Chapter 57
SERIES:
A Software Risk Estimator Tool Support for Requirement Risk Assessment

Chetna Gupta
Jaypee Institute of Information Technology, India

Priyanka Chandani
Jaypee Institute of Information Technology, India

ABSTRACT

Requirement defects are one of the major sources of failure in any software development process, and the main objective of this chapter is to make requirement analysis phase exhaustive by estimating risk at requirement level by analyzing requirement defect and requirement inter-relationships as early as possible to using domain modeling to inhibit them from being incorporated in design and implementation. To achieve this objective, this chapter proposes a tool to assist software developers in assessing risk at requirement level. The proposed tool, software risk estimator, SERIES in short, helps in early identification of potential risk where preventive actions can be undertaken to mitigate risk and corrective actions to avoid project failure in collaborative manner. The entire process has been supported by a software case study. The results of the proposed work are promising and will help software engineers in ensuring that all business requirements are captured correctly with clear vision and scope.

INTRODUCTION

Every software system exhibit uniqueness and contains significant numbers of uncertainties in terms of key objectives, specific features, preferences and user expectations. This makes software structurally complex and versatile which progressively evolves over time to accommodate changing customer requirements, latest market demand, new sophisticated technologies, imprecise estimation of budget, schedules, product deployment and maintenance (Mens, 2012). These factors have strong impact on software development and strongly support the need of proactive assessment measures to control these

DOI: 10.4018/978-1-6684-3702-5.ch057

uncertainties. If failing to do so, it raises the possibility of potential risk throughout the project lifecycle ranging from delays to economic losses to customer dissatisfaction. According to PMBOK reports (2017), the global software market which is at US$333 billion in 2016 estimated to grow by 7.2% and global software projects (US and Europe) success in 2015 is 29% only (CHAOS, 2015) while 71% of projects have failed, due to diverse reasons and risks (Vahidnia, Tanriöver&Askerzade, 2016). Therefore, it is desirable to follow software risk analysis and management practices to understand, identify, and manage underlying risks to prevent the loss further in expenditure (Samantra, Datta, Mahapatra, &Debata, 2016). In software engineering domain, risk is considered as a potential problem or unwanted outcome that might have positive or negative consequences on a project PMBOK (2017). However, according to to(Hijazi, Alqrainy, Muaidi, &Khdour, 2014) risks in software development do appear due to items (usually called software risk factors) that present a threat to software project success. According to (Chen & Huang, 2009) problems in requirement are considered as one of the major sources of project failure constituting nearly 32.65%. These problems include analyzing imperfection that compromises requirement correctness, completeness, stability and meeting the objectives/ project goals and such imperfections are categorized as defects within requirements. These requirement defects are most expensive problems that persist throughout the software life cycle (Hamill & Katerina, 2009) and can be generated from different perspectives of users, practitioners, project execution or knowledge. The cost to fix a defect varies according to how far along you are in the software life cycle. The cost of fixing requirements defects is 3 times higher in the course of design, 10 times higher during development, 50 times higher at the time of testing and up to 100 times higher after the release (Boehm &Basili, 2001; Pressman, 2014). Hence, it is desirable to include risk management practices in every software project as early as possible, in particular, within Requirements Engineering (RE) phase. A project without risk management admits severe problems such as reworks of project artifacts and cost/schedule overrun.

In the past, literature has explored various issues concerning risk factors in software projects (Hijazi, Alqrainy, Muaidi, &Khdour, 2014; Islam &Houmb, 2010; Christiansen, Wuttidittachotti, Prakancharoen, &Vallipakorn, 2015) but the research is general in nature, not concentrating on perspectives including risk assessment and estimation practices in requirement engineering phase itself, uncovering requirement defects. Risk management is one such influential approaches acknowledged by all the project management and software engineering guidebooks (PMBOK, 2017; Pressman, 2014, CMMI, 2010; Thayer &Dorfman, 2013; SWEBOK, 2014). Risk can arise in any phase of the software development lifecycle and can be detrimental to the project causing huge losses. In the study conducted by (Hijazi, Alqrainy, Muaidi, &Khdour, 2014), key risk factors that threaten each phase of Software Development Life Cycle (SDLC) are presented which according their study claimed that every phase of SDLC is vulnerable to several types of risks. Many such risk factors are discussed that are common to most software development projects. With a successful risk management practices employed particularly in RE phase and by identification/analysis of risk factors related to requirements, a project manager can prevent potential risks and deter project failures. However, a comprehensive risk management plan is not possible due to paucity of resources and more onus is on saving time and budget and due to that the benefit of risk management practices cannot be reaped.

This paper proposes a tool to assist software developers in assessing risk at requirement level. The proposed tool, **S**oftwar**ERi**sk**Es**timator, SERIES in short, helps in early identification of potential risk where preventive actions can be undertaken to mitigate risk and corrective actions to avoid project failure in collaborative manner. The business objective to mitigate risk is fulfilled through the proposed tool, though it is separated from the technology implementation alike the MDA (model driven architecture)