Towards Risk Based Effort Estimation: 
A Framework to Identify, Analyze, and Classify Risk for Early Identification at Requirement Engineering Phase

Priyanka Chandani, Jaypee Institute of Information Technology, Noida, India
Chetna Gupta, Jaypee Institute of Information Technology, Noida, India

ABSTRACT

Studies have shown that requirement defects are among the major sources of failure constituting 32.65%. It is one of the overlooked aspects in requirements engineering and is generally considered as a potential problem that can affect the projects in a negative way. The main objective of this article is to propose a risk-based effort estimation technique that identifies, analyzes, and classifies risk at requirement engineering phase to restrain them from propagating to the later stages of the project lifecycle. This article extends the scope by integrating both threats and opportunities and their further classification based on extensive requirement analysis. The validation of the proposed approach was conducted on successfully delivered real project data. A survey is also conducted as a part of qualitative analysis for analyzing the applicability of the proposed approach. The results of the proposed method are promising and strongly support findings of literature stating that the effort needed to fix issues at a later stage in project lifecycle are costly as compared to early stages.

KEYWORDS

1. INTRODUCTION

Requirement Engineering (RE) is “a systematic approach to understand, formally describe, evaluate/validate and attain an agreement on the nature of the problem” (Lamsweerde, 2000). Any failures during RE phase have an adverse impact on the overall development process (Hall et al., 2002) as it acts as a roadmap for calculating schedule and cost of the project. Studies have shown that inappropriate and misleading requirement gathering is the most expensive and are one of the fundamental drivers of project failures (Glass, 1998). As reported by (Pohl and Rupp, 2010), 60% of project venture disappointments fall into requirements engineering phase and generally aren’t found until late in development life cycle or when the project has gone live (Boehm, 1981). The same facts are supported by (Lindquist, 2005) which conclude that “poor requirements management can be attributed to 71% of software projects that fail: greater than bad technology missed deadlines, and change management issues”. Therefore, one of the significant challenges in requirements engineering is to have legible requirements, which are free from unknowns and failures.

DOI: 10.4018/IJISMD.2018100104
A study on software risk management (Dedolph, 2003) showed that only 16.2% of software projects are on time and budget. From the remaining, 52.7% are delivered with reduced functionality, and 31.1% are canceled before completion. A significant reason for failure is lack of software risk management practices in projects. Therefore, it is recommended to do risk management for requirements during requirement elicitation (Maciaszek, 2001; Weigers, 2003). If the risks are not controlled at the early stages of the project, it will result in an exponential increase in the cost of the project (Mangione, 2008) and its identification and mitigation would also be extremely difficult.

As per guide to the Project Management Body of Knowledge (PMBOK), “project risk is an uncertain event or condition, that, if occurs, has a positive or a negative effect on a project objective” (Project Management Institute, 2000). The risk which is considered good for the ecosystem of a software project and has a positive effect regarding opportunities helping software project is termed as a positive risk. Whereas risk which is a threat to software project such as failure of functionality, schedule slip, cost overruns, etc. is termed as a negative risk. The overall goal of project risk management is to minimize potential negative risk while maximizing potential positive risk (Hillson, 2002). However, as practiced by the majority of project managers they tend to concentrate on potential negative risk by spending considerable effort on identifying and managing threats, ignoring positive side of risk (Hillson, 2002).

On the other hand, effort estimation is carried out with the aim of estimating the realistic amount of effort required to develop a software system. Since 1960’s effort estimation has remained a challenging task for software developers and practitioners. Significant research in various quadrants has been focused on the construction of formal software effort estimation models. Decades of research concludes that there is no “best approach” when one estimation model or approach is compared with another because relative accuracy depends on the context (Shepperd and Kadoda, 2001). Most of effort estimation techniques takes KLOC, scale factors, cost drivers (REVIC, 1991); function points, use cases, bottoms-up, object, feature (SEER); size (SLOC, function points, use cases, etc.), constraints (size, duration, effort, staff), scale factors, historical projects, historical trends (SLIM); components, structures, activities, cost drivers, processes, functional software size (SLOC, function points, use case conversion points (UCCP), predictive object points (POPs), etc.) (True Planning) as input to estimate effort of software development.

This paper proposes an effort estimation technique to eliminate risk at the requirement level for minimizing the overall cost of project development by analyzing each requirement for the type of risk it possesses. It further exploits risk and categorizes it into positive risk and negative risk. This early detection of risk and corresponding effort can prevent its propagation to design, coding and maintenance phases. This kind of risk analysis and detection will be helpful in predicting the effort required for implementing a particular requirement before they are finalized for inclusion in the project. Based on the analysis, developers and stakeholders can collectively decide whether to include, exclude or modify the requirement. It is an integrated approach which focuses on exploring both threats and opportunities in order to minimize negative effects and maximize the chances of positive effects. The proposed approach will have following benefits during requirement engineering phase of software development:

(a) Focus on delivering clear, concise and risk-free requirements to stakeholders.
(b) Foresee the possibilities of positive and negative impact on the project and then enhancing opportunities vis-à-vis mitigating threats simultaneously.
(c) Estimate the effort around the positive and negative requirement risk.
(d) Reduction in time of requirement implementation.

2. RELATED WORK

Risk assessment and management practices provide a coherent and structured approach to identify, analyze and manage risk. Traditionally, analysis of risk is critical in software development through
Related Content

Requirements Elicitation for Complex Systems: Theory and Practice
www.igi-global.com/chapter/requirements-elicitation-complex-systems/28401?camid=4v1a

Trends in Improving Performances in Distributed Database Management Systems
www.igi-global.com/chapter/trends-improving-performances-distributed-database/37045?camid=4v1a
Service-Oriented Agents and Meta-Model Driven Implementation
[www.igi-global.com/chapter/service-oriented-agents-meta-model/28959?camid=4v1a](www.igi-global.com/chapter/service-oriented-agents-meta-model/28959?camid=4v1a)

Enhancing ERP System with RFID: Logistic Process Integration and Exception Handling
[www.igi-global.com/article/enhancing-erp-system-rfid/58513?camid=4v1a](www.igi-global.com/article/enhancing-erp-system-rfid/58513?camid=4v1a)