Fast Track Technique for Software Testing and Quality Assurance Practice in Project Development Life Cycle

¹Chodagam Suresh Kumar, ²Mikkili Dileep Kumar, ³Donavalli Venkata Vidya Deepthi, ⁴Prathipati Ratna Kumar

¹Assistant Professor in Computer Science and Engineering Department, Anurag Engineering College, Kodad, Khammam, ²Assistant Professor in Computer Science and Engineering Department Khammam Institute of Technology and Science, Poneekal, Khammam, ³Assistant Professor in Computer Science and Engineering Department, SirCRRCoE, Eluru⁴, Associate Professor in Computer Science and Engineering Department, SirCRRCoE, Eluru
¹churesh_kmr@yahoo.co.in, ²dilipmtech@gmail.com, ³deepthidonavalli@yahoo.com, ⁴rk30111972@yahoo.com

Abstract- Technology is growing rapidly in today's world. The system that produces technology is also changing. According to surveys it tell that unemployment is high and most of the company's are slashing there expenditure to meet there requirements. Companies are not maintaining good employees. When no quality employees then no quality product. These terms are interrelated to each other. In this period no company can offer good quality product. Human inventions are changing based on towards world requirements but still software industry is facing problems for its survival. Experienced computer professionals and upcoming computer professional are victims of present precision. Software industry is now a patient on ventilator. Most of software companies are winding up because of precision and some exist with limited employees. Our paper gives explanation about how a software industry can maintain quality within short span. The cost of software problems or errors is a significant problem to global industry, not only to the producers of the software but also to their customers and end users of the software. There is a cost associated with the quality of software to companies who purchase a software product and also to the companies who produce the same piece of software. The task of improving quality on a limited cost and time base is a difficult one.

The foundation of this paper lies with the difficult task of evaluating software from its inception through its development until its testing and subsequent release. Even we can make software project success with few employees in the company. The focus of this paper is on the improvement of the testing & quality assurance task in a Software company with software quality problems but with a limited budget and reduced cost of time.

Testing practices and quality assurance methods are outlined in this paper explaining what was used during the software quality improvement process in the company.

Index Terms - Software Problems, Software Quality, Technology, Testing, Ventilator.

I. INTRODUCTION

A.. Present situation of Software Employees

Japan like country is booming in humanoid and robotics. But India, USA and European countries which depend on software industry are facing problems. They name it as precision period. In this period they reduced staff/employees because lack of software projects. As days are going on expenditure of companies increasing and no one shows interest towards software projects. This is main cut-throat to the software industry. So Employees are maintaining consistency in same company. Some software industries are sending employees on long live and some of them even terminating existing employees. The main reasons for software crisis are lack of new software projects. Because of this unemployment in technical field like computer science and engineering and information technology are suffering. Lot of well worth computer and information technology professionals are facing problems. Hiring trends in India is shown in Fig. 1

Fig. 1 Hiring trends across top cities

B. Present situation of Software Companies

They cannot disappear from global and they pretend that they exist. But there are lots of problems surrounding them as tsunami. One major problem is the financial problem, to face such problems they have to reduce or cut-off expenditure. Reduce employee size, perks, incentives, bonus soon. If we compare recruitment from last 3 to 4 years this year it is reduced

to more. Even some software employees are committing suicide along with their families. Based on some or other reasons, the software industry is facing precision and unable to recruit good software professionals. When no worthy professionals, then they cannot obtain quality. Now non-IT industries revenue is good and more when comparing with IT industry as shown in Fig. 2 and 3. Now the graph related to software industry is touching base of it. It is our responsibility to rise it up.

Fig. 2 Hiring trends across functional areas.

Fig. 3 Hiring trends across top industry sectors.

C. Customer should benefit

According to above said information whatever the problems may surmount Software Industry but customer should be benefited with good quality of software.

D. Competitive Landscape of the industry

Present Software Industries are facing competitiveness in Fig. 4[1]. They want to exhibit that their share or revenue on projects is more. Even some Software Companies are showing fake or false revenue income. There are few things in considering competitiveness in the Software industry. They are given as follows.

1. Financial Problems
2. Employee Treats
3. Treat of new Entrants
4. Rivalry among Employees
5. Bargaining Power of Buyers
6. Bargaining power of Suppliers
7. Treat of Substitutes

Fig. 4 Competitive Landscape of the Industry.

II. PROBLEMS RELATED SOFTWARE PROCESS

A. Software related problems

Inception of software development life cycle starts at requirements elicitation and analysis, later designing and implementation. This should be done in Fast Track. Model for Fast Track is given Fig. 5. Software related problems should identify and solved as early as possible. In Fast Track model capture information as early as possible by using various techniques. End user should stay with Software Professionals work area for more than 3 days if possible or professionals should travel to client area or video conference. Based on End User requirements make a template. This template can change at any stage. The changes should be like that they should be minor or simple. We cannot measure requirement. The requirement should not be major. Software Professional work should be like that it is easy. We can recruit new employees for effective collection of requirements. We can use existing employees also. After that we carry our work to Design and Implementation phase. All three phase Requirements Elicitation [3], Analysis and Design or Architecture phase should be completed as early as possible. Software Professionals can use template Requirements Elicitation à Analysis à Design triangle where Software Professional identifies requirements, analysis and Prototype of designs at simultaneously around speculated time. The three phases should be tested whether they meet the goals which we planned. We make rotations till we satisfy with the system.

After completion of three stages we enter into implementation, testing stages. Quality of the software is based on test results. Whether we maintain Software Quality or not depends on test. If testing is done up to User, Client and Stakeholders requirements design, then possibility of Quality.

Fig. 5 Fast Track Model

B. A software company with software quality problems

This paper is focused on the creation and provision of a testing & quality assurance (QA) process for software quality improvement in any software company and also for the creation of a framework for similar quality improvements in the process. Next phase after implementation is test, very critical and important and is lead by a team. To ensure that the software released to the customers is of the highest standard. To raise the bar on this standard we decided to conduct research into testing and QA practices and to implement improved practices within the company. This process will combine elements of testing and QA into one process, this one process in turn will be inserted into the any company's development lifecycle.

The aim of this paper is to investigate the best test and QA practices in industry and to design and evaluate a process for implementing best practices in the software lifecycle of a small to medium enterprise (SME) over successive projects.

There are lots of methodologies for testing and software industry may follow some of them. Already we know some of the testing principles. Next objective is to evaluate what constitutes software quality and what factors affect this quality and how, when and where QA can be used in the project life-cycle for improving product quality. And last objective is to outline the test and QA effort during a project in a particular company and to evaluate the adoption of improved practices during subsequent projects in the same company and study a framework.

III. ANALYSIS AND SOLUTION TO THE PROBLEM

A. Principles for Software Testing

The purpose of software testing [2] is to detect errors in the software. The tester should ideally detect all errors before the software is released to the customer. Full test coverage of a program is impossible. "Proving that a program is fault free is equivalent to the famous halting problem of computer science. The main principle of software testing "is the process of executing a program with the intent of finding errors". To test the program more thoroughly a tester would

need to evaluate the program to detect both types of errors. This principle is thus more detailed to "Test the program to see if it does what it is supposed to and to see if it does what it is not supposed to do". In order for the tester to find these errors, we will devise a number of tests to execute on the software itself. These tests must be based on prior knowledge of the software. The two main thrusts of testing are firstly based on the composition of the software, i.e. its internal structure. Secondly based on the business or intended purpose of the software, i.e. the functional aspect of the software. Based on one of these box test paradigms the tester will write a series of tests (test cases) to detect any errors and to evaluate if the outcome of the test meets with the software design. We are having various methods for testing.

1. Black Box Testing
2. White Box Testing
3. Grey Box Testing
4. Thread Testing
5. System Testing

Various Software Companies following there own testing methodologies on projects. System Testing is last stage of testing process. Here we suggest a Template for check list of System Testing. Template is given in below Table. 1. Subsequent to integration testing a complete system or application has been developed with working interfaces. This does not mean that the system is necessarily complete. In order to be satisfied that a system is both entirely complete and correct, you would need to be confident that all of its intended functionality exists and that it performs each correctly under every foreseeable circumstance that is possible during its operation. System testing is an attempt to demonstrate if the program as a whole does meet its stated objective.

S. No.	Testing	Check
1.	Performance Testing	
2	Load/ Volume Testing	
3	Stress Testing	
4	Compatibility Testing	
5	Conversion Testing	
6	Backup Testing	
7	Recovery Testing	
8	Installation Testing	
9	Reliability Testing	
10	Usability Testing	
11	Acceptance Testing	
12	Functional Testing	
13	Maintenance Testing at Client Area	

Table. 1 Template for System Testing.

The Testing Process consists of following elements:

1. Test planning
 - a. Test preparation – test strategy

- b. Test planning – test plan
- c. Test design – test scripts
- 2. Test execution
- 3. Review Testing Process
- 4. Defect management
- 5. Release management

B. Testing Cycle

Although variations exist between organizations, there is a typical cycle for testing. The sample below is common among organizations employing the Waterfall development model.

- **Requirements analysis and design:** Testing should begin in the requirements phase of the software development life cycle. During the design phase, testers work with developers in determining what aspects of a design are testable and with what parameters those tests work. Analysis is a study that is related to tell it is correct, right, up to mark and consistent as planned. When requirements and analysis phase ends we can generate prototypes for it. To cross check whether they meet requirement what we planned in detail.
- **Test planning:** Test strategy, test plan, test bed creation. Since many activities will be carried out during testing, a plan is needed. Some companies follow templates for testing based on strategy and plan.
- **Test development:** Test procedures, test scenarios, test cases, test datasets, test scripts to use in testing software. Test is a set of input data and expected results that exercises a component with the purpose of causing failures and detecting faults shown in Table 2.
- **Test execution:** Testers execute the software based on the plans and test documents then report any errors found to the development team.
- **Test control:** After execute there should be cross check where test data resides, what we can make with the data, how we maintain it and where we want use it.
- **Test reporting:** Once testing is completed, testers generate metrics and make final reports on their test effort and whether or not the software tested is ready for release.
- **Test result analysis:** Or Defect Analysis, is done by the development team usually along with the client, in order to decide what defects should be assigned, fixed, rejected (i.e. found software working properly) or deferred to be dealt with later.
- **Defect Retesting:** Once a defect has been dealt with by the development team, it is retested by the testing team. AKA Resolution testing.
- **Regression testing:** It is common to have a small test program built of a subset of tests, for each integration of new, modified, or fixed software, in order to ensure that the latest delivery has not

ruined anything, and that the software product as a whole is still working correctly.

- **Test Closure:** Once the test meets the exit criteria, the activities such as capturing the key outputs, lessons learned, results, logs, documents related to the project are archived and used as a reference for future projects.

Table. 2 Attributes of the Class Test Case	
Attributes	Description
Name	Name of test case
Location	Full path name of executable
Input	Input data or commands
Oracle	Expected test results against which the output of the test is composed
Log	Output produced by the test

Table. 2 Attributes of the Class Test Case

C. Structure of a Software Quality Assurance

In order to derive a plan for SQA, we must revisit the elements of software quality assurance. The fundamentals of SQA deal with a planned activity to evaluate the development process during its progress. This plan or architecture must be placed around the entry to and the output from each stage of the development effort.

If the location and cause of the software defects or errors are taken into consideration during the software development, then there is a starting point for assuring the quality of each stage. These defects can also be considered in relation to the factors that affect the software quality. The classification of the causes of the defects can be addressed by SQA.

These combined factors that concern software quality, are the building blocks of an SQA Architecture as per Fig. 6 (V-model of verification versus validation).

V-model of verification versus validation

Fig. 6 V-Model

SQA is a continuously evolving entity with an emphasis on improving. There are three parts to this architecture; they are listed below.

The Architecture of SQA

SQA Component Activities	
1. Planning from the project initiation and	• Review and plan the project in its entirety

planning stage	<ul style="list-style-type: none"> Create the QA plan
2. Management of the Project life-cycle activities and components	<ul style="list-style-type: none"> Create a defect removal and defect injection prevention
3. Refactoring the Management of all SQA components	<ul style="list-style-type: none"> Instigate Software Quality improvement

D. Elements of Software Quality

Elements of Software Quality plan are given in Fig. 7. Based on list in Fig. 7, we can have tentative good results in Software Quality.

Elements of the Quality Plan	
1. Plan for entire project	Make a detail report
2. List of Quality goals	<ul style="list-style-type: none"> Quantitative – error severities Qualitative measurements (Downtime, response time, throughput etc)
3. Review Activities	<ul style="list-style-type: none"> Design review, test case reviews, etc
4. Software tests	<ul style="list-style-type: none"> Test strategy, plan, test design, environment etc
5. Acceptance tests	<ul style="list-style-type: none"> Test strategy, plan, test design, environment etc
6. Configuration management	<ul style="list-style-type: none"> Change control, version control etc.

Fig. 7 Elements for Software Quality Plan.

E. Software Quality Measurement

Software metric is a measure of some property of a piece of software or its specifications. Since quantitative measurements are essential in all sciences, there is a continuous effort by computer science practitioners and theoreticians to bring similar approaches to software development. The goal is obtaining objective, reproducible and quantifiable measurements, which may have numerous valuable applications in schedule and budget planning, cost estimation, quality assurance testing, software debugging, software performance optimization, and optimal personnel task assignments.

Common Software Measures

Common software measurements include:

- Balanced scorecard
- Bugs per line of code
- COCOMO
- Code coverage
- Cohesion
- Comment density
- Connascent software components

- Coupling
- Cyclomatic complexity (McCabe's complexity)
- DSQI (design structure quality index)
- Function point analysis
- Halstead Complexity
- Instruction path length
- Number of classes and interfaces
- Number of lines of code
- Number of lines of customer requirements
- Program execution time
- Program load time
- Program size (binary)
- Robert Cecil Martin's software package metrics
- Weighted Micro Function Points

For good software we can follow different Software Metrics.

After completing Requirements [5], Analysis and Design we move to Implementation phase. Where there is different mechanism for implementing software. In Fig. 8 Parallel Model, we see parallel two strips. One strip is related to Regular or Existing System based on that we plan phases and below to it is Software Project Development Process. They both run parallel to each other. At certain stage Software Project Development Process is merged to Regular System. To maintain Software Quality at each phase we cross check with the planned one. By doing we can achieve best Quality. Same way any modifications can be done if necessary to planned or Software Development Process. In Software Testing one can follow any metrics which will suit to the process or requirement. Software Testing is done based on Test Plan and is matched with requirement outputs. So that we can maintain or delivery quality Software to Client with short period with defect frees Software.

Fig. 8 Parallel Model.

Based on comparison between planned and current process we can eliminate errors or bugs within short amount of time cost.

IV. RESULT TO THE PROBLEM

Software Testing and Quality Assurance should take place parallel.

A. Software Quality Assurance Defect Removal

Considering that there are several factors that affect software quality there are a number of activities that

can be followed to improve the development stages in terms of software quality. The activities are discussed below.

- Reviews
- Magnified Monitoring
- Walk through
- Testing
- Configuration management

“An inspection and walkthrough is an improvement over the desk-checking process (the process of a programmer reading his or her own program before testing it). Inspections and walkthroughs are more effective, again because people other than the programs author are involved in the process. These methods generally are effective in finding from 30 to 70% of the logic-design and coding errors in typical programs”.

Procedural order and teamwork lie at the heart of formal design reviews, inspections or walk-through. Each participant is expected to emphasize his or her area of expertise. The knowledge that the work item will be reviewed stimulates the team to work to their upper end of productivity.

For different stages of the development process, there are different defects that get injected into the software. The rate of defect injection differs for each stage of development. The QA activities must match the defect injection rate and type to be effective at their removal. Fig. 9 demonstrates the distribution of defect injection for each of the four phases of the development process.

Fig. 9 Characteristic Distribution of Software defects origin.

Fig. 10 Average Defect filtering effectiveness by QA.

Fig. 10 Average Defect filtering effectiveness by QA is shown in green color whereas other color is regular one. By using QA techniques with testing may raise quality.

Fig. 9 and 10 identifies the effectiveness at defect removal by QA activity and development phase. Lastly the cost associated with the QA activities are listed in Fig. 11.

“Defect origins (the phase in which defects were introduced) are distributed throughout the development process, from the projects initiation to its completion” and associated with the QA activities that are listed in Fig. 11. By doing so we can raise the quality of the Software and also produce the best product.

Fig. 11. Average Defect Filtering effectiveness cost.

V. CONCLUSION

Most of the software projects are not success in history. Even tough they are related with cost and time. Our paper will help in better planning of Software, Organizing of Software, Controlling Software and Termination of the Software Project. We are having various tools in for better design, implementation of software and Test Software. It is mandatory to maintain quality of the Software. We combined Test and Quality of the Software to make it better. This is called Fast Track System for Software Development Life Cycle. We have less number of resources and we have to use it efficiently. For effective use we can hire new employees part-time or full-time, take existing employees, bring User to working area/ contact User by Electronic Gadgets, make templates so that they can be modified at any stage, combine test and quality actives together for better software. Parallel model used to compare existing System with Software Development Process. Based on comparison with Planned and Current Process we can maintain Software Quality. Following such kind of actives we can make software success.

VI. REFERENCES

- [1] Best Software Test & Quality Assurance Practices in the project Life-cycle by Mark Kevitt

- [2] Test and quality Assurance by Greenhouse Gas Technology Center
- [3] A new approach to requirement elicitation based on stakeholder recommendation and collaborative filtering by Nilofar Mulla and Sheetal Girase.
- [4] Software debugging, testing and verification by Hailpern B, Thomas J. Watson and Santharam P.
- [5] Software Testing Research Achievements, challenges, Dreams by Bertolino Antonia.
- [6] A Systematic review of software fault prediction studies by Cagatay Catal and Banu Diri.
- [7] Is Mutation an appropriate tool for testing experiments? by Andrews J. H, Briand L. C and Labiche Y.
- [8] The role of inspection in software quality assurance by Parnas D. L and Lawford M
- [9] Java quality assurance by detecting code smells by Van Enden E and Moonen L
- [10] Component-based software engineering: technologies, development frameworks and quality assurances schemes by Xiacai, Lyu M. R, Kam-Fai Wang and Roy Ko.