Applying FMEA to Assess the Probable Failure Modes Criticality at the Planning Phase of the Software Development Life Cycle (SDLC)

[1]Bashiru Lawal, [2]Olowonusi Victor Oluwaseyi, [3]Adamu Mohammad A. & [4]Mohamed M. Elzahaf
[1,2]Department of Computer Science, Federal College of Education (Technical), Gusau-Nigeria.
[3]National Research Institute for Chemical Technology – Basawa, Zaria – Nigeria
[4]Department of Computer Science, Faculty of Medical Technology, Libya – Derna Shiha Sharakia.

Received May 09, 2017 Accepted May 29, 2017

ABSTRACT *The Failure Mode Effect and Analysis (FMEA) is a safety engineering technique aimed at identifying and classifying potential failure modes, their effects on the system and defining actions to avoid these failures. Increasingly, this technique is being adapted to modeling software systems for improving reliability. The application of FMEA for risk management process in the Software Development Life Cycle (SDLC) have been studied extensively by researchers during the last decade and have been reported to contribute meaningfully to minimizing the incidence of software failure and improve the quality of software product. In this study, we make a case for the adoption of FMEA in software engineering. Furthermore, we apply the methodology to identify, analyse and evaluate the risk factors associated with the probable failure modes that exist in the planning phase of software development project of any kind. We later propose a formalized approach to determine the level of criticality of the established failure modes in the planning phase of the SDLC.*

Key words: FMEA; Risk Management; Failure mode; Software Development Process/SDLC; Planning Phase

INTRODUCTION

The FMEA is a well-known method to find potential failures in a proposed system design or process before they occur. Its application focuses primarily on individual failure mode characteristics while its major benefits centralizes on the identification of system failure modes so as a mitigation action can be designed before the failure occur. The method was introduced in the late 1940's for military usage by the US army and later found big usage in the aerospace and rocket development to avoid errors and following fatal outcomes [7].

The FMEA application had been effective in failure prevention in other discipline such as: industrial processes [3]; real time embedded projects [4]; general engineering [5] etc. Over the years, it has becomes more and more acceptable and is used in different big industrial companies. Although, it is widely popular and widely used, there is relatively little information published on the use of FMEA for information systems [6] and even less published about the usage in the software development process [7] and [8].

The application of FMEA for risk management process in the software development projects have been studied extensively by researchers during the last decade and have been reported to contribute meaningfully to minimizing the incidence of software failure (e.g. [18]; [9]; [10]). Prominent of which are the software risk assessment at architectural level using UML diagrams by [18] and [8], which had been proven to be very effective. There are other examples in other areas which apply FMEA in the software development over concrete programs: [6] and [12] and are very practically oriented.

[7], proposes a guide for the conduct of FMEA over the software development process. [14], highlights the role of FMEA in developing quality software. He discusses the early identification of potential software failure modes as a critical and excellent practice in software development process because it gives the possibility to verify the presence of correct code. However, methodological applications of the existing risk management models have not been effectively integrated into the software development process and consequently failed to adequately address the risk factors that are associated with the failure modes, which can minimize the incidence of failure in software development projects. The failure mode is the specific manner by which a failure is observed in terms of failure of the item in a process (or sub-process) under investigation at each stage of the software building process; and generally describes the way the failure occurs.

Potential failure modes in SDLC should be described in physical or technical terms, not as a symptom noticeable by the customer. For example, typical failure modes in the design phase of the SDLC could be: (i) the design fails to meet the low-level concepts where the high coupling and low cohesion make the design incomprehensible and not easily maintainable and (ii) the components in the design are not correctly combined, not well documented and there are components that affect other ones in unexpected way, are each a separate probable failure mode that can exist in the design stage of any (SDLC).

Also, Framework to address such problems has not been explicitly formulated by researchers. And the underlying dimensions of the failure mode criticality and their influence on software development projects remain largely unexplored. Because of this, the proposed study is expected to put a new milestone in the use of FMEA in the software industry by identifying the potential failure modes in the software development process. The emphasis is on the planning phase of the SDLC. We examined the planning phase of SDLC, which has the main concepts as: schedule estimation, risk management, project planning, team organization and costs planning. The presentation will also synthesize a formalized approach to classify the criticality of each of the established failure mode.The outcome of this work is a single FMEA document containing a list of potential failure modes prioritized by their associated risk in the planning stage of the software development process. The prioritized list identifies the potentially high-threat areas to focus extra developmental resources to produce more reliable system.

RELATED WORK

Over the years, most programs and organizations use risk management when developing and operating software-reliant systems, preventable failures continue to occur at an alarming rate. In many instances, the root causes of these preventable failures can be traced to weaknesses in the risk management practices employed by those programs and organizations [1]. There have been many approaches proposed or developed to help improve existing risk management practices by researchers and scholars, mostly focused on the application safety and reliability engineering tools for software risk management. For example, [5], have proposed a risk assessment method by considering possible failure modes of a scenario and computing the complexity of the scenario in each failure modes. Their proposal method is for risk assessment at the requirement level. So, the risk associated with low level details such as component or connector is not considered. [18] propose a reliability-based risk assessment methodology at the early stage of SDLC to estimate the risk the risk factors for various state of a component within a scenario. Complexity and severity were the key data used to quantify the associated risk of component/connector and scenario within the system. Interestingly, the assessment integrated the analytical and subjective views and considers both the technical and some non-technical (business requirement factors) risks. The weakness of the assessment was based on fictitious assumption of fixed requirements and never addressed the situation of any conflicting opinions of the assessment team. The assessment also considers only the early stage of the SDLC of software based system.[7],put a new milestone in the use of FMEA on the software industry. Georgieva proposes a methodology for applying FMEA during the Software Development Process with the aim of analyzing the failure modes in the SDLC. The work exhaustively describes steps to follow in order to conduct an effective application of FMEA over the SDLC. About 52 probable failure modes were identified from the 8 stages that are common to most software development process.

Though, a comprehensive list of failure modes, which is common to all software development process in each phase of SDLC, was generated, details about their associated risk factors were not considered. Neither any of the risk associated with the failure mode was estimated nor was information about the criticality and importance of individual failure mode in relation to different phases and stages within the SDLC determined. However, this is highly necessary in order to guide the developers and testers in effective decision making for allocating development resources to high level requirements.

There are other few examples which apply FMEA in the software development over concrete programs: ([6] and [12]) and are very practically oriented. [14], highlights the role of FMEA in developing quality software. Luke discusses the early identification of potential software failure modes as a critical and excellent practice in software development process because it gives the possibility to verify the presence of correct code. However, methodological applications of the failure method to assess the risk factors that are associated with the failure modes across the software development process have not been explicitly formulated. And the underlying dimensions of the failure mode criticality and their influence on software project development remain largely unexplored.

The Technical Benefit of FMEA Application in Software Engineering

It has been a productive, resourceful and successful reliability based techniques in software risk assessment methodologies. There was no case reported from literature where FMEA application in software was unsuccessful. This is evidently shown from the large number of articles returned after a search from the libraries of IEEE Explore, ACM and Computer Reviews.

Although FMEA was not originally created for software systems, we can translate the principles over to software engineering and take advantage of the many benefits that the methodology has to offer. Major benefits derived from a properly implemented FMEA effort in software engineering are as follows:

1. Unlike other traditional software risk assessment techniques such as testing, which focuses on scenario success, FMEA focuses on what can go wrong and how
2. Makes requirement, design and code reviews more effective
3. Identify abnormal behaviour that might be missing from the requirement or design specifications

4. FMEA is often targeted on potentially high risk area
5. A documented uniform method of assessing potential failure mechanisms, failure modes and their impact on software operation, resulting in a list of failure modes ranked according to the seriousness of their system impact and likelihood of occurrence.
6. Early identification of single failure points (SFPS) and system interface problems, which may be critical to mission success and/or safety. They also provide a method of verifying that switching between redundant elements is not jeopardized by postulated single failures.
7. An effective method for evaluating the effect of proposed changes to the design and/or operational procedures on software development project success and safety.
8. It provides a documented method for selecting a design with a high probability of successful operation and safety.
9. Criteria for early planning of tests.
10. Address one failure mode could mean eliminating several failures

From the above list, early identifications of SFPS, which cannot be addressed by redundancy or other hardware controls, input to the troubleshooting procedure and locating of performance monitoring / fault detection devices are probably the most important benefits of the FMEA. Inaddition, the FMECA procedures are straightforward and allow orderly evaluation of the design.

PROPOSED APPROACH

Our proposed approach is based on the general steps of conducting effective FMEA using a constructive approach to examine potential problems in every activity that must be carried out during the planning phase of any software development project. This approach can be generally summarized into four major parts, which are shown on the Figure 1: The FMEA methodology. However, this general approach was slightly modified and used to investigate the target phase (planning phase) of any software development process using the modified approach steps listed below:

- Step 1. Gather a team and review the activities of the planning phase, which are peculiar to software development projects
- Step 2. Brainstorm on potential failure modes in the planning phase of the software development process.
- Step 3. Assign different effects caused by the failures.
- Step 4. Prioritize – assign severity, occurrence and detection rankings for each failure mode.
- Step 5. Calculate the RPN number.
- Step 6. Collect data, analyze and classify the level of criticality of the individual failure mode

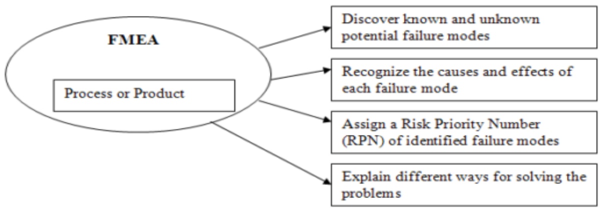
FMEA
Process or Product
Discover known and unknown potential failure modes
Recognize the causes and effects of each failure mode
Assign a Risk Priority Number (RPN) of identified failure modes
Explain different ways for solving the problems

Fig. 1: FMEA Methodology

3.1 Conducting the FMEA Process

To be able to conduct the FMEA methodology, a team of software development experts who have participated in different kinds of software development projects were assembled from different software industries to participate in this research. A totally of four (4) software organizations from Nigeria and which are selected based on criteria that will categorize them into certain types and sizes of businesses, and certain types of software projects form the Population of the study. The interviewees representing the organizations to be studied, in turn, are elected based on several criteria including minimum experience of the software development and/or risk management processes and role. The minimum requirements are that they were or had been involved in at least one developed project using any development model such as: waterfall, agile or iterative (RUP) and risk management processes in their organizations. They also represent different roles, which are expected to yield valuable data representing complementary perspectives of the development process (management, product owners, and developers). Note that these criteria were used to target the organizations and to restrict the population to be studied. It is in this population that the sample was found. Eventually, the organizations and individual interviewees that participated in this research are randomly selected based on convenience sampling from this population.

The team engaged in a brainstorming exercise to deeply examine the important activities that are peculiar to the planning phase of software development project of any kind. We haves selected the planning phase of the software development process as the focus area, which we formally describe its central objectives and goals. The Planning Phase focuses principally on required project planning work. Proper comprehensive project planning is essential to a successful software development project, and incomplete project planning and analysis are frequently root causes of project failure. Most project planning is conducted as part of the PMBOK Integration Management work, which includes defining the processes necessary to identify, define, combine, unify, and coordinate all project activities for successful project deployment [19].

THE PLANNING PHASE

The information gathered from the requirement analysis of the intended software project is used in the planning phase to plan the basic project approach and to conduct product feasibility study in the economic, operational, and technical area. All risks that are associated with the intended project are identified and analyzed in this phase. The quality assurance requirement planning is also performed in the planning phase. Below are some of the objectives and goals of the planning phase of any software development project [19]:

Objectives

Successful completion of the Planning Phase should comprise:

• Assessment and description of the procurement management strategy

• Elaboration and refinement of the project scope, schedule, risks, and costs

• Assessment and description of activities to coordinate all relevant subsidiary plans

• Definition of procedures for how the project will be executed, monitored, controlled, and closed

• Planning the future course of action

• Development of the Project Management Plan(s) (PMP)

• Approval to progress to the Requirements Analysis Phase

Goals

The purpose of the Planning Phase is to plan all project processes and activities required to ensure project success and to create a comprehensive set of plans, known as the PMP, to manage the project from this phase until project termination.

In addition to their expertise and experiences the facilitators (authors) supply a list of deliverables that is expected from the planning phase of any software development project. The list contains the following:

Project Management Plan – Deliverables [19]

• Scope Management Plan

• Schedule Management Plan

• Cost Management Plan

• Quality Management Plan

• Staffing Management Plan

• Communication Management Plan

• Risk Management Plan

• Procurement Management Plan

• Change Management Plan

ASSESSING THE FAILURE MODES AT THE PLANNING PHASE

The tean together with the researchers had a focus group discussion to elaborate the activities and deliverables of the planning phase of software development projects. The group discussion focusess on three main artifacts: the documet containing list of project management plan deliverables common to all projects development (software inclusive); specification desribed by the objectives and goals of project planning phase and list of established probable failure modes by [7]. The team braistorm on the three listed artifactes together with their expertise to identify the potential failure mode in the planning phase that are common to software development projects. We assess each of the identified failure mode and

Table 1: Suggested criteria of rating for Severity of a failure in FMEA (adopted from [2])

Rating	Effect	Severity of effect
10	Hazardous without warning	Highest severity ranking of a failure mode, occurring without warning and consequence is hazardous
9	Hazardous with warning	Higher severity ranking of a failure mode, occurring with warning, consequence is hazardous
8	Extreme	Operation of system or product is broken down without compromising safe
7	Major	Operation of system or product may be continued but performance of system or product is affected
6	Significant	Operation of system or product is continued and performance of system or product is degraded
5	Moderate	Performance of system or product is affected seriously and the maintenance is needed
4	Low	Performance of system or product is small affected and the maintenance may not be needed
3	Minor	System performance and satisfaction with minor effect
2	Very minor	System performance and satisfaction with slight effect
1	None	No effect

Table 2: Suggested criteria of rating for Occurrence of a failure in FMEA (adopted from [2])

Rating	Probability of occurrence	Possible failure rate
10	Extremely high: failure almost inevitable	⩾1/2
9	Very high	1/3
8	Repeated failures	1/8
7	High	1/20
6	Moderately high	1/80
5	Moderate	1/400
4	Relatively low	1/2000
3	Low	1/15000
2	Remote	1/150000
1	Nearly impossible	⩽1/1500000

Table 3: Suggested criteria of rating for Detection of failure in FMEA (adopted from [2])

Rating	Detection	Criteria
10	Absolute uncertainty	Potential occurring of failure mode cannot be detected in concept, design and process FMEA/mechanism and subsequent failure mode
9	Very remote	The possibility of detecting the potential occurring of failure mode is very remote/mechanism and subsequent failure mode
8	Remote	The possibility of detecting the potential occurring of failure mode is remote/mechanism and subsequent failure mode
7	Very low	The possibility of detecting the potential occurring of failure mode is very low/mechanism and subsequent failure mode
6	Low	The possibility of detecting the potential occurring of failure mode is low/mechanism and subsequent failure mode
5	Moderate	The possibility of detecting the potential occurring of failure mode is moderate/mechanism and subsequent failure mode
4	Moderately high	The possibility of detecting the potential occurring of failure mode is moderately high/mechanism and subsequent failure mode
3	High	The possibility of detecting the potential occurring of failure mode is high/mechanism and subsequent failure mode
2	Very high	The possibility of detecting the potential occurring of failure mode is very high/mechanism and subsequent failure mode
1	Almost certain	The potential occurring of failure mode will be detect/mechanism and subsequent failure mode

assigned different effects caused by the failures. We then prioritize the established failure mode by assigning Severity (S), Occurrence (O) and Detection (D) rankings for each of the established failure mode using the Tables 1, 2 and 3 respectively. The Risk Priority Number (RPN), which is a mathematical product of the Severity, the failure modes Occurrence and the Detection, was later calculated using the formula:

RPN = Severity * Occurrence * Detection

The Severity is a rating corresponding to the seriousness of an effect of a potential failure mode. Severity or seriousness of the risk is considered just in case of "the effect"; reducing the risk severity is possible only through changing the process and the manner of performing activities [17].

Also, the Occurrence will be ranked according to the failure probability, which represents the relative number of failures anticipated during the design life of the item. The effects of a failure mode are normally described by the effects on the user of the product or as they would be seen by the user [17]. Similarly, Detection determination is an assessment of the ability existing to identify a cause/mechanism of a risk occurrence. In other words, detection possibility is a rating corresponding to the likelihood that the detection methods or current controls will detect the potential failure mode before the product is released for production for design, or for process before it leaves the development facility.

The RPN Methodology

The probable failure modes that can exist in the planning phase of software development process as observed by the team of software development experts are listed in the potential mode column in Table 4. The potential failure modes were established after a critical examination of the planning specifications and deliverables contained in the provided artifacts. After the establishment of the probable failure modes in the planning phase, we rank the individual probable failure mode by S,D and O using the Tables 1,2 and 3 as guide. There were no conflict of opinions from the team during the ranking and the resulting indexes are provided in the column RPN parameter (i.e S, O and D) in Table 4.

The RPN methodology was implemented using the equation (RPN = S*O*D) for each failure mode to quantify and prioritize the critical value of the failure mode. The higher the RPN value the higher the Critical RPN [17].

CLASSIFICATION OF FAILURE MODE CRITICALITY

In determining the criteria for the critical level classification, our attentions were not only focus towards RPN values alone but also each of the three failure factors (S, O and D) would be evaluated. For this purpose, a criterion would be defined as the level of crisis. The crisis is a criterion that expresses the importance of a potential/actual risk of the failure mode in the planning phase of the SDLC.

In addition, it would be used to measure the level of crisis in the phase level studied. Crisis grade would be classified as: normal, semi-critical, and critical levels, which methods for considerations are explained in detail below (adopted from [16]):

- Level 1: Normal level in which all of the three factors of RPN (especially the severity and probability of occurrence) have values less than 5. Or RPN number is very low and does not require corrective and preventive actions (usually RPN < 70)).
- Level 2: Semi-critical level in which at least a factor of three factors of RPN (especially the severity and probability of occurrence) has a value greater than 5 but RPN is relatively low. In this case, corrective/preventive action is essential (typically 70 < RPN < 140).
- Level 3: Critical level, in which at least two factors of the three-factors of RPN (i.e. S, D and O) have high values or RPN value is too high. Since this level has been considered for high RPN,

it is obvious and clear that it has a corrective/preventative action (usually RPN > 140). The detailed analysis of the classification of the critical level probable failure mode in the planning phase of software development project of any kind is presented as shown in Table 4 below.

Findings and Discussion

Table 4 shows all the pertinent failure modes identified and the detail analysis of their critical level classification at the planning phase of the SDLC. The table revealed that for most software development project of any kind, the area with the highest-threat during the planning phase is failure mode with serial number 9: **if d*uring the development process the schedule made at the beginning fails to determine the needed time for all activities from the beginning to the end and also the cost seems inaccurate.***The lesson to be learnt here is that softwaredevelopment team should pay much more attention to this failure mode and make accurate decision concerning this task.

Table 4: Detail analysis of the critical level classification of failure mode in requirement phase

SDLC Phase	Potential Failure Mode	S	O	D	RPN	Critical Level
	(1) The chosen software development model is not appropriate to the scope, magnitude and the complexity of the defined task	7	6	3	126	Critical
Planning Phase	**(2)** The chosen model does not fit the complexity of requirements for the project at the very beginning.	7	5	5	175	Critical
	(3) The documentation and results obtained by the chosen software model does not reach the concerned management, development or maintenance team	2	2	3	12	Normal
	(4) Specifications for a certain component of the software project are not clearly defined or not correctly documented	7	7	6	294	Critical
	(5) The information and documentation about the selected standards, methods, tools and programming language is not enough to have the required tasks completed	6	6	5	180	Critical
	(6) The required assignments are not achievable because of the limitations of the chosen framework or					

	program tools	4	3	3	36	Normal
Planning Phase	**(7)** There are "wholes" in the system which makes it vulnerable to external attacks	8	5	6	240	Critical
	(8) There is no clearly defined hierarchy model for the different type of people using the system	3	4	1	12	Normal
	(9) During the development process the schedule made at the beginning fails to determine the needed time for all activities from the beginning to the end and also the cost seems inaccurate	**9**	**9**	**8**	**648**	**Critical**
	(10) The project fails because of not enough experienced engineers and insufficient abilities of the team members to cope with the different problems	8	8	7	448	Critical
	(11) There is no clearly defined hierarchy model for the different type of people using the system	5	4	2	22	Normal
	(12) Cost estimation of the projects is not correctively	9	7	5	315	Critical

Conclusion

In this paper we propose a novel methodology of applying FMEA to assess the risk associated with the probable failure modes across the phases of SDLC. Although the area of study is the planning phase of the SDLC, similar approach and process can be conducted across the remaining phases of the SDLC. We also proposed a formalized approach to determine the critical level of probable failure mode that can exist in all phases of the SDLC.

We demonstrated the applicability of the approach further, using the studied phase (Planning) of the SDLC. We synthesized the planning phase requirement with information contained in thethree presented artifacts: the documet containing list of project management plan deliverables common to all projects development (software inclusive); specification desribed by the objectives and goals of project planning phase and list of established probable failure modes developed by [7]. We establish and assess the risk factors associated with each of the established failure mode with the S, O and D ranking using the FMEA. We later prioritized the criticality of the risk value through the RPN methodology.

Using the planning phase of the SDLC, we show the effectiveness and the viability of the proposed approach. Eight areas needed immediate and corrective actions to make a balance for an effective software development project planning. These eight areas are failure modes with serial numbers: 1, 2, 4, 5, 7, 9, 10 and 12, as shown in Table 4.

REFERENCES

[1] Software Engineering Institute. 2012. Software Risk Management. Technical Report CMU/SEI-12-TR012.June, 2012

[2] Chin K.S., Wang Y. M., Poon G. K. K., Yang J. B. 2009. Failure mode and effects analysis using a group-based evidential reasoning approach. ComputOper Res 2009;36:1768–79.

[3] Ebrahemzadih, M., Halvani, G. H., Behzad, S. and Giahi, O. 2014. Assessment and Risk Management of Potential Hazards by Failure Modes and Effect Analysis (FMEA) Method in Yazd Steel Complex. Open Journal of Safety Science and Technology, 4, 127-135. http://dx.doi.org/10.4236/ojsst.2014.43014

[4] Khaiyum, S. and Kumaraswamy, Y. S. 2014. Integration Of FMEA And FTA For Effective Failure Management In Real Time Embedded Projects. IJBRITISH, Volume 1 2014 Issue 3

[5] Appukkutty, K., Ammar, H. H. and Goseva-Popstojanova, K. 2005. Software Requirement Risk assessment using UML. In the Proceedings of the ACS/IEEE International Conference on Computer Systems and Applications. Pg.112-115.

[6] Lauritsen, T., Stålhane, T., 2005. Safety methods in software process improvement. 12th European conference on Software Process Improvement (pp. 95105). Berlin: Springer-Verlag Berlin.

[7] Georgieva, K. 2010. Conducting FMEA over the Software Development Process. ACM SIGSOFT Software Engineering Notes May 2010 Volume 35 Number 3. Pg 1-3. Available at: http://doi.acm.org/10.1145/10.1145/17648 10.1764819

[8] Goseva-Popstojanova, K., Hassan, A., Guedem, A., Abdelmoez, W., Nassar, D. E. M., Ammar, H., and Mili, A. 2003. Architectural-level Risk Analysis Using UML. IEEE Transactions on Software Engineering, Vol. 29, No. 10, pg. 946-960.

[9] Vucovich, J.P., Stone, R.B., Liu, X. &Tumer, I.Y. 2007. Risk Assessment in Early Software Design Based on the Software Function-Failure Design Method. In COMPSAC '07: Proceedings of the 31st Annual International Computer Software and Applications Conference. IEEE Computer Society.

[10] Hassan, A., Goseva-Popstojanova, K., Ammar, H. 2003. Methodology for Architecture Level Hazard Analysis, A Survey. ACS/IEEE Intl. Conference on Computer Systems and Applications (AICCSA 2003), Tunis, Tunisia, Pg. 68 – 70.

[11] Gordon, P. (1999). To err is human, to estimate divine. Information Week, 65 – 72.

[12] Ozarin, N. and Siracusa, M. 2003. A Process for Failure Modes and Effects Analysis of Computer Software, In Proceedings of Annual Reliability and Maintainability Symposium, 365-370.

[13] Banerjee, N. 1995. Utilization of FMEA concept in software lifecycle management. Proceedings of Conference on Software Quality Management.

[14] Luke, S. 1995. Failure mode, effects and criticality analysis (FMECA) for software. 5th Fleet Maintenance Symposium.

[15] Carbone, T.A. and Tippett, D.D. 2004. Project Risk Management Using the Project Risk FMEA. Engineering Management Journal, Vol. 16 No. 4, p. 28-35

[16] USA Department of Defence. 1980. Military Standard, Procedure for Performing a Failure Mode, Effects and Criticality Analysis. Department of Defence, Washington DC, 27-33.

[17] Narayanagounder, S. and Gurusami, K. 2009. A New Approach for Prioritization of Failure Modes in Design FMEA Using ANOVA. World Academy of Science, Engineering and Technology, 3, 477-484.

[18] Mitrabinda, R. and Durga Prasad, M. 2011. A Novel Methodology for Software Risk Assessment at Architectural Level Using UML Diagrams. SETLabs Briefings, 9(4), pp 41 -60

[19] Project Management Institute (PMI). 2004. PMBOK: A Guide to the Project Management Body of Knowledge, fourth edition.