Microscopic Comprehension of P1 Type of Pre Production Defects in Software Development Process

Bhagavant Deshpande
Research Scholar
Dept. of Computer Science Engg.,
JJTU, Rajasthan

Suma. V, Ph.D.
Dayanand sagar Research and Industry Incubation Centre,
Dayananda Sagar Institutions,
Bangalore,India

ABSTRACT

Existence of defects during software development process is basically attributable to the intrinsic complex nature of software. Hence, various defect management techniques are adopted in software industries in order to enhance the level of confidence during software development process such that the end product is deployed with as minimal defects as possible. Nevertheless, there still reside defects which escape the production system and gets deployed onsite as defect leaks. However, it is not just the defect leak which is matter of concern, but the type of defect that had escaped from quality assurance team. This paper thus aims to provide a comprehensive examination of defect leak and its root cause analysis. The study is carried out in one of the leading product based software industry where projects for investigation comprise of non-critical applications. The inferences throw light on the possible root causes for occurrence of P1 type of pre-production defects.

Keywords

Software Development Life Cycle, Software Quality, Defect Management, Defect Types

1. INTRODUCTION

Ever since the inception of Software Engineering concepts, generation of software undergoes the entire process of software development life cycle. Complete production cycle has human intervention in addition to the use of automated tools. However, due to the complexity of software, the advancement of technology is still unable to produce defect free software. Defect in software is deemed to be an important issue that needs to be addressed instantaneously. This defect is observed as a flaw either in the product or observed during the production process. Defect left undetected or eliminated is prone to customer dissatisfaction [1].

Hence, software industry has implemented several defect management strategies in order to effectively manage the defects [2]. The main intention of this mode of operation is to ship the product with confidence to the customer’s site. A defect free software product is representation of high quality which thereby ensures the attainment of total customer satisfaction. Since, the industrial atmosphere is all the time proven to by dynamic and highly competitive, it has become an inevitable concern to realize complete customer satisfaction through the production of high quality software.

However, it is worth to note that software generation materialize through various phases of software life cycle which includes requirements engineering, design, code construction, testing and quality assurance scrutiny process [3]. Despite of implementation of high end technology and tools during the production process, due to the human intervention, development inexorably goes on with defect injection.

It is time to recall that defect injected needs to be detected and eliminated as close to its origin as possible. This is because defect is not static but propagates across the phases of development. Additionally, this proliferation has proven to have ripple effect on the quality of software. Hence, it is imperative to incorporate effective defect management techniques. Further, the impact of defect is not the same when injected at any phase or even when discovered at some phase. Besides, the impact of defect yet again varies with type of defect identified [4].

Thus, it is not just defect count which is of prime significance but the type of defect. This is because; nature of defect can be blocker, critical, major, minor or trivial. Further, the impact of each of these defects is estimated to be blocker or severe type of defect, work around or medium type of defect, cosmetic or low severity type of defect. Accordingly, blocker and major defects are deemed to be having high severity impact on the quality of the software. Major or minor nature of defect is considered to have an impact which is of medium severity type. Furthermore, trivial defect is the one whose impact is very low upon the quality of the software [5][6][7][8].

The aim of this paper is however focused towards analysing the impact of severe type of defect during the pre-production process.

2. LITERATURE SURVEY

Since, quality attainment is the motto of any software organization, there always persist various research and innovative thoughts towards accomplishment of the same. Works of various authors indicate that there is always a scope for betterment in the modes of developing software during the developmental process.

Authors of [9] have worked to introduce a qualitative and quantitative metric in order to enhance the quality of software generation process. They state that by improving software inspection technique, it possible to reduce the testing time and also it is possible to capture maximum number of defects. The aim of their work is to ensure that defects do not leak and get identified as customer reported defects [9].

Authors of [10] have further investigated maturity level of testing and recommends that testing should be given equal emphasize at all phases of software development. They feel that time given for testing especially at requirements and design phases are insufficient to capture all defects which are injected at the respective phases. They hence suggest that if defect is not identified and eliminated nearing to its inception point, cost, time rework and overhead of the project will increase resulting in customer dissatisfaction [10].

Therefore, authors of [11] have explored the significance of integrating exploratory testing during the testing life cycle such that maximum defects can be unearthed. They have compared the impact of testing without exploratory and also success nature of exploratory testing in capturing maximum number of defects using empirical investigation of software projects [11].

However, authors of [12] have proven that defects are requirements phase has severe impact on the quality of software [12]. But, authors in [13] have proven that a defect that gets injected in design is majorly due to improper coupling and cohesion. They have used graph theory to correlate the issues of design flaws so that it becomes possible to resolve them using mathematical visualization [13].

Authors in [14] have viewed the quality of software to be depending on efficiency of the project managers in effectively allocating the resources. The aim of their research is to enable one to assign right skilled personnel in order to ascertain development of nearly zero defect free software [14]. Further, authors in [15] have worked to prove that scope creep is one of the modulating factors to achieve quality of software. They have proven that impact of scope creep is observed on defect count, time, cost, customer satisfaction level and success of the project [15].

It is now time however to not only estimate or evaluate the impact of defect through defect prevention strategies but also to formulate defect prediction techniques [16]. Authors in [17] surveyed different data mining algorithms used for defect prediction in software and also discuss the performance and effectiveness of data mining algorithms [17].

With reference to the work of authors, authors of [18] defect prevention is given significance in software organizations. They hence strongly suggest that defect prevention should be emphasized at both the project and organizational level. Their work provides a general framework of defect prevention activities whose intention is to aim at reduction of post defect density in order to improve quality of software produced [18].

In support of the above stated authors, the work carried out by the authors of [19] positions towards reducing residual defects. The authors have thus identified the factors that influences defect injection and defect detection to reduce residual defects [19].

It is interesting to know about the work of [20]] who has presented the most common root causes for coding defects. He explains the expensive nature of cost of software defects when they remain undetected close to their injection points [20].

However, the aim of this research is to comprehend the impact of pre production defects during software development with the main area of focus to be on impact of existence of high severity defects as defect leak.

3. RESEARCH METHODOLOGY

Since, this research is to analyze the impact of defects during software development process, this research team has investigated various software industries. However, this paper presents an empirical investigation carried out in one of the leading product based software industry. This, randomly sampled industry is developing various non critical applications. However, this paper illustrates sample of Telecom projects which are developed in Java J2EE technology. Data for these projects are collected from Quality Assurance department. The data is further analyzed to understand the impact of pre production defects on the quality of the software and hence the customer satisfaction level.

4. CASE STUDY

An empirical investigation of software projects is carried out in one of the leading product based software industry. These projects are developed since 2013. The sampled projects are evaluated based on complexity. Since, the company follows complexity measurement in the levels of 1 to 5, a project having complexity 1 is deemed to be simple project while a project whose complexity is measured as 5 is considered to be highly complex project.

Table 1 depicts randomly sampled projects from the telecom domain. The table provides information about total project development time, cost for developing the entire project, complexity of the project, number of defects captured during the production cycle, number of defect classification, number of defect escapes, number of customer reported defects and customer satisfaction index of every project. Projects in the table are arranged in ascending order of total project development time since complexity is same for several projects.

Table 1 infers that total defects captured need not increase with either complexity or even with project development hours. However, cost has shown increase in the sorted projects. Further, these projects indicate increase in complexity. It may be observed that as complexity increases, total number of defects need not get increased.

Table further indicates total number of defect classification captured. The sampled projects depicted in this paper indicate only P1 type of defects being captured. This is because these projects are selectively presented in this paper in order to understand the reasons for this type of defects to be present during production process. The main aim of this manner of investigation is twofold namely

(i) to reduce the total number of defects in the production cycle

(ii) to reduce total number of defects which has high severity impact on quality of software

It may be noted that conventionally high severity defects are represented as P1 defects, P2 defects as medium severity defects while defects having low severity is represented as P3 defects. Table 1 infers that with increase in complexity and total number of project development hours, P1 defect count need not increase which is observed in project P4.

Further, it is worth to recall that a defect escape is treated as escape of those defects within the software development life cycle that has gone past in quality assurance section. Some defects however, escapes beyond quality assurance which is known as defect leak. However, they are detected and captured in user acceptance test (UAT) well before the production is completed and ready for deployment. Table 1 infers that total number of defect escapes also need not depend either on complexity or project development time as witnessed in projects P4 and P6.

Defects that cross detection beyond UAT are treated as customer reported defects since they are identified in the deployed state of the product in the customer’s site. Table 1 indicates that neither does complexity nor do the time has an impact on the total number of customer reported defects as seen in project P6. However, customer satisfaction level is all the time measured in a rate of 1 to 10 in industry. Accordingly, a Customer Satisfaction Index (CSI) rated 1 indicates least customer satisfaction while CSI of 10 indicates total customer satisfaction. Table 1 specified that all the projected depicted in this paper has CSI rated 9 and above.

Table 1 Pre Production defect profile for Telecom domain projects

Domain	**Parameters**	**Project -1**	**Project -2**	**Project -3**	**Project -4**	**Project -5**	**project -6**	**Project -7**	**Proejct -8**	**Project -9**	**Project-10**
Telecom	Project hours of development (*)	1260	1390	1460	1475	1890	2140	2850	3250	3440	4100
	Cost (**)	1400	1400	1800	1400	2100	16270	2900	3700	3200	4200
	Complexity (***)	3	3	3	3	3	3	4	4	4	4
	# of defects captured	67	72	79	62	91	84	92	107	99	114
	# of defects classification	6P1	9P1	13P1	8P1	14P1	14P1	17P1	19P1	19P1	21P1
	# of escapes	3	2	4	2	4	3	5	5	5	5
	# customer reported defects	1	1	1	1	2	1	2	3	3	3
	# customer satisfaction index(CSI)	9.3	9.2	9.1	9.3	9.2	9.2	9.2	9.1	8.9	9

(*)- Measured in Man hours; (**) – Measured in US Dollars; (***) – Measured on a scale of 1 to 5

Despite the above inferences, it is yet not apparent the impact of P1 defects on quality of software. Hence, further analyzing the various parameters as depicted in the table draws few more inferences as given below.

When P1 defect is less, CSI is more indicating that

$$P1\ defect\ count \propto CSI$$

(Eq. 1)

Since, P1 defects can cause problems both in the cycle and post production, it is required to reduce defects and analyze especially P1 defects. Therefore, it is imperative to capture P1 defects as pre production defects instead of getting it caught as post production defects by the customer. Other benefits of capturing P1 defects during pre production cycle include

(i) Reduction of testing time

(ii) Faster rate of delivery of the product to the customer

(iii) Increased productivity

(iv) Enhanced customer satisfaction

(v) Reduced scope for rework in terms of time and cost

(vi) Reflection of maturity level of the company.

Due to the aforementioned benefits resulting from the identification of P1 defects, this research further directed towards analyzing various root causes for injection of such severity defects. Our deep investigation brings out the rationale for the possibility of injection of P1 type of severity bearing defects. Table 2 indicates the enumerated root causes for P1 defects.

Table 2 Enumerated factors for possibility of P1 defect occurrences at various phases of software development

Phase of software development	Enumerated list of factors
Requirements Engineering	Requirements not well understood
	Requirements improperly translated into design
Design	Design characteristics not properly understood
	Design characteristics not properly coded
	Missing design characteristics
Code Construction	User Test does not cover all the functionalities
Quality Assurance/Quality Control	Tester improper understanding of functionality
	Inadequate/insufficient code coverage

	Improper selection of test cases
	Insufficient Regression Testing
	Environmental factors
	Configuration errors in terms of hardware/software platforms

Table 2 thus infers that for any of the above stated reasons, it is possible for the occurrence of P1 severity defect. A complete knowledge of P1 defects and its impact on CSI certainly ensures one towards either elimination or reduction of existence of P1 defects in pre production cycle as against its total avoidance of the same as customer reported defects.

5. CONCLUSION

Software production is one of the major thrust areas of any production systems. However, generation of high quality software is always a challenge to achieve. This is because of complete customer satisfaction is dependent on quality level of the project. Further, quality is dependent on defect count, type of defect and its impact on other project success deciding factors such as time, cost etc. This paper put forth a case study carried out in one of the leading product based software industry. An empirical investigation telecom project is carried out in order to analyze the impact of P1 type of defects on customer satisfaction index. Further, root cause analysis is carried out to explore the reasons for occurrences of P1 defect at various phases of software development in the production cycle. This knowledge enables one to aim at bringing out methodologies to either eliminate or reduce the existence of P1 defects.

6. ACKNOWLEDGMENT

The authors would like to sincerely acknowledge the industry personnel who has provided their invaluable guidance, suggestions and support in carrying out research under the framework of Non Disclosure Agreement.

7. REFERENCES

[1] V. Suma, T. R. Gopalakrishnan Nair, “DefectManagement Strategies in Software Development”,Book on Recent Advances in Technologies", ISBN978-953-307-017-9, pp 379-404, Intec webPublishers, Vienna, Austria, November 2009.

[2] V. Suma, T. R. Gopalakrishnan Nair, “Effective Defect Prevention Approach in Software Process for Achieving Better Quality Levels”, WASET Journal, vol.42, pp 258-262, 2008

[3] Roger S. Pressman: Software Engineering, A Practitioner’s Approach, McGraw- Hill International publisher, Sixth Edition, New York, USA, 2005, ISBN-007-124083-7.

[4] T. R. Gopalakrishnan Nair, V. Suma, “The Pattern of Software Defects Spanning across Size Complexity”, International Journal of Software Engineering (IJSE), Software Engineering Competence Centre of Ministry of Communications and Information Technology Publications, vol. 3, No. 2, pp.53-70, July 2010.

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

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

[7] 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.

[8] Mukesh Soni: Defect Prevention: Reducing Costs and Enhancing Quality, iSixSigma.com, 19 July 2006,

[9] T. R. Gopalakrishnan Nair, V. Suma, "Defect Management Using Pair Metrics, DI and IPM” CrossTalk, The Journal of Defense Software Engineering, Vol. 24, No 6, 2011, pp.22-27, 2011

[10] T. R. Gopalakrishnan Nair, V. Suma, Pranesh Tiwari, “Significance of Depth of Inspection and Inspection Performance Metrics for Consistent Defect Management in Software Industry”, IET Software, December 2012, Volume 6, Issue 6, pp. 524 – 535

[11] Rashmi N, Suma V., “Defect Detection Efficiency of the Combined Approach of Testing”, 48th Annual Convention, Computer Society of India, Vishakapattanam Chapter, 13th -15th December 2013, Vishakapattanam, India.

[12] B. R. Shubhamangala, Suma V, Manjunatha Rao L, “Multilayer security requirements model: Effective collaboration of compliance with Application security in IT Enterprises” 2013 IEEE International Conference, ICARDPET 2013, 29th – 30th March 2013, Nagapattinam, Tamilnadu, India

[13] Poornima U. S., Suma V., “Factors Modulating Software Design Quality”, International Conference on Advanced Computer Science and Information Technology (ACSIT), 10th March 2013, Chennai, India

[14] T. R. Gopalakrishnan Nair, Suma. V, Shashi Kumar. N. R, “Impact Analysis of Allocation of Resources by Project Manager on Success of Software Projects”, International Conference on Data Mining Computer, Communication and Mechanical Engineering (ICDCCME) 2012, 21st-22nd December, Bangkok, Thailand

[15] 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

[16] P. Singh, Comparing the effectiveness of machine learning algorithms for defect prediction, International Journal of Information Technology and Knowledge Management, 2009, pp. 481-483

[17] Suma.V, Pushpavathi T.P, and Ramaswamy. V, “An Approach to Predict Software Project Success by Data Mining Clustering", International Conference on Data Mining and Computer Engineering (ICDMCE'2012), pp. 185-1908

[18] Li Meng, Xiaoyuan He and Sontakke Ashok: Defect Prevention-A General Framework and Its Application, Sixth International Conference on Quality Software (QSIC'06), Beijing, China, October 27-28, 2006, pp. 281-286.

[19] Jef Jacobs, Jan Vol Moll, Rob Kusters, Jos Trienekens and Aarnout Brombacher: Identification of Factors that Influence Defect Injection and Detection in Development of Software Intensive Products, Information and Software Technology Journal, Vol. 49, No. 7, 2007, pp. 774-789.

[20] Ben Chelf: Avoiding the Most Common Software Development Goofs, Exploring the Root Causes of Many Coding Defects and Possible Solutions, Defect Analysis and Error Prevention – Sources, Dr. Dobb's Portal, http://www.ddj.com/architect/193001588?cid=RSSfeed_DDJ_ArchitectDebug. 2006.