ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 5, October 2014

# Significance of Effective Defect Management Strategies to Reduce Pre Production Defects

Bhagavant Deshpande[1], Suma V[2]

Research Scholar, Department of Computer Science Engineering., JJTU, Rajasthan, India[1]

Dayanandsagar Research and Industry Incubation Centre, Dayananda Sagar Institutions, Bangalore, India[2]

**ABSTRACT:** Development of customer satisfied software is one of the prime considerations of any software industry. Hence, there exist several organizational strategies to accomplish the same. Since, quality has various dimensions, one of the dimensions of quality emphasizes upon generation of defect free or nearly zero-defect software. As such, a number of defect management techniques are incorporated during software development process. Nevertheless, existing techniques has not proven to achieve the above said target. The position of this paper is therefore to bring in the awareness of the defect management techniques, its significance on the quality and impact of existence of preproduction defects on the success of the project. This is achieved through an empirical investigation carried out in one of the leading software industry. The inference drawn from this case study brings in awareness and enables project personnel to accordingly allocate the resources and perform an effective capacity planning in the shop floor.

**KEY WORDS**: Software Development Life Cycle, Defect Management, Software Engineering, Software Quality**.**

## I. INTRODUCTION

It is essentially not possible to imagine the existence of any operational fields without the component of software. Every frontier of life is influenced by the application of software. Hence, it is very essential to ensure the generation of high quality software. However, with the emergence of technology and progress of science and engineering, there are several challenges and demands in retention of high quality.  This is because, human expectations towards standard of living has enhanced to an extent that holding on to such estimated quality levels becomes an around a clock activity in production systems. The term software quality has various definitions based on the viewpoints of the customer, user, and developer. Stephan H. Kan defines quality as an intangible concept that can be discussed, felt, and judged from the customer's perspective [1]. Definition of quality includes fitness for use when the customer's requirements and expectations are considered.

Quality definition must also consider the users' perspective. Watts Humphrey states that the principal focus of software quality is the user's needs [2]. Gerald M. Weinberg defines quality as value to some person [3]. However, quality adds value to business too. From the industry standpoint, the term quality has a different definition. Ram Chillerge defines quality as the existence of characteristics of a product that can be assigned to meet requirements [4]. According to Crosby, quality should meet the requirements [5].

Quality also means satisfaction of the requirements of the stakeholders such as clients, customers, and sponsors of the product, developers, and auditors. Another popular definition for quality is development of defect-free product. Stephan H. Kan [1] further defines quality at two levels. The first level is small q, which denotes intrinsic product quality. It defines the defect rate and reliability of the product. The second level is big Q which includes product quality, process quality and customer satisfaction. He further emphasizes that customer satisfaction can be improved by reducing defects and overall problems.

High quality software, thus, attributes to the development of defect-free product, which is competent of producing predictable results and is deliverable within time and cost constraints. It should be manageable with minimum interferences, maintainable, dependable, understandable, and efficient. It should serve the fitness for purpose [6].

A systematic approach towards development of high quality software is necessary due to the increased competitiveness in today's business world, technological advances, increase in hardware complexity and frequently changing business requirements [7]. Production of high quality software is hence, an innovative task that reduces the business risk through reduction in cost of development, time for rework and is an indicator of the maturity of the organization [8].

This paper therefore aims to bring in the awareness of defect impact on the success of the project and thereby a travel light to overcome the negative impact of the same towards retention of customer satisfaction and sustainability of the industry in the industrial market.

## II. LITERATURE SURVEY

Geoff Dromey suggested the curative and preventive approaches to tackle the defect related issues for the development of quality software. He stated that curative approach focuses on testing where both developers and users identify defects and fix them. In preventive approach, he emphasized on three aspects. They are i) anticipation of the defects, ii) use of formal inspections and prototypes to discover defects early in the development cycle iii) use of tools for detection of defects. He recommended prevention of defects as a better approach in comparison to curative approach. He emphasized on the consideration of quality requirements similar to functional requirements to facilitate complete benefit of preventive approach [9].

Defect Prevention is one of the most significant activities in the software development process. An analysis of defects at the early stage reduces the time, cost, and the resources required for rework [10][11].

Purushotham Narayan has proved that DP is one of the reliable components of software quality assurance in any project. He indicates the expensive nature of defects when identified late in the project. He states that the cost to fix a defect found at requirements phase after deployment of the product is 100 times the cost of fixing the same defect at the phase. Hence, he proposes an enhanced software process, which includes defect prevention strategy to achieve high quality and bug-free product. He explains the significance of defect prevention through a real time scenario as a case study. He shows that 60% of defects can be reduced due to the implementation of defect prevention activities in the development of the product [12].

Hence, early defect detection prevents defect migration from requirements phase to design and from design phase into implementation phase [13]. It enhances quality by adding value to the most important attributes of software like reliability, maintainability, efficiency and portability [14].

Gursimran and Jeffrey emphasizes on the defect detection and removal of both early and late faults in the software development as a primary focus for quality. They further described that regardless of advancement in the quality research, the development of quality product is still a challenge. They state that lack of understanding of the source of problems by the developers, inability to learn from mistakes, lack of effective tools and incomplete verification process are the major reasons for the introduction of defects. Hence, they suggest the need for research to provide more insight into the understanding of sources of the faults rather than just the fault themselves [15].

Vasudevan emphasizes on defect prevention in software industry. He states that identification and classification of defects play a vital role in the measurement-based process and in the product improvement. He further assigns severity level to the classified defects. He recommends defect prevention activities, which includes commitment from the management, creation of an action plan related to defect prevention activities, periodic review, defect measurement, and causal analysis of the defects [16].

Adeel et al., conducted a survey to study the impact of DP in industries. The survey results have shown that defect detection at the early stages is less expensive than detection at later stages. They have recommended several techniques like prototyping, use of CASE tools, training techniques, Quality Function Deployment (QFD) technique, and Joint Application Development (JAD) to avoid defects in the requirement analysis phase [17].

Thus, it is required to either eliminate or reduce the existence of defects during the production period such that rework overheads is tremendously cut down.

## III. RESEARCH METHODOLOGY

In order to analyze the impact of defect during the production cycle, this research is focused towards collection of information for pre production defects. The work is initiated through visits to various software industries whose maturity of the company is CMMI level 5. The intention behind this type of selective sampling of the software company is that a company of highest maturity will have lesser pre production defects since they adapt effective defect management strategies.

Further, with the selection of software industries for investigation purpose, the next activity in this research was to select the type of projects. This challenge is overcome by deliberate sampling of domains such as Healthcare and Retail as they are non critical applications and collection of required data was possible.

In addition to the above said challenges, this research then encountered yet another challenge of selecting the projects within the specified domains. Again, random sampling technique was followed to select few projects which are sampled in this paper.

The data was collected from the data centre of the company and also through interviews and face to face communications from the project personnel of the quality and project team. The obtained data set was analyzed to understand the impact of preproduction defects on the quality and success of the project.

## IV. CASE STUDY

This paper presents a case study of a software industry which is CMMI Level 5. Sampled 10 projects are presented in this part of the research work.

Projects P1 to P5 represents healthcare projects, which are developed using mainframe technology and COBOL programming language. The operating system used is windows and the process model followed is waterfall. It infers that requirements are well understood and hence defect existence should be as minimal as possible.

Projects P6 to P10 represents Retail projects are developed using web technology and Java J2EE programming language. Again, the operating system is Windows and followed water fall model.

It is worth to note that complexity of the sampled projects depicted in this work is medium size. Table 1. depicts the project profile in terms of time, cost and defect count which is measured to evaluate the customer satisfaction index.

Table 1. Project Profile to evaluate Customer Satisfaction Index

| Project Profile | P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 | P9 | p10 |
|---|---|---|---|---|---|---|---|---|---|---|
| PTime (*) | 2340 | 2640 | 3350 | 3440 | 3640 | 4350 | 4040 | 4170 | 3475 | 3920 |
| PCost (**) | 18290 | 19850 | 24320 | 16370 | 312290 | 34320 | 33280 | 34820 | 34820 | 37840 |
| PComp | 3.5 | 4 | 4.2 | 4.35 | 3.6 | 3.8 | 4.2 | 3.6 | 4.3 | 3.3 |
| DCount | 67 | 82 | 103 | 81 | 117 | 93 | 97 | 97 | 119 | 129 |
| CSI | 8.6 | 8.7 | 9 | 8.5 | 8.8 | 8.95 | 9.05 | 8.85 | 9.26 | 9.2 |

PTime- Project Total Development Time; PCost- Project Total Development Cost; DCount-Defect Count; CSI-Customer; PComp-Project Complexity; Satisfaction Index; (*)-Measured in Person hours; (**)-US Dollars

Table 1 infers that customer satisfaction index is evaluated using total project development time, total project cost incurred and defect count. The projects are arranged in ascending order of complexity in order to conduct analysis of the impact of preproduction defects upon these projects. In order to clearly understand the impact of defects in the preproduction cycle, this data is visualized on a graph where project development time and cost is scaled down to be in illustrative unit. Table 2 illustrates the scaled unit for evaluating CSI with total development time and total project cost. The projects are evaluated by arranging them in ascending order of the complexity and performing a comparative analysis between CSI and Complexity.

Table 2. Comparative Study of CSI with Project Complexity

| Project Profile | p10 | P1 | P5 | P8 | P6 | P2 | P3 | P7 | P9 | P4 |
|---|---|---|---|---|---|---|---|---|---|---|
| Scaled Unit (Pjt) | 3.92 | 2.34 | 3.64 | 4.17 | 4.35 | 2.64 | 3.35 | 4.04 | 3.475 | 3.44 |
| Scaled Unit(Pjc) | 37.84 | 18.29 | 31.22 | 34.82 | 34.32 | 19.85 | 24.32 | 33.28 | 34.82 | 16.37 |
| PComp | 3.3 | 3.5 | 3.6 | 3.6 | 3.8 | 4 | 4.2 | 4.2 | 4.3 | 4.35 |
| DCount | 129 | 67 | 117 | 97 | 93 | 82 | 103 | 97 | 119 | 81 |
| CSI | 9.2 | 8.6 | 8.8 | 8.85 | 8.95 | 8.7 | 9 | 9.05 | 9.26 | 8.5 |

Table infers that

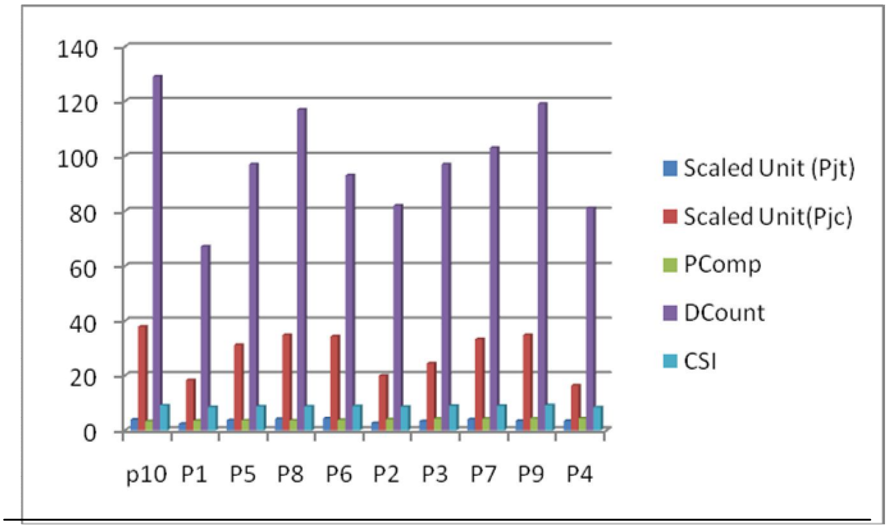
- Projects P5 and P8 are having same complexity.
- However, they are from different domains.
- Yet Customer Satisfaction Index is varying.
- This indicates that defect has an impact on satisfaction level of customers.
- It is worth to note that cost and time has increased in P8 despite of having higher level of customer satisfaction.

Similarly,

- Projects P3 and P7 are having same level of complexity but in different domains.
- Again, CSI is higher with even increased time and cost
- This is because the defect count is lesser than P7. Hence, CSI of P7 is more than CSI of P3

Figure 1 indicates the above inferences in a graphical mode.

Figure1. Graphical representation of impact of CSI and other project profiles



The work is limited to understand the impact of defects on customer satisfaction level. However, our forth coming work is to focus upon the parameters which have led to the introduction of defects in these projects and their root cause analysis.

## V. CONCLUSION

Due to the improvement in current day technology, the industrial market especially with software is sustainable only if the customer satisfaction is achieved. Hence, retention of high quality is one of the mandatory aspects of development systems. However, quality has various perspectives. Defect management is one of the vital perspectives of high quality. Thus, there exist various approaches to achieve quality. The aim of this paper is to bring in awareness in understanding the impact of pre production defects in the shop floor. The knowledge of impact of defects during production aids the project personnel to identify the right staff and allocate them accordingly such that preproduction defect count is reduced.

## ACKNOWLEDGEMENT

The authors would like to thank all the industry personnel who have helped in conducting this industrial investigation under the framework of Non Disclosure Agreement. We are thankful to their support, guidance and highly valuable suggestions.

## REFERENCES

[1] Stephen H. Kan: Metrics and Models in Software Quality Engineering, Addison-Wesley publisher, Edition 2, Boston, MA, USA, 2003.
[2] Watts Humphrey: A Discipline for Software Engineering, Addison-Wesley Professional publisher, ISBN: 0201546108, 1995.
[3] Gerald M. Weinberg: Quality Software Management: Systems Thinking, Dorset House publisher, 1997, ISBN-10: 0932633226, ISBN-13: 978-0932633224.
[4] Ram Chillarege: The Marriage of Business Dynamics and Software Engineering, IEEE Software, Vol. 19, No. 6, 2002, pp. 43- 49.
[5] Philip B. Crosby: Quality without Tears, McGraw-Hill Book Company, 1984, pp. 64.
[6] Ramaswamy. V, Pushpavathi T. P., Suma. V, "Position Paper: Defect Prediction Approaches for Software Projects Using Genetic Fuzzy Data Mining", 48th Annual Convention, Computer Society of India, Vishakapattanam Chapter, 13th -15th December 2013, Vishakapattanam, India.
[7] Ramaswamy V, Suma V and Pushpavathi T P, "Literature Review on Software Defect Management Approaches used in Software Engineering", International Journal of Computer Science", July, 2014

[8]K. Lakshmi Maduri, Suma V, “Comprehension of classification of Parameters Influencing Software Project Success”, International Journal of Research in Computers, Special Issue: International Conference on Intelligent Computing Applications. ICICA 2014, ISSN: 2278 – 1781, Vol. 1, Issue 2, pp139 – 143, 2014.

[9] Dromey R Geoff: Software Control Quality - Prevention Verses Cure?, Vol.11,Issue 3, July 2003, ISSN: 0963-9314, pp. 197- 21.

[10]Suma. V, T. R. Gopalakrishnan Nair, “Impact of Test Effort in Software Development Life Cycle for Effective Defect Management”, International Journal of Productivity and Quality Management (IJPQM), Vol. 13, No 3, 2014.
Web Link: http://www.inderscience.com/info/ingeneral/forthcoming.php?jcode=ijpqm

[11]Shashi Kumar, T.R. Gopalakrishnan Nair, Suma V, “SLI, a new Metric to Determine the Success of a Software Project”, International Conference on Electronics and Communication System (ICECS’14), Karpagam College of Engineering, Coimbatore, Tamil Nadu, 13th – 14th February 2014

[12] Purushotham Narayan: Software Defect Prevention in a Nutshell,iSixSigma.com, 11 June 2003, http://software.isixsigma.com/library/content/c030611a.asp

[13] Jeff Tian: Software Quality Engineering: Testing, Quality Assurance and Quantifiable Improvement, Wiley John & Sons publisher, New Jersey, 2005.

[14] Ciolkowski, M. Laitenberger, O. Rombach, D. Shull, F. Perry and D.Fraunhofer: Software Inspections, Reviews and Walkthroughs, 24th International Conference on Software Engineering, ICSE, Orlando, Florida, USA, May 19-25, 2002, ISBN 1-58113-472-X, pp. 641-642.

[15] Gursimran Singh Walia and Jeffrey C. Carver: “A systematic Literature Review to Identify and Classify Software Requirement Errors, Technical Report MSU-071207

[16] Vasudevan S: Defect Prevention Techniques and Practices, Fifth Annual International Software Testing Conference, India, February 21-22, 2005.

[17] Kashif Adeel, Shams Ahmad and Sohaib Akhtar: Defect Prevention Techniques and its Usage in Requirements Gathering-Industry Practices, International Conference on Engineering Sciences and Technology, Karachi, Pakistan, August 30, 2005, pp.1-5.