Defect Prevention

Software Quality Management- Assignment 4

Navneet Chamala
Department of Software Engineering
Blekinge Institute of Technology
nach14@student.bth.se
940207-1477

Dilip Somaraju
Department of Software Engineering
Blekinge Institute of Technology
diso14@student.bth.se
930720-3373

***Abstract*— The current document is a report containing our thoughts and opinions on defect prevention in the software engineering context. The report charts the various methods or techniques employed in defect prevention and their advantages and disadvantages. We also provide a motivated account of our apprehensions on which technique would be most suitable for a phase in software development. The cost and benefit analysis of the defect prevention analysis is also stated in the document. All in all, the report is an attempt to give an overview about the concept of defect prevention.**

Keywords—defect prevention; software engineering; software quality; styling; insert (key words)

I. Introduction

A defect is defined as a shortcoming or a flaw in a product. In any context, defect prevention is one of the most important and primary goals while developing or building a product[1]. The number of defects has a lot to do with the quality of the product. It is obvious that a product is considered good if it has less number of defects. “Prevention is better than cure.” This quote also holds equally well in the software engineering context.

In the current report, we attempt to provide answers to the following questions:

- The team’s interpretation of the definition of Defect Prevention.
- Positives and negatives of the types of techniques used for defect prevention.
- Alternative solutions to defect prevention to avoid defects in the final project.
- Cost and benefit analysis of the defect prevention techniques.
- Recommendation with motivation why a specific company/project shall use the technique

In the document, we make an attempt to provide proper motivated answers to the above questions and also provide our implication regarding defect prevention.

The document is structured in such a way that each of the further section holds the answer for one of the questions stated above. The final section holds the conclusion of the report.

II. Defect Prevention

This section speaks about the definition of defect prevention based on our understanding and experience. We give a brief explanation about defect perfection with the help of literature.

We comprehend defect prevention is a method of preventing or avoiding defects during developing a software in order to produce a quality product. Defect prevention is a technique by which developers try to eliminate possible defects that could arise in the future. However, no matter what, some or the other defects will always pop up in real time. The developers try to counter attack as many defects as possible before releasing the product.

Defect, in the context of software engineering is the errors or shortcomings in the software development lifecycle that cause the software product to fail or not function properly[1]. In another article, author Nancy S. Eickellmann stated that defect prevention is “an activity of continuous institutionalized learning during which common causes of errors in work products are systematically identified and process changes eliminating those causes are made.[2]” Thus, from the above definitions we comprehend that defect prevention is a process employed throughout the software development life cycle (SDLC) to improve the software development process and in turn produce a quality product. There are a number of techniques engaged in defect prevention of which few of the techniques like Joint Application Design (JAD), prototyping and reusable designs etc [3]. The advantages and disadvantages pertaining to the techniques shall be discussed in the next section.

III. Types of Defect Prevention Techniques- Pros and Cons

Like mentioned in the previous section, there are several defect prevention techniques currently employed. Among the several methods available, Joint Application Design (JAD), prototyping, structured tools, CASE tools etc., have gained prominence.[3] Hence, it is impossible to discuss about all the methods here, thus we provide the positives and negatives of few of the above mentioned techniques.

Here, we discuss the advantages and disadvantages of three important and techniques among the many methods available.

Joint Application Design (JAD): JAD is an approach where users and analysts observe equal responsibility for requirement definition in developing a software system [4]. Studies have shown that use of JAD has helped in improving software defect prevention[5].

Failure Modes and Effective Analysis (FMEA): Failure modes and effective analysis is a systematic procedure for defect prevention which is utilized in the early stages of the design phase.”FMEA is an engineering technique used to define,identify, and eliminate known and/or known potential problems,errors , and so on from the system, design,process, and/or service before they reach the customer”[6].It also helps in prioritizing the identified potential defects based on the serious impact,likelihood of reccurance.It provides the available preventive actions for the identified defects and are used to mitigate the defects earlier in the design phase.

Fault Tree Analysis (FTA) :Fault Tree Analysis is a Boolean rationale used to depict all the combinations of the events that may lead to the failure of the software product.A single defect is taken and analyzed with all the possible causes for that failure. FTA finds the chances to recognize the causes for the specific failure and tries to prevent those causes from reccrance. [7]

Table 1: Pros and Cons of Defect Prevention Techniques

Sl. No	**Technique**	**Advantages**	**Disadvantages**
1.	Joint Application Design (JAD)	-Knowledge sharing and understanding among different people helps in mitigating the common errors.[8] -Active participation of the stakeholders helps in agreeing for reasonable requirements instead of going for an over-ambitious requirements. [4,5]	-Wrong issues may arise due to the absence of homework of the members. [4] -Wrong individuals may be invited by the JAD coordinator because of poor selection criteria. -Partial involvement of the members may take place if the coordinator does not give a careful consideration. The result is: assessments, thoughts and opinions of specific individuals are incorporated.
2.	Failure Modes and Effective Analysis (FMEA)	-Identify and evacuate the failure modes in ahead of the development phase. -Diminish the recurrence of the same sort of failure in the future.[6]	-Failures experienced in previous projects can have an impact in exploring new defects. -Concentrate on significant failure mode. -Owing to the top-down method of FMEA, it might be difficult to identify the significant failures.[6]
3.	Fault Tree Analysis (FTA)	-All the possible combinations for a particular failure are identified. -Analyses Significant and multiple defect modes can be found.[7]	-Only a single defect can be considered in one instance. -To cover a complete rundown of defects, various numbers of FTA’s are required. -Time taking and resource intensive. [7]

IV. ALTERNATIVE SOLUTIONS FOR DEFECT PREVENTION

The defect prevention framework gathers the defect data and performs case analysis for the identified defects, focusing mainly on the balancing activities by providing a clear understanding of how to overcome the future defects and prevent them. These days,Organizations are mainly concentrating on the defect prevention to reduce costs by putting effort on quality assurance in the software product. Together with the defect prevention techniques as mentioned in the previous section, there are few other alternative procedures which can also be used as a defect prevention approaches. Some alternative solutions for the defect prevention are:

- *Prototyping*
- *Root Cause Analysis and Primitive Measures Determination*
- *Periodic Review and Defective Analysis*

Prototyping: The Prototyping process can be used for preventing defects in technical projects where the team is an ambitious on the system's behavior. The process is interactive starting with the analysis phase where the basic requirements are churned into functional requirements, giving a clear view of the product to be developed or designed. This enables the team to change the requirements which would match the customer's needs precisely in the initial stages of development. This helps in defect prevention in the later stages of development of filtering the precise requirements. [3]

Root Cause Analysis and Primitive Measures Determination: Root Cause Analysis commences with the defect tracking. A document is maintained where all the defects detected in the development process are saved. The document contains the log of the defects and their detailed description. The information in this document is analyzed to resolve their cause. This process invokes Root cause analysis where the defects are classified in a Pareto chart. Root cause analysis helps in figuring the root cause of the defect and a plan is made to eradicate those causes. A Fishbone diagram is made in this process for the defect categories and their cause. This can be used for analysis and brainstorming. [9]

Periodic Review and Defective Analysis: A Periodic review is conducted to review the defect prevention plans and sets priorities for them based on the root cause and other factors causing the defect. The defects which are detected during the periodic review will be logged into a document called defects register. This contains detailed information about the defect, its severity and the phase where it occurred. At the end of each developmental phase the defect registered is strengthened and a cause analysis report is made. This report identifies the problem area, giving the developer a scope to improve it. The plan is shared with the team so that a similar problem will not occur in the future. [10]

V. COST BENEFIT ANALYSIS

The cost benefit analysis of the above mentioned techniques is addressed based on return of investment, effort involved and the effectiveness of technique.

JAD: JAD involves both users and analysts. There is a lot of human resources involved in this approach of defect prevention. Testers, Programmers, Analysts are involved in JAD, the investment rate might increase based on the type of the project. Effort in JAD process varies based on the experience of the personnel involved. Experienced professionals find it easier when compared to non-experienced professionals, which in turn leads to increase in effort. JAD is useful in case of projects where requirements are weakly defined. [11]

FMEA: FMEA is a defect prevention tool used in the early stages of the software development. This helps in identifying results at an early stage in the project which leads to less development costs in the long run. Thus, this method would lead to a high return on investment. The effort involved in finding different failure modes is high as it needs some considerable amount of effort to identify them. [11]

FTA: FTA is a technique where the root cause of the technique is primarily identified. From the identified defect the other causes for this defect are identified. This method involves one defect at a time. Hence, it takes a lot of time and effort to implement this method. A lot of time and effort implies high development costs and a possible chance of low return on investment. [11]

VI. MOTIVATION FOR USING THE TECHNIQUE

Defect prevention is a key aspect for the software quality assurance of a product. This should be examined in every phase of the development and the defects should be identified and resolved using a plan. The following are the different techniques to be used in different phases of the development.

Requirement Phase: for the requirement phase JAD is recommended. In JAD the defect information is gathered from all the members and the final requirements are approved by all the members. This gives a chance to share the opinions and thoughts of all the members and thus decreasing the chance of misleading requirements. This guarantees clear requirements and hence the quality of the product is benefited. Also in the JAD the requirements are gathered from all the members so it accelerates the development process and thus would increase the quality of the product. So, JAD is highly recommended in the requirement phase of development. [4,8]

Design Phase: FMEA (Failure mode and effects analysis) is the best recommended in the design phase of a software development process. FMEA is used to detect the defects in the initial stages where it is less complex and easy to prevent. This increases the software quality greatly. Though there are other techniques available for defect prevention in the design phase FMEA is recommended above them because it reduces risk at the budding stages and reducing the delayed charges for the defect prevention. [11]

Implementation Phase: Test driven development is recommended in the Implementation phase. In TD automated test cases are framed which test the newly implemented functionalities before the actual implementation. This increases the efficiency in testing and thus increases the quality of the product. When a new code is added, TDD is used to identify the defects in it by using the automated test cases. Hence TDD is the efficient technique for defect prevention in the implementation phase. [11]

Testing Phase: This is the last phase in the development process. This is the penultimate stage before the product is deployed. All the remaining defects should be prevented and reduced in this stage. Automated testing tools are recommended for this. The testing tools test the functionalities and thus assuring that the product is quality personified. Testing greatly increases the quality of the product by reducing the defects in the product's untested functionalities. [11]

VII. References

[1] S. Kumaresh and R. Baskaran, “Defect analysis and prevention for software process quality improvement,” *Int. J. Comput. Appl.*, vol. 8, no. 7, 2010.

[2] N. S. Eickelmann, “Empirical studies to identify defect prevention opportunities using process simulation technologies,” in *Software Engineering Workshop, 2001. Proceedings. 26th Annual NASA Goddard*, 2001, pp. 22–25.

[3] C. C. Jones, *Software quality: Analysis and guidelines for success*. Thomson Learning, 1997.

[4] R. B. Jackson and D. W. Embley, “Using joint application design to develop readable formal specifications,” *Inf. Softw. Technol.*, vol. 38, no. 10, pp. 615–31, Oct. 1996.

[5] E. J. Davidson, “Joint application design (JAD) in practice,” *J. Syst. Softw.*, vol. 45, no. 3, pp. 215–23, Mar. 1999.

[6] D. H. Stamatis, *Failure Mode and Effect Analysis: FMEA from Theory to Execution*. ASQ Quality Press, 2003.

[7] M. Towhidnejad, D. R. Wallace, and A. M. Gallo, “Fault tree analysis for software design,” in *27th Annual NASA Goddard/IEEE Software Engineering Workshop, 2002. Proceedings*, 2002, pp. 24–29.

[8] “Defect Prevention Techniques and its Usage in Requirements Gathering - Industry Practices.” [Online]. Available: http://www.academia.edu/4713822/Defect_Prevention_Techniques_and_its_Usage_in_Requirements_Gathering_-_Industry_Practices. [Accessed: 07-Jan-2015].

[9] M. Faizan, M. N. A. Khan, and S. Ulhaq, “Contemporary trends in defect prevention: A survey report,” *Int. J. Mod. Educ. Comput. Sci. IJMECS*, vol. 4, no. 3, p. 14, 2012.

[10] R. G. Mays, C. L. Jones, G. J. Holloway, and D. P. Studinski, “Experiences with defect prevention,” *IBM Syst. J.*, vol. 29, no. 1, pp. 4–32, 1990.

[11] “Study On Defect Prevention Information Technology Essay.” [Online]. Available: http://www.ukessays.com/essays/information-technology/study-on-defect-prevention-information-technology-essay.php. [Accessed: 07-Jan-2015].