ISSN: 2321-7782 (Online)
Volume 4, Issue 2, February 2016
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

A Study on the Strategic Risk Management in Software Engineering Projects

Biswa Ranjan Mohapatra[1]
Research Scholar,
P.G Department of Business Administration,
Utkal University, Vani Vihar,
Bhubaneswar-751004, Odisha - India

Dr. Jaya Krushna Panda[2]
Research Supervisor,
P.G Department of Business Administration,
Utkal University, Vani Vihar,
Bhubaneswar-751004, Odisha - India

Abstract: Many software engineering projects have very high risks and failure rates during their project life. The projects mainly suffer from risks developed in software development which are mostly caused by operational risks, technological risks, management practices, risks in achieving customer acceptance of the product and business risks. Risk can be reduced with certain workflow steps for the software project. Normally, risk management exist at any point of time when a customer analyzes and attempts to quantify the strategic risks for losses in an investment and then takes those appropriate actions to their investment objectives and risk tolerance. Improper risk management may result in severe penalty for both companies and customers. The aim of this paper is to reduce the strategic risks of software engineering projects if we maintain proper workflow steps in each development stages of software engineering projects which results in assure project success.

Keywords: risk management, software engineering, risks analysis, risks, software development life cycle, software project management.

I. Introduction

Risk is an uncertainty that can have a negative or positive effect on meeting project objectives. Risk management is the process of identifying, analyzing, and controlling risk throughout the life of a project to meet the project objectives [1]. The software engineering processes is defined as the interdisciplinary tasks that are required throughout a software product's life cycle to transform customer needs, requirements, and constraints into a system solution [2]. Managing Risk is having two step process namely, one is identifying risks involved in an investment and other is handling those risks to get the project objectives[3]. Risk factors should be considered contractual risk, technical risks, operational risk, risks due to size and complexity of the product, business risks[4]. The field of software development often encounters many kinds of risks with high failure rate which in turn influence the success of software project [5].

II. Literature Review

Tummala and Leung (1999) has produced a methodology for risk management prevailing risk identification, measurement, assessment, evaluation and risk control and monitoring for application of managing cost risk for an EHV transmission line project. Freimut et al. (2001) has highlighted implementation of software risk management for industrial case study. The results indicated that the risk method is experimental, value addition to the software project, and key concepts are understood and usable in practice. Padayachee (2002) highlighted a field investigation framework for risk management for a particular software development company. It was practically tested for several companies. Flinn and Stoyles (2004) proposed risk management phenomena for building trust and confidence for the Internet users. Huang et al. (2004) has highlighted a risk prioritization method by using analytic hierarchy process for enterprise resource planning implementation and the suggested framework considers both qualitative and quantitative factors. The study develops an integrated framework approach for managing risks in software development of an organization within the Government of Barbados. Baccarini et al. (2004) had acknowledged and concerned IT project risks by empirical research and suggested possible responses without any framework for software risk management. Redzic, et al. (2006) offered Six Sigma DMAIC approach used for software quality improvement of software products. Based on data analysis, experts had decided to implement new technologies (tools, methods, standards, and training) to reach project objective. Khanfar, et al. (2008) highlighted that for the success of software projects is depends on involvement of different control factors and different software risk factors which uses chi-square test to control the risks in a software project. Zheng, et al. (2009) projected an estimation method for software effort based on function points by using the linear relations between function points and software efforts. Hribar, et al (2009) highlighted on software quality ranks (SQR) an important method to manage and improve software quality short development lifecycles. Alshathry and Anicke (2010) projected a regression-based model which allows project managers to estimate the trade-off among quality cost and development time of a software development product. Guoheng, et al. (2010) highlighted an Analytic Hierarchical Process (AHP) phenomena to estimate the relative importance of each functional variable feature on a quality attribute. Bukhari and Arif (2010) emphasized a multi-agent framework to help the quality manager to achieve quality objectives of the software product. Lincke, et al. (2010) formed statistically compared quality models which have previously been derived in empirical studies by applying them to software systems. Thus in this study, a proper workflow steps for software development process are proposed to reduce risks and failure rates in software engineering projects.

III. Level of Risk in Different Stages of Software Engineering Project's Life Cycle

In any software development process, there are certain chances of risk but level of risk may vary at the different stages of software development life cycle. At first level i.e software requirement analysis, chances of risk is very less because it is primary level of software development life cycle. Level of risk in software development stages will increase along with the development stages (design, coding, testing& Integration) of the software (Figure 1). A very high chance of risk in software occurs at the software development. If proper work flow steps are followed during each stages of software development life cycle, then level of risk can be minimized. So software engineer should identify, analyze, plan, check, control for each of the strategic risk associated with the different stages of software development life cycle.

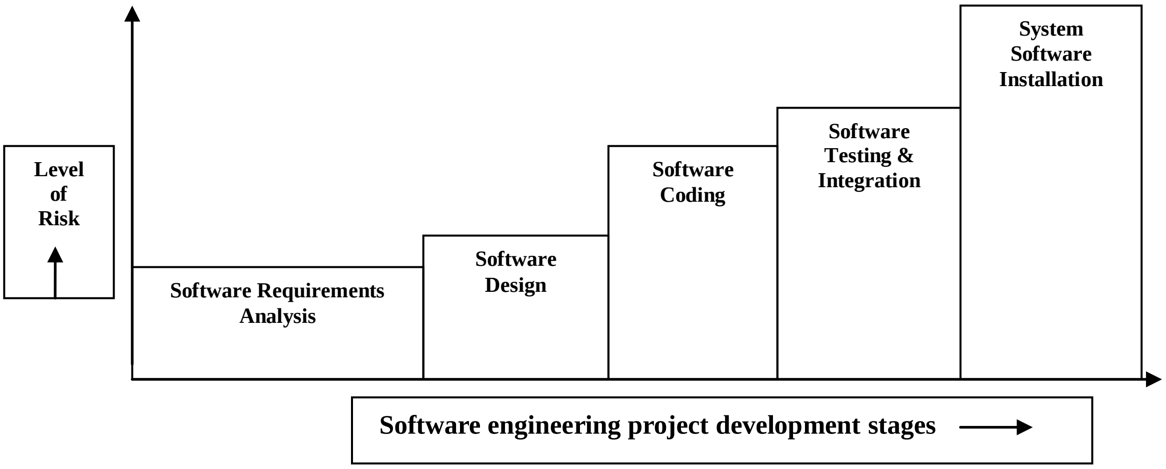
Level of Risk
Software Requirements Analysis
Software Design
Software Coding
Software Testing & Integration
System Software Installation
Software engineering project development stages

Figure 1. Level of risk Vs. Software engineering project development stages

A. Software requirement analysis

When an external customer gives their requirements to project team, the team analyzes software requirements for each software item. Each software item consists of different software components. The software item includes the functional and capability specifications, performance, and external interfaces to the system, qualification requirements, data definition, database requirements, installation and acceptance requirements of the delivered software product, user documentation, user operation and execution requirements. The software requirements shall be evaluated for the traceability to the system requirements and system design, external consistency with system requirements, internal consistency, testability, and feasibility of software design.

The input requirement for software requirement analysis is software architecture, interface design document, system architectural design. Then software developer will develop software requirement analysis (SRA) document for the software engineering project. The software test team and software module developer will do test and validation plan (TVPL) based on software requirement document. Software review team will do software requirements review through minutes of meeting (MOM) which delivers MOM, compliance statement, software change proposal for parent document, base line software requirement analysis document.

B. Software design

Software developer will follow two steps of software design namely preliminary design and detailed design. Software preliminary design step entail transformation of the requirements for the software item into architecture that describe its top-level structure and identifies the software components. It shall be ensured that all the requirements for the software item are allocated to its software components and further refined to facilitate detailed design. A document to describe the top-level design for the interface external to the software item, preliminary test requirements and schedule for software integration shall be generated. The software architecture shall be evaluated for the traceability to the software requirements, external consistency with the software requirements, internal consistency between the software components, appropriateness of design methods and standards used feasibility of detailed design. The projects envisages developing the guidelines for top level software architecture design for structural and object oriented design approach, tailor the template of top level architecture document and provide checklist for top level design review [2].

Software detailed Design involves this phase involves development a detailed design for each software component of the software item. The software components shall be refined into lower levels containing software units that can be coded, compiled and tested. It shall be ensured that all the software requirements are allocated from the software components to software units. A document on detailed design for the interface external to the software item, between the software components and between the software units, test requirements and schedule for testing software units would be generated. The detailed design of the interfaces shall permit coding without the need for further information. The software detailed design shall be evaluated for the traceability to the requirements of the software item, external consistency with the architectural design, internal consistency between software components and software units, testability, feasibility of testing, operation and maintenance.

Based on SRD, the software module developer will do software design which delivers SAD, SCP for parent document. The software test team will do software test design which gives test and validation procedure (TVPR) document to software module developer. Software review team will do software design review which delivers MOM, compliance statement, SCP for parent document, base line SRD.

C. Software coding

In this phase, the developer for each software item shall develop software unit and database, test procedures and data for testing each software unit and database.. The software code & test results shall be evaluated based on the criteria of traceability to the requirements and design of the software item, external consistency with the requirements and design of the software item, internal consistency between unit requirements, test coverage of the units, appropriateness of coding methods and standards used, feasibility of software Integration and testing.

The Software Module Developer will do computer software unit (CSU) level coding based on Software architecture document and software coding guidelines which delivers CSCI Code, Test Driver Code, SCP for parent document. The Software Module Developer will do CSU level testing which is called unit testing.

D. Software testing & integration

In this stage, the developer shall test each software unit and database, update the user documentation, test requirements and schedule for software integration. The developer shall develop an integration plan to integrate the software units and software components into the software item. The plan shall include test requirements, procedures, data, responsibilities and schedule. The developer shall integrate the software units and software components and test as the aggregates are developed in accordance with the integration plan. This phase involves conducting qualification testing in accordance with the qualification requirements for the software item. It shall be ensured that the implementation of each software requirements is tested for compliance. The developer shall evaluate design, code, tests, and test results considering the criteria for test coverage of the requirements of the software item, conformance to the expected results, feasibility of system integration and testing. The developer shall support the audit and after completion of the audit update the deliverable software products and establish a baseline for the design and code of the software item. The project envisages to list down the activities to be carried out by the developer during this phase.

The software test team (Internal) will do computer software configuration item (CSCI) activity based on SCPs, SRD, TVPL, TVPR, IDD, CSCI Code, and Test Driver to produce CSCI test results. The software module developer will do CSCI defect resolution based on CSCI test results, SRD, SAD, TVPL, TVPR, IDD. The system test team (internal)/system review team will do integration test planning /review activity based on SyRS, SARAD, Sw. architecture/ITP document.

E. System software installation

The software configuration items shall be integrated with hardware configuration item, manual operations and other systems as necessary, into the system. The integration and test results shall be documented. For each qualification requirement of the system, a set of test, test cases and test procedure for conducting system qualification testing shall be developed and documented. The integrated system shall be evaluated based on the criteria for the test coverage of system requirements, appropriateness conformance to expected results, and feasibility of system qualification testing.

There are different types of risks associated in software development such as operational risks, technical risks, management practices, and operational risks. The operational risks are: risks of loss due to improper process implementation, failed system or some external events risks. Management practices Risk includes wrong budget estimation, cost overruns, time overruns, project scope expansion. Technical risks associated with failure of functionality and performance. The advantages of managing risk are saving resources, human, income, property, infrastructure, cost, time [3]. It also protects environment, public image and prevents/reduces legal liability. A plan to install the software product in the target environment as designated in the contract to help the developer shall install the software product. The resources and information necessary to install the software product shall be determined and be available.

The system acceptance test team (external- customer) will do system installation testing which delivers system test results (external). The system software integration team will do system defect isolation that results CSCI wise defect allocation. The software module developer will do CSCI defect resolution activity which completes system software installation.

In order to minimize the risk in the software engineering projects, the following steps may be followed (Table 1);

TABLE 1. Workflow steps to be followed for reducing risks in software engineering projects

Steps	*Activity*	*Deliverable*	*Input Required*	*Team Required*
1	Software Requirement Analysis	SRD ,SCP for Parent Doc.	Software Architecture, IDD, SARAD	Software Module Developer
1.1	Software Test Planning	TVPL,SCP for Parent Doc.	SRD	Software Test Team/ Module Developer
1.2	Software Requirements Review	MOM, Compliance Statement, SCP for Parent Doc., Base Line SRD	SRD	Software Review Team
2	Software Design	SAD,SCP for Parent Doc.	SRD	Software Module Developer
2.1	Software Test Design	TVPR, SCP for Parent Doc.	SRD	Software Test Team/ Module Developer

Steps	*Activity*	*Deliverable*	*Input Required*	*Team Required*
2.2	Software Design Review	MOM, Compliance Statement, SCP for Parent Doc. Base Line SAD, TVPL, TVPR	SAD, TVPL, TVPR	Software Review Team
3	Software CSU Coding	CSCI Code, , Test Driver Code, SCP for parent doc.	SAD, Coding Guideline	Software Module Developer
4	Software CSU Testing	CSCI Code, , Test Driver Code, SCP for parent doc.	SAD, Coding Guideline	Software Module Developer
4.1	CSCI Qualification Testing (Internal)	CSCI Test Results	SCPs, SRD, TVPL, TVPR,IDD, CSCI Code, Test Driver	Software Test Team (Internal)
4.2	CSCI Defect Resolution	Compliance statement, SCP for parent doc., Baseline CSCI Code & Test Driver Code	CSCI Test Results, SRD, SAD, TVPL, TVPR,IDD	Software Module Developer
4.3	Software Integration Test Planning /Review	Segment, System Integration Document, ITP Document /MOM, Compliance Statement, SCP for Parent Doc. Base Line ITP	SyRS, SARAD, Sw. Architecture/ITP document	System Test Team(Internal) / System Review Team
5	System Installation Testing	System Test Results(External)	SyRS, SARAD, IDD, System Acceptance Test document, CSCI code, System Test Driver	System Acceptance Test Team (External-Customer)
5.1	System Defect Isolation	SCPs for Parent Doc., CSCI Wise Defect Allocation	SyRS, SARAD, Sw. Architecture, IDD, ITP document, System test results, CSCI code, System Test Driver	System Software Integration Team
5.2	CSCI Defect Resolution	Compliance statement, SCP for parent doc. Updated CSCIs Code & ITP Doc.	CSCI wise Defect Allocation report, SCPs, CSCI Code , CSCI Test Driver	Software Module Developer

N. B.: SRD: Software Requirement Document, SCP: Software Change Proposal, IDD: Interface Design Document, SARAD: System Architecture Design, TVPL: Test and Validation Plan, MOM: Minutes of Meeting, CSCI: Computer Software Configuration Item, SAD: Software Architectural Design, TVPR: Test and Validation Procedure Report, SyRS: System Requirement Specification, ITP: Integration Test Plan, Sw: Software, Doc: Document, CSU: Computer Software Unit.

IV. Conclusion

For this research, it is concluded that, the following five major workflow steps to be taken care while in development stages of software engineering projects so that the software risks can be managed to achieve project success.

References

1. Alex Down, Michael Coleman and Peter Absolon, "Risk Management for Software Projects", McGraw-Hill Book Company, 1994.
2. R. S. Pressman, "Software engineering: A practitioner's approach", McGraw-Hill Series, 7th edition, 2009.
3. Trushar B Patel and Premal Soni, "Risk Management in Software Engineering", Oriental Journal of Computer Science & Technology, Vol. 6, No. (1), pp 131-133, March 2013.
4. Barry W. Boehm, "Tutorial: Software Risk Management", IEEE Computer Society, 1989.
5. H. Hashimi, A. Hafez, and M. Beraka, "A Novel View of Risk Management in Software Development Life Cycle", I-SPAN 2012 International Symposium on Pervasive Systems, Algorithms and Networks, San Marcos, TX, pp 128-134 CPS conference publishing services, December 2012.

Author(s) Profile

Biswa Ranjan Mohapatra, received the M.E degree in Electrical & Electronics Engineering from PSG College of Technology, Bharathiar University and Post Graduate Diploma in Business Management (through distance learning program) from Institute of Management Technology-Ghaziabad affiliated to Vidya Sagar University in 2001 and 2008, respectively. During 2001-2011, he served as a fulltime employee in Central Research Laboratory-Delhi, Bharat Electronics Limited (A Government of India enterprise).He qualified in GATE national level examination conducted by IIT, Bombay in 1999; completed certificate course in Intellectual Property Rights & Intellectual Property Services from IIT, Delhi and Faculty Development Program in Management academic course from IIM, Indore, India. Currently, he is pursuing his Ph.D degree in Business Management in P.G. Department of Business Administration from Utkal University, Vani Vihar, Bhubaneswar, Odisha, India, Pin: 751004.