# Framework for reducing post production defects in software industry

Divakar Harekal, Suma V

**Abstract:** Since, software has laid its impact on every field of operation, development of customer satisfied software is the prime hour of the day. In view of the fact that, defect is one of the major contributing factor to retain customer satisfaction. Production of software which is defect free is all the time the main objective of any software organization. However, accomplishment of the same is only plausibility. Though several defect management strategies exist in all IT industries towards development of nearly defect free software, yet there is a wide possibility of software to contain post production defects. This work comprises of a case study carried out in one of the leading software industry in order to explore the probability of post-production defect pattern. Hence, this research focused towards introducing an integration of a novel framework which ensures reduced post-production defects. In this framework, every activity of pre-production development of software has to be integrated with novel approaches to reduce post-production defects.

**Key words:** Software Engineering, Defect Management, Software Quality, Software Process, Software Development Life Cycle, Software Metrics and Measurements, Total Customer Satisfaction

## 1 INTRODUCTION

From the time of industrial revolution, people are emphasizing upon various modes through which effective sustainability of the organizations can be achieved. Since the time of modern era, the introduction of software has made an impact on the live hood of the society. In fact software has become one of the inevitable organs of any system in this globalization era. Therefore, every organization which is using software components in their systems is working towards generating high quality software.

Defect is any deficiency found either in the end products or in the deliverables [1]. Also, defect can be caused due to inaccurate ways of developing the product, imprecise process of implementing the activities ultimately resulting in the final end product to be declared ineffective or defective. Thus, effective defect management is one of the most decisive activities in software industry. The awareness of defect detection methods enables defect prevention [2][3].

This is because by detecting the defects during the time it gets injected either in the product or in the deliverables or even during the process of coming out of deliverables also enables one to handle the defects effectively. Detection certainly ensures one to understand the defect, its nature, its impact and its vitality in getting rectified. There are therefore several defect detection techniques such as testing, correctness proof etc which helps the developers to find and fix the defects. However, if these defects are not identified at the earliest, it has led towards failure of the software or even losing of customers [5][6][7].

The main reason for above said claim is that cost, time for rework of these detected defects increases with delayed detection process. Several researchers have therefore worked towards techniques which enable one to conduct early defect detection during the development process. In view of these points, this research started to investigate the defect impact especially when the product is deployed into the customer's site.

------------------------------------------------------------------------

- *Divakar Harekal is aresearch scholar at JJTU Rajasthan in computer science and engineering in India. E-mail: divakarhv0@gmail.com*
- *Dr Suma V is currently, Head of Dayanandsagar Research and Industry incubation Center India. E-mail: sumavdsce@gmail.com*

## 2 LITERATURE SURVEY

Purushotham Narayan [15] proposes an enhanced software process, which includes defect prevention strategy to achieve high quality and bug-free product [15]. Vasudevan [8] 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 [8].

Jay Xiong and Jonathan Xiong [12] have proposed a new model called Defect Prevention and Traceability model in software development. This is a non-linear model, which supports both forward and backward traceability [12]. Pankaj Jalote et al. [9][10] states that development of high quality software product can be realized when the product has few defects [9][10].

Hence, an awareness of all facets of defects enables the practitioners to achieve effective defect management and develop products with minimal defects. Kollanus and Koskinen [11] present a survey of research work in the area of inspection, since 1991 to 2005. They further express that research in the direction of the introduction of novel inspection methods is very less [11].

DP technique reduces the number of defects in the development life cycle. It further enhances business performance. Thus, investing in defect prevention reduces the cost of defect detection and elimination. Defect-free product reduces support cost, programming cost, development time, and competitive advantage. It therefore has a direct and strong impact on the time, cost, and quality of the deliverables.

## 3 RESEARCH TACTIC

From various research observations made all throughout these many years by various researchers, it is found that rework time and cost is extremely expensive especially if the defect is detected during post production phase. To get more clarity on the above facts, this research progressed to find more information about the types of defects, their occurrences in the developmental phase. Thus, various software organizations were visited. However, the population of software organizations are so high, that this research redirected towards study of selected software companies. Since, the aim of this research is to reduce post production defect as it is extremely expensive to fix at that point and needs complete rework of the project, this research focused towards those organizations which are following defect detection and prevention schemes. These companies will have strategies which will make our research to get more empirical information and thereby help us to come to with solutions to reduce post production defects.

Hence, this research marched towards selecting only CMMI level 4 and 5 matured organizations. Among those randomly sampled industries, there were yet several challenges to face. One such challenge is that all these industries are again developing huge number of projects of various types and domains. In order to resolve this challenge, again this research moved towards selecting only projects of domains like telecom and healthcare. Since, these projects were developed during 2013 to 2014, it was possible to analyze the current defect patterns in this recent era. These projects were developed using COBOL programming language.

Sources of data collection included data collection from log, defect prevention centres, from quality assurance departments, from project personnel using face to face communications, records, and emails using questionnaire etc. Data thus collected was analyzed and the inferences drawn are explained in subsequent sections.

## 4 RESEARCH WORK

The objective of this part of the research is to discover the connotation of existence of post production defects in software industries. This research now progressed to present a case study carried out in the sampled leading software industry. Table 1 thus presents a sample of 5 projects which are drawn from varying project complexity. These projects are having a complexity between 3 and 4 indicating that a complexity level of 1 is simple project and level 5 complexity in projects indicate the highest degree of complexity since in this set of samples from the industry, complexity is measured in a scale of 1 to 5. The table further provides information about defect escape, customer reported defects, time and cost of rework.

Table 1. Defect Summery of the sampled projects

| Parameters | Project-1 | Project-2 | Project-3 | Project-4 | Project-5 |
|---|---|---|---|---|---|
| Project hours of development(*) | 1390 | 1890 | 1460 | 2850 | 3440 |
| Cost (**) | 1400 | 2100 | 1800 | 2900 | 3200 |
| Complexity (to the scale of 1 to 5) | 3 | 3 | 3 | 4 | 4 |
| # of defects captured | 72 | 91 | 79 | 92 | 99 |
| # of escapes | 3 | 4 | 4 | 5 | 5 |
| # of customer reported defects | 1 | 2 | 1 | 2 | 3 |
| # customer satisfaction index(CSI) | 9.2 | 9.2 | 9.1 | 9.2 | 8.9 |
| Rework Cost(**) | 300 | 550 | 400 | 600 | 670 |
| Rework Time(*) | 12 | 22 | 18 | 24 | 27 |

(*) - Measured in person hours; (**) - Thousand US Dollars

### 4.1 INFERENCES FROM TABLE 1

Table 1 thus presents a sample of 5 projects which are drawn from varying project complexity. These projects are having a complexity between 3 and 4 indicating that a complexity level of 1 is simple project and level 5 complexity in projects indicate the highest degree of complexity since in this set of samples from the industry, complexity is measured in a scale of 1 to 5. The table further provides information about defect escape, customer reported defects, time and cost of rework.

From the empirical investigation carried out in those software industries with healthcare and telecom domain projects, the defect captured by testing team and defects captured by user acceptance team are identified. It was observed from the study that number of defects increases with complexity. However, this growth is not exponential indicating that with increase in complexity, the probability of defects getting injected also increases and that this increase need not be linear. However, the count of defects in complex projects will be certainly more than the count of defects that one can predict to be introduced by the development team of low complexity projects.

### 4.2 Progress Of Research Due To Investigation Inferences

From the investigation of the empirical projects, in this research, it is found that if project has a complexity of low, then defect count in terms of defect escapes will be less than 3. It is also observed that whenever complexity of project is medium, defect leaks will not be more than 3 to 6. It is important for organization to always maintain a healthy customer satisfaction index. Hence, it was found that total number of defects reported by customers has a direct relation with their satisfaction levels. However, this count of customer reported defects is in turn depending on the probability in which defect leaks were finally captured as a last remedy during pre-production phase of software life cycle. Also, it was found out that the count of defect leaks is in turn depending on the best ability of the testing team to capture maximum defects during their testing phase of the software process. Consequently, it is the procedures that are followed during pre-production phase that affects the post-production defect count. This study enables us to develop a test framework that can detect all defects in the development life cycle. Such a framework can only be possible by building quality in every step of the development cycle.

### 4.3 Introduction Of Framework For Reducing Post Production Defects

Though, several researches as indicated in the above literature survey indicates various techniques, methods, tools for defect prevention, yet there is still a challenge to achieve complete defect prevention status in any project. Hence, in this research, it is recommended to follow the novel framework for reducing post-production defects using the steps suggested to be strictly followed during the pre-production activities. The entire framework suggested in depicted in the figure 1.

**Framework that reduces post-production defects**

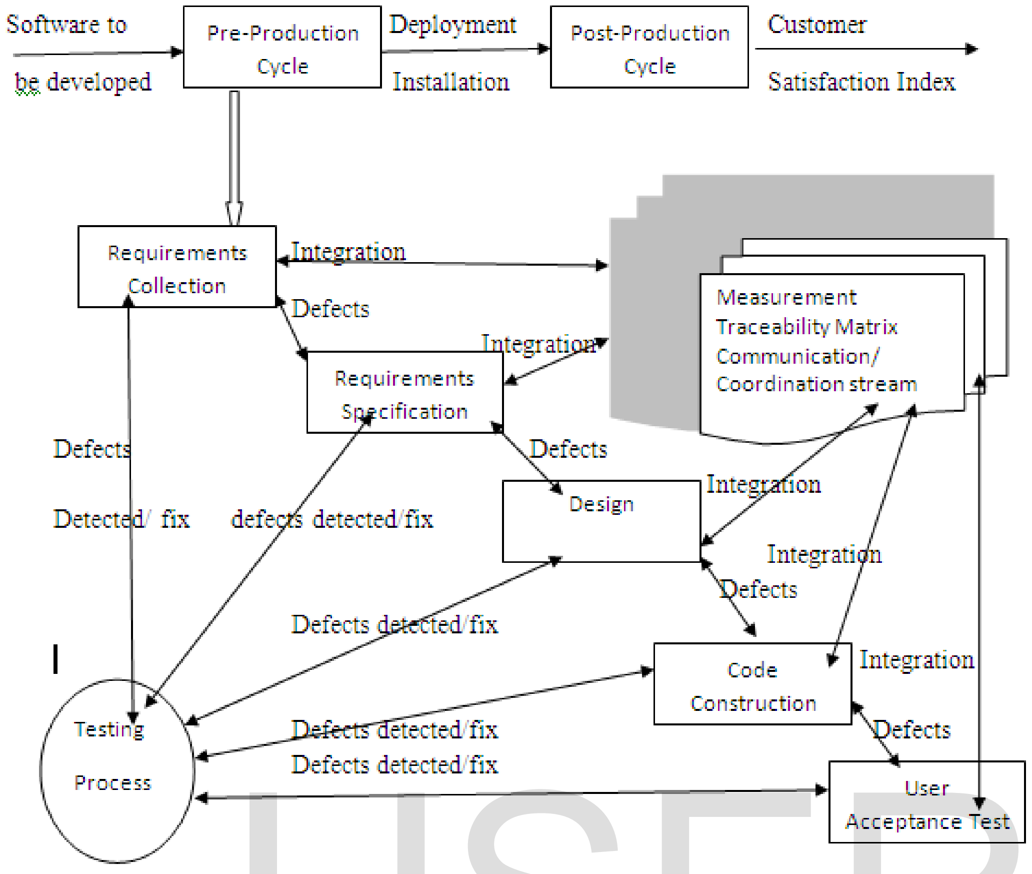

Figure 1. Framework to reduce post-production defects

This approach involves normal process that is followed and the suggested actions during the normal flow of activities such that pre-production defect count is reduced. Hence, in this part of the research, both normal flow of process and the suggested remedies are both explained. Further an explanation about impact of normal flow of operations and benefits that are ensured to be realized due to the following of remedial actions as per the new framework is also explained.

**Normal Flow of activities: Step 1: Requirements Collection**

- According to the framework, software that needs to be developed will as usual undergo developmental process.
- Until the product is deployed to the customers, it is in pre-production phase. During the beginning of pre-production phase, as per the habitual mode of working, requirements are collected from the potential stakeholders. Stakeholders are those people who are directly involved in the application or end product which is getting developed.
- These are the people who will be directly affected by the product. Hence, requirements are collected from them.
- The requirements collection activity is an oral communication task which involves understanding the problem better. Modes of collecting requirements can be face to face communication, telephonic conversations, interviews, video conference etc.
- Entire task is undertaken using natural language namely English especially for non-critical applications. Whenever applications require extremely high quality and fully defect free product many a times, specifications are collected using S language, Z language and other such mathematical modelling techniques.
- Further, it is worth to note that testing of collected requirements does takes place in many organizations to ensure defect less requirements elicitation process. Sometimes, these testing involve static verification such as reviews or inspections.

**Outcome:**

- The above stated entire process involves collection of requirements from various streams of people.
- Hence, there is always a wide possibility of not understanding the requirements clear, not able to express precisely, possibility of misinterpretation, miscommunication, political and economical barriers which hinders the requirements collection process to be effective.
- All these reasons attributes to requirements being defective in nature. Some amount of defects is however captured due to validation approaches like reviews and inspections.

**Impact:**

- Since, stringent actions of validation is not yet emphasized, due to defective requirements, subsequent activities in the developmental process further get affected.
- This is because what is collected will be specified in the requirements specification documents based on which further development and testing occurs.

**Enhanced Actions:**

- Testing is not just enough for validating the requirements elicitation process but needs a logical step in the cycle to measure defect injection in this activity such that these measures should weed them out during that phase and not allow them to pass on to the next phase. Hence, it is required to introduce measurements to measure the quality of requirements collection process, effectiveness of the collection process and efficiency of the analysts in collecting the requirements.

- Further, solution is to bring in the concept of strong traceability matrix updated frequently for every change that is made in the requirements collection process and in fact the entire communication process and carries through until testing.
- It is important to communicate these changes to all the downstream phases in such a way that the corresponding documents are updated and the latest version are available to all users, stakeholders and analysts.
- The entire software development process is akin to an orchestra where precise coordination is the essence of the game
- In order to achieve this close coordination we need to put in place proper checks and balances.
- These checks and balances could be in the form of checklist, template, inspection sheet and reporting tools
- The communication plan is an important document which brings in all the coordination together. A sample of that document is attached o the appendix.

**Benefits:**

- Due to following of above said integration of actions in the normal flow, certainly requirements collection process ensures reduced defective requirements elicitation.
- Hence, requirements become more complete, consistent, precise and accurate.
- In turn, defects addressed due to reasons of requirements collection process are reduced.

**Normal Flow of activities: Step 2: Requirements specification**

- The collected requirements are analyzed, prioritized and finally specified in the requirements document popularly known as software specification.
- The specification is usually a document which is specific to the company's template. It includes functionalities to be developed, constraints in which they have to developed and operated, quality requirements that needs to be met for customer satisfaction and so on.
- The document is usually written in English language and hence possible misinterpretations of statements are wide.
- The document further contains models to make things clear. Validation however does occur even for this activity in the form of reviews and inspections.
- Nevertheless, there still preside incomplete requirements specifications in the document.

**Outcome:**

- Due to misunderstanding or incomplete collection process, specification also becomes incomplete and ambiguous.
- Due to specifications being in natural and unstructured language, defects occur.
- A defect in this document may be misinterpreted requirements, incomplete requirements, skipped out requirements, masked requirements due to some political or economical backgrounds, lack of clarity on the requirements etc.

**Impact:**

- Due to defective requirements specifications, subsequent activities get affected. Hence, about 58 percent of total defects in any project are attributed due to requirements engineering process.
- It is also proven that defects due to requirements is very expensive in terms of time, cost for rework.

**Enhanced actions:**

- It is recommended that requirements specifications be validated using measurement approach where metrics are introduced to analyze the completeness, preciseness, ambiguity in the requirements specified.
- Though, these are existing, it is quite evident that practise of these metrics and measurements are still immature.
- Further, qualitatively measured and quantitative analyzed specifications should be dynamically updated in traceability matrix such that maintenance of these requirement changes is taken care and defects are reduced.
- Additionally, measured, traced and maintained specifications has to be completely communicated and coordinated among the downstream and upstream people such that it opens opportunities for bringing in clarity about specifications among all the members of the team and also enables to discover hidden defects due to brainstorming approaches.

**Benefits:**

- Due to measurements of specifications, it is now possible to analyze the quality of specified requirements.
- Defects are uncovered in greater extent due to communication and coordination policies
- Reduction of requirements defect certainly brings down defect count in a larger extent and its rework time, cost resources to fix such defects reduces to a greater extent.

**Normal Flow of activities: Step 3: Design**

- Once requirement specifications are completed with validation check in the normal flow of activities, one copy of the specifications is handed over to chief architect to start the design process.
- The architect comes out with system level architecture such that entire collection of subsystems, its relations with each other and rationale for choosing such an architectural pattern sis justified.
- Hence, it becomes critical for the right choice of architectural pattern for the requirements specified.
- With the selected pattern, high level design of components is made. It includes decisions for the relationship between inter components and intra components, classes included in the components, interfaces between the components etc.
- Having made the high level decision on the components and its details, low level design is undertaken which includes data structure choice and algorithms for implementation of pin to pin details of these components, their logic etc.
- All these activities are further validated with reviews and inspections in the normal flow of process.

**Outcome:**

Design is made using design languages such as UML and uses various other supporting modelling diagrams like data flow diagrams, schematic diagrams etc.

- Tools are extensively used for the above purpose. However, these design decisions are highly creative which binds the technology to the user needs.
- The design specifications are given to the programmers to come out with code which is implementable.

**Impact:**

- It is found that about 27 percent of defects in any project are attributed mainly due to design imperfections.
- These further lead towards programmers developing code for imperfect design.
- Since design is not flexible due to the inapproarpirate choice of archeitetural patterns or desings decisions, it is not possible for various systems to get it adaptable to changing requirements.
- This further leads towards nonoperational functionalities in case of changing requirements, which in turn acts as need for modifications. In case of non tracking of these change requests, it leads to defects.

**Enhanced Actions:**

- Since, design is one of the creative activities of software development process, it is required for analyzing the effectiveness of choice of design patterns, architectural styles and design decisions using measurements.
- A qualitatively and quantitatively measured design by introduction of metrics for measuring all the quality constrains such as coupling, cohesion, relationships etc enables reduction of design defects.
- Any decision on alternate solutions to design has to be dynamically updated in traceability matrix such that further enhancements to design leads to reduced defect count.
- Additionally, communication of design decisions and tactics used in the development process needs to be communicated and coordinated among the team of developing personnel so that it ensures discovery of flaws, elimination and thereby enhancement of quality of design.

**Benefits:**

- Integration of measurements for every choice of design decisions and tactics ensures reduced injection of design defects
- Maintenance of traceability matrix for every changes incorporated in the design ensures clarity in design modifications at the times of enhancement of the project or maintenance of the project
- Due to reduced number of design defects due to complete awareness of design tactics to all the project personnel, it ensures reduced programming defects. This is because design decisions are well known to the developers and testers and also to the analysts so that their developmental activities are incompliance with the design made.
- Due to reduced design flaws, it is now possible to go with minimal rework cost and time in case of any customer reported bugs.
- Reduced design defects enable the product to be more adaptable, portable and interoperable.

**Normal Flow: Step 4: Code construction**

- Once design is decided and completed, the low level design document is handed over to the programmers for them to start their coding activities.
- Code is developed using some programming languages and in a selected technology.
- Choice of technology selection, language selection is also a part of requirements phase. Based on the design selected modules are handed over to various teams to construct code to them.
- As a part of this activity, programmers conduct unit testing to find out defects injected due to code written.
- These defects may be syntax or semantic defects.
- Hence, testing is performed both statically and dynamically to the code.
- Static testing is conducted using reviews and walkthroughs.
- Sometimes inspections are also conducted to detect and eliminate static defects.
- Static defects are those defects which can be detected and eliminated by visual examination of the deliverables.
- However, dynamic defects are those which are detected only using testing of implementable code.

**Outcome:**

- Due to code generation, project development is now complete such that it can be given to the clients for their validation process.
- Code developed consumes most of the effort and it is this activity which involves large number of people working when compared to other phases of software development.
- Since, code is written by programmers and involves coordination of large number of people, their skill set and competences in the programming languages, effectiveness of this part of the project is highly people oriented.

**Impact:**

- It is observed that code defects are around 16 percent of total defect count in any project.
- Due to involvement of technology, people and skill set, based on the complexity of projects, defects injection is decided.
- Projects with well known technology, highly used programming languages where their pros and cons are well aware, it is quite possible for defect to be lesser in number.
- However, with increasing complexity, code defects also gets increased.

**Enhanced actions:**

- Though unit testing is undertaken during coding activity, yet programmers are not able to develop reduced pre-production defects. Hence, it is required to measure the quality of code written and it impact
- There is thus need for introducing qualitative and quantitative metrics which measures the skill set, depth of domain knowledge of the programmers, their ability to develop quality code in optimal effort
- Every change made to the code should be dynamically maintained as per the traceability matrix so that every fix of a defect in code does not lead to bad fix.
- Every modifications made to the modules should be coordinated and communicated so that chain defects and ripple effect of defects are reduced.

**Benefits:**

- Due to integration of quality measurements, dynamically updated traceability matrix and complete cooperation, coordination and communication philosophy in the code construction phase with all project personnel upstream the ladder of the organization and downstream the ladder of organization of the project chart ensures reduced defect count.

- Due to reduction of coding defects, defect to be captured due to defect leaks will reduce indicting effective project development process
- Further, reduction in defects captured reduces testing effort and improves code quality

**Normal Flow: Step 5: User Acceptance Test**

- Once testing is completed, the product developed will be put forth for user acceptance test popularly known as UAT.
- This activity is the transition period for the entire software development process where the product is either accepted by the customers or rejected or subjected for revisions.
- This team of users tests the product for cross verifying if all the specified requirements as per the SRS is in compliance with the operations of the product or not.
- It is also here that those defects which were undetected during testing process are discovered. Such defects identified by the UAT team are called popularly as defect leaks or defect escapes.
- Effectiveness of defect capturing at this point of development ensures reduced post-production defects.
- This is because, this is the last opportunity for the developing personnel to weed out the defects before it is deployed to the field
- This acts as the last quality gate crossing which every defect becomes highly expensive to fix.

**Outcome:**

- Due to user acceptance test, any defects which have escaped from the testing process due to reasons like time constraints, resource constraints etc can still be discovered.
- The maximum capture of defect escapes ensures higher quality level
- Detection of defects by the user acceptance testing community brings in confidence for the developing team to deploy the product to the customer site.

**Impact:**

- It is observed that higher the number of defect escapes lesser the number of customer reported defects.
- Once the product is deployed to the customers, it is very expensive to fix the defect.
- Some of the reasons being accepting the modifications to fix the defect may sometimes act as a new project which involves allocating project team and time, cost, resources etc.
- Further, the team who developed may not be the team who will fix the defects as per the needs of the customer.
- Hence, it becomes time consuming and the learning curve will be more. Learning curve is the time required for project personnel to read, understand and analyze the deliverables.
- If the detection of defects is in house before deploying it to the customers, the people who were involved in the development phase will be still in the project wherein the learning curve is almost nil.

**Enhanced Actions:**

- Integration of measurements in user acceptance activity provides users to understand the product and thereby evaluate the product in the level of customer required needs.
- Metrics which measures the satisfaction level of customers needs to be introduced such that it enables the developers to develop the product as per the customer wants and hence need not expect the risk of defects occurring in the operational phase.
- Every defect detected during user acceptance test needs to be dynamically updated in the traceability matrix such that maintenance of the product or enhancement to the product is easier and accurate
- A good communication, coordination and collaboration of users with the developing team certainly make sure that product is ultimately developed as per the specifications of the customers.

**Benefits:**

- Due to the integration of measurements in the user acceptance test activity, it becomes possible to comprehend the depth in which defect leaks are captured by the customers
- Metrics further enables the customers to know about the quality of the product.
- Measurements acts as quality indicators and eye opener for customers to justify for the amount negotiated and efforts put to develop the product.
- Due to dynamic updating of every defect captured and fixed, it is easy for maintenance of the product and customers are aware of effort put by the developing personnel towards fixing of defects.
- Communication of customers with all streams of developing team enables them to understand the reasons for developing the project in the way it is completed, its justifications, its strengths and limitations.
- Due to collaboration and coordination, every project developed leads towards reduced post-production defects and enhanced customer satisfaction levels.

Thus, integration of this framework in the normal development life cycle ensures reduced pre-production defect injection, increases defect detection and elimination process. It further enables detection of defects close to the point in which the defects were introduced by the developing personnel. Closer the detection of defects to its origin, reduced is the cost, time and effort required to fix the same. It further enhances customer satisfaction as customer report defects also known as post-production defects are greatly reduced.

## 5 CONCLUSION:

The total customer satisfaction promotes the continued existence of a software industry. Effective defect management is the fundamental activity to attain complete customer satisfaction. The two successful approaches of effective defect management are defect detection and defect prevention. Awareness of defect pattern enables better defect detection and defect prevention in software development.

Thus, an empirical investigation was carried out across projects and in inferences was drawn. Hence, this research focused towards introducing an integration of a novel framework which ensures reduced post-production defects.

In this framework, every activity of pre-production development of software has to be integrated with novel approaches to reduce post-production defects. Therefore, every activity such as requirements collection, requirements specifications, design, code construction and user acceptance testing needs to be subjected to integration framework. Integration formwork includes introduction of measurement systems at all levels of production such that

every stage is meticulously tested for quality of development and ensures detection of defects in case of their injection nearing to their origin. Further, the framework emphasizes upon dynamic updating of defects details and changes incorporated due to change request made in the developmental activities. This updated information enables the maintenance and enhancement of the product very easy and clear. This clarity further reduces the scope of defects to be injected and detected at later stages of post-production. Further, the new framework includes complete interactions of the developing team with customers. Hence, every activity is communicated, coordinated and collaborated among the upstream and downstream people in the project organizational hierarchy which reduces the scope of defects getting introduced due to miscommunication, misinterpretations, ambiguity and so on. Hence, reduced pre-production defects ensure higher degree of reduction in post-production defects. Reduced pros-production defects certainly bring in enhanced customer satisfaction index. This framework thus acts as a solution towards reducing post-production defects in any software projects in any software developing organization.
The analysis is applicable to small, medium and large projects developed with similar technology, environment and programming languages. However, these observational inferences may not be applicable to innovative projects and projects developed with dissimilar platforms.

## 6 ACKNOWLEDGMENT

Authors would like to thank all the industry personnel who have enabled this research to progress with their valuable suggestions, data and ideas under the framework of non-disclosure agreement policy.

## 7 REFERENCES

[1] Suma V. and Gopalakrishnan Nair T.R.: Effective Defect Prevention Approach in Software Process for Achieving Better Quality Levels, International Conference on Software Engineering, WASET, Singapore, August 29-31, September 01, 2008.

[2] Suma V.andGopalakrishnan Nair T.R.: Enhanced Approaches in Defect Detection and Prevention Strategies in Small and Medium Scale Industries, IEEE 3rd International Conference on Advances in Software Engineering, Sliema, Malta, October 26-31, 2008.

[3] Suma V. and Gopalakrishnan Nair T.R.: Better Defect Detection and Prevention Through Improved Inspection and Testing Approach in Small and Medium Scale Software Industry, International Journal of Productivity and Quality Management (IJPQM), Vol. 6, No. 1, 2010, pp.71-90.

[4] T. R. Gopalakrishnan Nair, Suma. V, Pranesh Kumar Tiwari, "Analysis of Test Efficiency during Software Development Process", Submitted, 2nd Annual International Conference on Software Engineering and Applications (SEA 2011), Singapore, 12th -13th December 2011

[5] T. R. Gopalakrishnan Nair, V. Suma, "Implementation of Depth of Inspection Metric and Inspection Performance Metric for Quality Management in Software Development Life Cycle" International Journal of Productivity and Quality Management (IJPQM), InderScience Publishers, USA, 2011.

[6] 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.

[7] T. R. Gopalakrishnan Nair, V. Suma, PraneshTiwari, "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.

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

[9] PankajJalote, Dinesh K., Raghavan S, Bhashyam M.R. and Ramakrishnan S. Quantitative Quality Management through Defect Prediction and Statistical Process Control, 2nd World Quality Congress for Software, Japan, September 2000.

[10] PankajJalote, "An Integrated Approach to Software Engineering", 3rd Edition, Narosa Publishing House, ISBN: Narosa: 81-7319-702-4

[11] Sami Kollanus, Jussi Koskinen Survey of Software Inspection Research: 1991-2005

[12] Jay Xiong and Jonathan Xiong "**Dpt Methodology: the defect prevention and traceability - driven methodology for software engineering.**" **C** Conference: Proceedings of the International Conference on Software Engineering Research and Practice & Conference on Programming Languages and Compilers, SERP 2006, Las Vegas, Nevada, USA, June 26-29, 2006, Volume 1

[13] Suma V. and Gopalakrishnan Nair T.R.: Influence of Skill of Inspectors in Effective Defect Detection and prevention, Masters and Doctoral Symposium, Bangalore, 2008.

[14]V. Suma, T.R. Gopalakrishnan Nair, "Defect Management Strategies in Software Development", Book on Recent Advances in Technologies", ISBN 978-953-307-017-9, pp. 379-404, November 2009, Intecweb Publishers, Vienna, Austria.

[15] MukeshSoni: Defect Prevention: Reducing Costs and Enhancing Quality, iSixSigma.com, 19 July 2006, http://software.isixsigma.com/library/content/c060719b.asp

[16] Raj Kothari C.: Research Methodology Methods and Techniques, Second Edition, New Age International (p) Ltd., publishers, Delhi, 2004.

[17] Grady Booch: Object-Oriented Development, IEEE Transactions on Software Engineering, Vol. 12, No. 2, February 1986, pp.211-221.

[19] Ian Somerville: Software Engineering, Edition 6, Pearson Publications, Chapter 20, 2004, ISBN-10: 0321313798, pp. 440-466.