Information and Software Technology 56 (2014) 117–133

Contents lists available at SciVerse ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

# An empirical study on the implementation and evaluation of a goal-driven software development risk management model

Shareeful Islam [a,*], Haralambos Mouratidis [a], Edgar R. Weippl [b]

[a] School of Architecture, Computing, and Engineering, University of East London, 4-6 University Way, London E16 2RD, United Kingdom
[b] SBA Research gGmbh, Sommerpalais Harrach, Favoritenstrasse 16, 1040 Wien, Austria

ARTICLE INFO

*Article history:*
Received 17 October 2012
Received in revised form 23 June 2013
Accepted 25 June 2013
Available online 4 July 2013

*Keywords:*
Software risk management
Goal modelling language
Requirements engineering
Empirical study
Goal-risk model

ABSTRACT

*Context:* Building a quality software product in the shortest possible time to satisfy the global market demand gives an enterprise a competitive advantage. However, uncertainties and risks exist at every stage of a software development project. These can have an extremely high influence on the success of the final software product. Early risk management practice is effective to manage such risks and contributes effectively towards the project success.
*Objective:* Despite risk management approaches, a detailed guideline that explains where to integrate risk management activities into the project is still missing. Little effort has been directed towards the evaluation of the overall impact of a risk management method. We present a Goal-driven Software Development Risk Management Model (GSRM) and its explicit integration into the requirements engineering phase and an empirical investigation result of applying GSRM into a project.
*Method:* We combine the case study method with action research so that the results from the case study directly contribute to manage the studied project risks and to identify ways to improve the proposed methodology. The data is collected from multiple sources and analysed both in a qualitative and quantitative way.
*Results:* When risk factors are beyond the control of the project manager and project environment, it is difficult to control these risks. The project scope affects all the dimensions of risk. GSRM is a reasonable risk management method that can be employed in an industrial context. The study results have been compared against other study results in order to generalise findings and identify contextual factors.
*Conclusion:* A formal early stage risk management practice provides early warning related to the problems that exists in a project, and it contributes to the overall project success. It is not necessary to always consider budget and schedule constraints as top priority. There exist issues such as requirements, change management, and user satisfaction which can influence these constraints.

## 1. Introduction

Software projects, by inherent nature, contain a significant number of uncertainties such as time-to-market, budget and schedule estimation, technology evolution, and stakeholders' expectations. Failure to control such uncertainties imposes potential risks on a project. Software risk management can be used as a tool to manage these risks and to reason about the uncertainties involved. There exist several risk management methods for managing risks in software projects [1–5]. Practitioners and researchers agree that for a risk management practice to be effective, it needs to be included at the early phase of the development process [6], since requirement problems are one of the main causes of project failure [7]. An advantage of considering risk management during the requirements engineering phase is that such integration enables the identification of expensive and persistent requirement problems [8]. However, the literature fails to provide comprehensive and detailed guidelines and clear evidence on how to integrate risk management activities at the early development stages [9,10,6]. Moreover, although several study results exist in the literature on identifying risk factors in software projects, only few reports are available on evaluating the impact of an overall risk management method onto a software project [4]. Research studies indicate that risk management is not well applied in practice [11] and practitioners are more concerned on the tangible development cost, which provides direct benefits in terms of project deliverables [9]. For a successful project, it is difficult to prove that any part of the resulting product is influenced by software risk management [12]. Therefore, it is necessary to integrate risk management approaches into the early development and to create awareness

[*] Corresponding author. Tel.: +442082237273.
*E-mail addresses:* shareeful@uel.ac.uk (S. Islam), haris@uel.ac.uk (H. Mouratidis), eweippl@sba-research.org (E.R. Weippl).

0950-5849/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.infsof.2013.06.003

among the practitioners about the impact of risk management practice on software projects.

Within this context, the novel contributions of this paper are (i) a goal-driven risk management method; (ii) an explicit integration of the risk management method into the requirements engineering phase; and (iii) an empirical evaluation of the impact of the risk management method into a software development project. The presented risk management method is based on the KAOS goal modelling language [8]. In particular, GSRM adopts goal and obstacle concepts from the KAOS goal modelling language and extends it with risk assessment and treatment [6]. We have decided to build our work on existing research on goal modelling because goals and risks are complementary entities of a software project. A risk is usually defined as negation to single or multiple goals or loss of attainment of corresponding objectives [8]. As such, the goal-driven approach anchors the risk management activities and allows to trace and rationalise the risk factors, events and control actions with respect to the goals. Furthermore, Goal-orient Requirements Engineering (GORE) has long been recognised in requirements engineering community as an important paradigm to elicit, analyse, and document requirements. As such, the decision to build on KAOS allows us to explicitly integrate risk management into the early development phase. The methodology is explained with the aid of a carefully designed case study for the development of an automation system in a public sector organisation (Ministry of Planning Commission) under the e-governance project. The main goal of empirical investigation is to evaluate the effectiveness of GSRM and in particular the impact of an early risk management practice on a software development project. Our work combines a case study method with action research, so that identified treatment actions can be used to control the potential project risks. The results from the case study outline the impact of GSRM on the project and compare the identified results with other literature results. Such comparison demonstrates the impact of risk management, at requirements engineering level, on software development projects.

The remainder of the paper is structured as follows: Section 2 outlines the state of the art on risk management methods and risk factors. Section 3 provides an overview of the goal-driven risk management model and Section 4 outlines the integration of GSRM into requirements engineering. Section 5 demonstrates the evaluation results on the implementation of the GSRM into a software project. Section 7 provides a critical discussion of the various parts of GSRM and it outlines some of our experiences from the studied project. Section 8 provides validity of the study results. Section 9 finally provides summary and directions for future work.

## 2. Related works

Risk, in ISO Guide 73:2002, is defined as combination of the probability of an event and its consequence [13]. It is constituted upon three basic concepts: event, likelihood, and severity. Risk management in software projects describes an integrated engineering approach with methods, processes and artefacts for identifying, analysing, controlling and continuously monitoring risks in order to reduce the chance of project failure [14,4]. This section focuses on existing work that relates to our approach.

### 2.1. Risk management framework

The theoretical foundation of putting risk management into a single framework is initially contributed to Boehm [15] risk-driven Spiral model. Later, Boehm extended the original Spiral model using the theory W (Win–Win) model [1] to satisfy the objectives and concerns of the stakeholders. The model also supports risk identification, resolution and continuous monitoring of risks. However, the approach requires intensive active involvement of project customers/users, which is difficult to attain in real on-going project situations. The Software Engineering Institute (SEI) provides a comprehensive framework to support continuous risk management activities [16,17]. The approach concerns identification, analysis, communication and mitigation strategies for software risk management and depends on risk taxonomy [18]. SEI also developed a process improvement model, Capability Maturity Model Integration (CMMI), which provides close correlation between the quality of software products and the quality of the software development processes [19]. CMMI is compatible with ISO standard for software process assessment, i.e. ISO/IEC 15504 [14]. CMMI considers continuous risk management as an important feature with concepts like risk management strategy, identifying and analysing risks and risk control. ISO 31000:2009 provides process, framework and a number of principles for an effective risk management practice [20]. Risk management is considered as an integral part of all organisational processes, including strategic planning and all project and change management processes. Kontio proposed the Riskit methodology [4], which provides a complete conceptual framework for risk management using a goal/expectation approach from the stakeholders and risks which threaten the goals. It provides precise and unambiguous definitions of risks and aims at modelling and documenting risks qualitatively. At the heart of the approach, is the visual formalism of the risk by analysing risk factors, risk events, risk reactions, risk effect sets and utility loss that would occur due to risk events. However, Riskit has some important limitations. There are no clear sources specified from where the goals originate and how the identified goals are modelled. Risks are analysed and prioritised by deriving scenarios, which is a non-trivial task when a scenario depends upon more than one probabilistic element. Moreover, it is always hard to formulate a scenario from factors and attempt to perform a comparison amongst them. Foo et al. [3] make use of a comprehensive questionnaire to construct the Software Risk Assessment Model (SRAM). A set of questions are chosen for nine critical risk elements, i.e. complexity, staff, targeted reliability, requirements, method of estimation, monitoring, process, usability and tools. However, the main limitations of this approach are the lack of a detailed implementation of the model and the lack of a set rule for common weight value, which means that practitioners need to determine a risk probability for each element. Roy's pro risk management framework [5] is an extension of the AS/NZS standard 4360:1999 [21]. The main attention of the framework is on the business component in which the project is created and the operational domain where the project is actually carried out.

### 2.2. Study results

#### 2.2.1. Risk factors

A well known *"top-ten"* list of risk factors is provided by Boehm [15]. After that several lists of risk factors have been published [22–26]. Among these contributions, Barki and Schmidt composed the most comprehensive ones. Barki et al. [22] compiled a list of 35 risk variables, which were represented in the form of a questionnaire consisting of 144 items. The results of their survey are based on the questionnaire showing five influential factors: technological newness, application size, lack of expertise, application complexity and organisational environment for the most interpretable solution of software risk. Moynihan [27] focuses on project constructs, i.e. personal constructs and application which need to be considered by the project manager. The study observed that risk variables identified by Barki et al. [22] lack the client's apparent knowledge. Schmidt et al. [23] published a comprehensive list of risk factors and categorised the risk factors into several different areas such