SIGNIFICANCE OF FMEA TO IMPROVE RELIABILITY OF SOFTWARE PROJECTS

SamithaKhaiyum
Assistant Professor and Research Scholar,
Department of MCA (VTU),
DSCE, Bangalore-78
Samitha.athif@gmail.com

Dr. Y S Kumaraswamy
Professor and Head,
Department of MCA (VTU),
DSCE, Bangalore-78
Yskldswamy2@yahoo.com

Dr. K Karibasappa
Professor and Vice Principal,
DSCE, Bangalore-78
karibasappably@gmail.com

Abstract

Failures are of high probability among majority of software projects. Both technical and social issues contribute to the software project failures. This paper brings out the significance of failure avoidance and also discusses the various system reliability tools like FTA, ETA, RBD and FMEA. To use these tools to accurately determine failure mechanism, one needs to be sound intellectually and understand the basic failure modes of the operation. Therefore the reliabilities or effectiveness of these tools depends on accurate human reasoning whether the operation is carried out manually or through the use of software which is prone to integrity issues. It also describes the significance of FMEA as a reliability tool and emphasises the techniques in which it is carried out. This tool is used to visualise and mitigate problems.

1. Introduction

All Software projects consist of many modules or components. These projects have to be carefully considered on account of avoiding failures. There are multiple failure modes which have to be considered. However, the analysis is complicated. The analysis has to mainly focus on the contribution of component failure that would eventually lead to system failure, the redundancy of the system, the effects of failure and how the rest of the system would react at a failure, the statistical correlation between failures along with the progressive failure of components. Thus safety analysis is necessary as it provides a framework for decision making by analysing all possible failure aspects.

2. Literature Survey

Authors in [1] and [2] bring out the importance of software but also state that software is one of the major problem areas faced by large corporations.
Authors in [14] described Risk based decision making is a process that organizes information about the possibility for one or more unwanted outcomes to occur into a broad, orderly structure that helps decision makers make more informed management choices.
Authors of [3] & [4] stated that on interviewing software project managers (PMs), they revealed the three major complaints against software projects which undoubtedly are real and serious. However, they also state that the corporate executives also contribute to software problems. The following are three complaints against top executives:

1. Executives often reject accurate and conservative estimates.
2. Executives apply harmful schedule pressure that damages quality.
3. Executives add major new requirements in mid-development.

Authors in [5, 6] state that software quality control is the only distinguishing factor between successful projects and challenged or failed projects. They also add that finding and fixing these bugs is most expensive. Their stud also proves that software projects with good quality control cost less and have shorter schedules than projects with poor quality control.
However, authors in [7, 8] state that the reason for successful projects to have such a high defect removal efficiency when compared to unsuccessful projects is the use of design and code inspections.

3. Root Causes

The root causes of software risk factors can be summarised as:

1. Inaccurate estimation and schedule planning.
2. Incorrect status reports.
3. Unrealistic schedule pressures.
4. Addition of new or changing requirements during development.
5. Inadequate quality control.

4. Case study

This case study includes study made on software industries of various production capabilities including both service based and product based industries. However, this research restricted towards companies which are certified by either CMMI level 4 or CMMI Level 5 standards. Analysis is carried out to find out the practical cost and time overruns that take place in software development projects.
Table 1 shows a sample of real time embedded projects analysed on the basis of time and table 2 on the basis of cost.

Table 1: Expected vs actual time for project completion.

PF	TIME	
Project	EXP	ACT
P1	320	370
P2	350	550
P3	450	550
P4	1550	2225
P5	4575	6345
P6	1500	1750
P7	1500	1640
P8	2500	3250
P9	4750	6500
P10	8200	9500

PF- Project Factor, EXP- Expected time of completion measured in person months, ACT- Actual time of completion measured in person months.

From the table it is seen that Project P1 has a time slippage of 15%, P2 of 57%, P3 of 22%, P4 of 44% and P5 of 39%. Project P6 shows a delay of 17%, whereas P7 shows a delay of 9%, P8 a delay of 30%, P9 of 37% and Project P10 a delay of 16%.

Figure 1 shows the graphical representation of time overrun in the projects that have been considered.

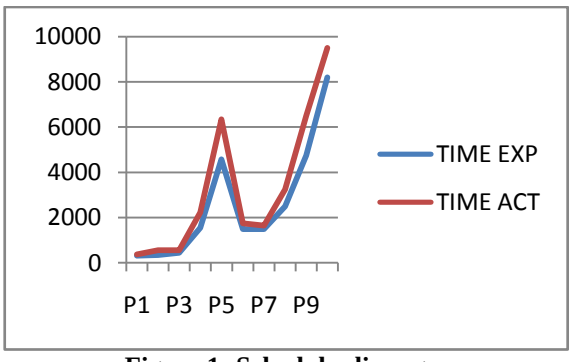

Figure 1: Schedule slippage

Table 2: Expected vs Actual cost for project completion

PF	COST	
Project	EXP	ACT
P1	2.75	3.5
P2	3.5	5.2
P3	4.5	5.75
P4	7.5	10.75
P5	5.25	6.75
P6	1.5	1.75
P7	14.5	15.75
P8	2.75	3.5
P9	4.25	5.75
P10	8.5	9.5

PF- Project Factor, EXP- Expected time of completion measured in person months, ACT- Actual time of completion measured in person months.

From the table it is seen that Project P1 has a cost overrun of 27%, P2 of 49%, P3 of 28%, P4 of 43% and P5 of 29%. Project P6 shows an extra cost of 17%, whereas P7 shows an extra cost of 9%, P8 a cost overrun of 27%, P9 of 35% and Project P10 an extra of 12% cost. This is represented graphically in figure 2.

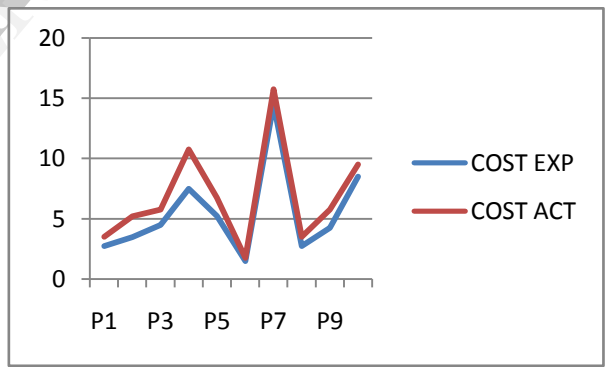

Figure 2: Cost overrun

Table 1 and Table 2 infers that practically any software project face a cost and time overrun, however ideally it is expected to maintain accuracy as planned. It is also found that the most common reason for schedule slippage and cost overrun is that it contains many defects at every stage of system development lifecycle that are not addressed to in a proper manner at the right time.
Since both project managers and corporate executives find assessing root causes of failure a high risk, study was done to find out what were the major causes for estimating problems. Few project managers and executives of large multinational companies of both service and product based development were interviewed who on an average stated that root causes were mainly based on incorrect estimation and schedule planning.

The major root causes for estimating problems:

1. Even before defining the requirements completely, formal estimates are demanded.
2. Previous data is usually unavailable to calibrate estimates.
3. When new requirements are added, the original estimate should not be changed.
4. Lack of usage of modern estimation tools.
5. The conservative estimates should be replaced by more aggressive estimates that are built on business needs rather than on the capabilities of the team to deliver.

5. System Reliability Analysis

The system reliability analysis is modelled with probabilities of events by considering logical combinations and different outcomes from these random events.
The two main methods used for the reliability analysis is the Fault tree analysis and the Event tree analysis. However the other qualitative graphical methods include the Failure Modes and Effects Analysis (FMEA), Reliability Block Diagrams (RBD).

Fault Tree Analysis (FTA): Fault trees are one of the most widely used methods in system reliability and failure probability analysis. A Fault Tree is a graphical representation of events which is hierarchical in nature and has a tree-like structure. Through various combinations of hardware, software, and human involvement the error failures which could be a risk or which ca lead to a system failure can be found. A deductive analysis using a Fault Tree begins with a system failure, which is displayed at the top of a hierarchical tree. This analysis determines failures which can be further prevented. Fault Trees also help in finding the multiple simultaneous failures or events [13].

Event Tree Analysis (ETA): Event tree analysis is an inductive failure analysis performed to find out the effects of a single failure for the overall system risk or reliability. Event Tree Analysis uses similar logic and mathematics as Fault Tree Analysis, but the approach is different - FTA uses deductive approach which analyses system failure and all the way to its reasons whereas ETA uses the inductive approach which analyses the basic failure up to its effects. It is a visual representation of single failure sequences and its influence on other events along with the whole system [12].

Failure Mode Effect Analysis (FMEA): The FMEA is a design tool used to systematically analyse the component failures and identify its effects on the system operations. The analysis is sometimes characterized as consisting of two sub-analyses, the first being the failure modes and effects analysis (FMEA), and the second, the criticality analysis [11]. FMEAs can be performed at the system, subsystem, assembly, subassembly or part level. It is analysed based on three parameters namely the severity of the failure, probability of occurrence of those failures and the rate of detection of those failures.
Reliability Block Diagrams (RBD): A reliability block diagram (RBD) is a diagrammatic method to show how component reliability contributes to the success or failure of a complex system. RBD is also known as a dependence diagram (DD).A RBD or DD is represented in the form of a series of blocks connected in parallel or series configuration. Each block represents a component of the system with a failure rate. Parallel paths are redundant, meaning that all of the parallel paths must fail for the parallel network to fail. By contrast, any failure along a series path causes the entire series path to fail [9][10]. A RBD may be drawn using switches in place of blocks, where a closed switch represents a working component and an open switch represents a failed component. If a path may be found through the network of switches from beginning to end, the system still works. A RBD can be converted to a success tree by replacing series paths with AND gates and parallel paths with OR gates. A success tree may then be converted to a fault tree by applying de Morgan's theorem.

6. Significance of FMEA

FMEA is a methodology that analyses the situation to identify potential failures that could occur and also develop plans to address them throughout the product and process development cycle.
FMEA helps to:

•Identify the potential failures, their potential causes and the risks that would affect the product or process.
•Develop mitigation plans to reduce the risk of failure.
•Re-evaluate the results of actions on the risks that were found.

A Design FMEA (DFMEA) is performed before the design of the product. A Process FMEA (PFMEA) is performed before the release of the design for the process. It is important that FMEA be performed at the right time to take action against the risk and still implement the changes within the design before its release.

Firstly, to start the FMEA, an inside study of past failures and preparatory documents like the block diagrams, process flow diagrams, parameter diagrams, etc. are required. This helps in understanding the potential causes from interfaces,

design choices, noises and environments, etc. A study of potential causes from past FMEAs is also analysed.

Secondly, the Functions, Failure Modes, Effects of Failure and Severity Rankings are identified and inserted into the spreadsheet. Functions consist of the scope, specifications of the design, Government Regulations, Program-specific Requirements and desired outputs. Failure Modes are Anti-Functions; Effects are the results of failure, where each individual Effect is given a Severity Ranking on a scale of 1-10 where 1 being the least severe and 10 being the most. Actions are required if the severity is high.

Then the causes are selected from the Boundary Diagram, Parameter Diagram, or past failures and the respective column in the spreadsheet is filled. The Occurrence rating is also given at this stage on a scale of 1-10, where 1 having the least frequency of occurrence and 10 having the highest. Actions are then developed to address high risk Severity and Occurrence combinations.

After which detection Controls are built to prevent design flaws and a detection ranking is given where 1 being the ones that are easily detectable and 10 being the most uncertain in terms of detection.

Each failure is then assigned a Risk Priority Number (RPN) for Action follow-up. RPN is calculated as the product of Severity, Occurrence and Detection Rankings for each potential failure / effect, cause and control combination.

Actions indicated from the FMEA analysis are closed monitored through the collection of data and observations after a counter measure has been taken. The purpose of an FMEA is to discover and mitigate risk.

After successful confirmation of risk mitigation actions the RPN values are re-ranked to attain the new RPN. This is then compared with the old RPN and a relative improvement is made to the design or process has been confirmed.

There are a myriad of quality and reliability tools available to corporations worldwide, but the one that shows up consistently in company after company is Failure Mode and Effects Analysis (FMEA).

7. Conclusion

This paper analyses real time embedded software projects of both product and service oriented companies. It discusses the significance of failure avoidance by considering practical issues like cost overruns and schedule slippages that occur during the software development lifecycle. It also analyses the different system reliability tools available and highlights the significance of FMEA in terms of parameters like severity, occurrence and detection.

8. References

[1] Yourdon, Ed. "Death March – The Complete Software Developer's Guide to Surviving "Mission Impossible" Projects." Upper Saddle River, NJ: Prentice Hall, 1997.
[2] Glass, R.L. Software Runaways: Lessons Learned from Massive Software Project Failures. Prentice hall, 1998.
[3] Jones, Capers. Assessment and Control of Software Risks. Prentice Hall PTR, 1994.
[4] Jones, Capers. Patterns of Software System Failure and Success. Boston, MA: International Thomson Computer Press, 1995.
[5] Jones, Capers. Assessment and Control of Software Risks. Prentice Hall PTR, 1994.
[6] Kan, Stephen H. Metrics and Models in Software Quality Engineering. 2nd ed. Boston, MA: Addison-Wesley Professional, 2002.
[7] Radice, Ronald A. High Quality, Low Cost Software Inspections. Andover, MA: Paradoxicon Publishing, 2002.
[8] Wiegers, Karl E. Peer Reviews in Software – A Practical Guide. Boston, MA: Addison Wesley Professional, 2002.
[9] Modarres, Mohammad; Mark Kaminskiy, VasiliyKrivtsov (1999), Reliability Engineering and Risk Analysis. Ney York, NY: Marcel Decker, Inc. p. 198. ISBN 0-8247-2000-8
[10] "Reliability Modeling and Prediction", Electronic Reliability Design Handbook. B. U.S. Department of Defense. 1998. MIL–HDBK–338B.
[11] Project Reliability Group (July 1982). Koch, John E., ed. Jet Propulsion Laboratory Reliability Analysis Handbook. Pasadena, California: Jet Propulsion Laboratory. JPL-D-5703.
[12] Silviantia, Mohd. FarisKhamidi, Kurian V John, "Critical review of a risk assessment method and its applications", 2011 International Conference on Financial Management and Economics, IPEDR vol.11 (2011) © (2011) IACSIT Press, Singapore
[13] Y. Geum, H. Seol, S. Lee, Y. Park, "Application of Fault Tree Analysis to the Service Process: Service Tree Analysis Approach", Journal of Service Management, Vol. 20 No.4, 2009 pp.433-454.
[14] B.Macesker, JJ.Myers, V.H. Guthrie, D.A., Walker, S.G.Schoolcraft, 2004. Quick Reference Guide to Risk Based Decision Making (RBDM): A Step by Step Example of the RBDM Process in the Field. EQE International,Inc., an ABS Group Company Knoxville, Tennesses.