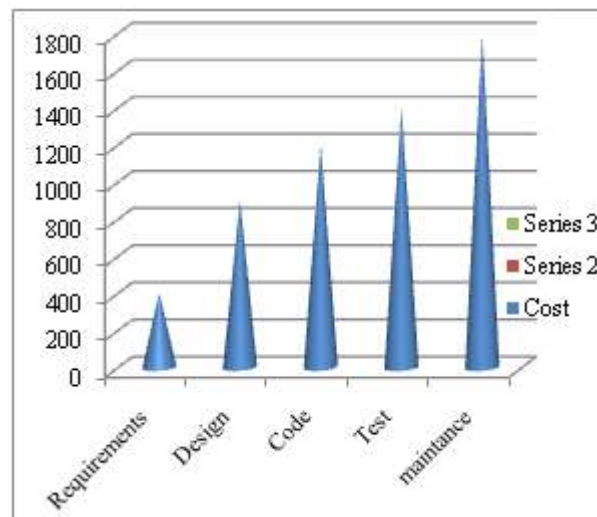
Figure: SDLC stages with cost

2.4. Defect Clustering

Most of defects discovered in small modules during pre-release testing. Some areas in the projects which can be tricky and complex, thus these are the hot spots of the application for the testing. Such areas are called clusters or groups .It is an important task to identify the potential clusters.

2.5. Pesticide paradox

If some test cases are repeated again and again for checking, they show some defects again and again, there are no any new bugs. So for this 'Pesticide paradox' is required. In which the test cases regularly need to be reviewed and revised on regular interval in order to find new defects and new test cases need to be written for exercise.

2.6. Testing is Context Dependent

Firstly context of testing means how much testing we do and how the testing is done. Context is nothing but it is framework for testing. If you doing testing for web application and mobile application using some strategies of testing then it is absolutely wrong because by this principle the testing strategies are different for different applications, it is totally dependent on type of applications. Generally it depends on the level of risk and impact associated with the work product. It also depends on some factors like

deadlines, time pressure, resources availability and market needs.

2.7. Absence of Error Fallacy

This principle states that if system tasted 100% but we can't say it is bug free. Every system has at least 0.01% errors. So we can't say system fulfilled user's all requirements. Due to this some systems are failed in user acceptance testing because of high level of defects or low level of defects.

3. Conclusion

Based on study and analysis of seven principles of testing, there were some conclusions drawn.

Testing software is to find out the defects, not to prove that the system or software is totally bug free. Testing starts from requirements gathering phase i.e. early testing reduces the amount of budget and defect cost.

Testing an application exhaustively is impossible i.e. testing all kind combinations of inputs is not feasible. Testing is context based- software testing is always based on the purpose to which software build will be used.

Performing similar kind of testing again and again does not identify the defects. Absence of errors in an application does not mean that, the application is free from defects.

References

[1] Rex Black, Erik Van veenendaal Dorothy Graham- Foundations of Software Testing ISTQB Certification.
[2] www.guru99.com/software-testing.html.

Author Profile

Vaishali Borse is an Assistant Professor at K.K.Wagh Institute of Engineering Education and Research , Nasik, India .Affiliated to University of Pune. She is Post Graduate in MCA and also ISTQB Certified Tester.