An Empirical Study on the Implementation and Evaluation of a Goal-driven Software Development Risk Management Model

Shareeful Islam[a], Haralambos Mouratidis[b], Edgar R Weippl[c], Jan Jürjens[d]

[a]*School of School of Architecture, Computing, and Engineering, 4-6 University Way London E16 2RD, University of East London, United Kingdom*
shareeful@uel.ac.uk
[b]*School of Architecture, Computing, and Engineering, 4-6 University Way London E16 2RD, University of East London, United Kingdom*
haris@uel.ac.uk
[c]*Secure Business Austria, Sommerpalais Harrach, Favoritenstrasse 16, 1040 Wien, Austria*
eweippl@sba-research.org
[d]*Chair for Software Engineering(14), Technische Universität Dortmund, Baroper Straße 301, 44227 Dortmund, Germany*
jan.jurjens@cs.tu-dortmund.de

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 combined case study method with action research. This allows to guide the development team for managing risks and to attain goals as well as to identify ways to improve the proposed methodology. The data is from multiple sources and analysed both qualitative and quantitative way.
Results: When risk factors are beyond the control of the project manager and project environment then it is difficult to control these risks. Project scope affects all dimensions of risk. GSRM is a reasonable risk management method that can be employed in an industrial context. The study results compare with other existing study results, to generalize findings and to identify contextual factors.
Conclusion: A formal early stage risk management practice provides early warning related to the problems that exists in the project and contributes to the overall project success. It is not necessary to always consider budget and schedule constraints top priority. There exist issues such as requirements, change management, and user satisfaction influence these constraints.

Keywords: software risk management, goal modeling language, requirements

engineering,goal-driven software development risk management model, empirical study, and goal-risk model

1. Introduction

Software projects, by inherent nature, contain a significant number of uncertainties related to time-to-market, budget and schedule estimation, technology evolution, and stakeholders expectations. Failure to control these uncertainties imposes potential risks to a project. Software risk management can be used as a tool to manage these risks and to reason under high degree of the uncertainties involved. There exist several risk management methods for managing risks in software projects [1, 2, 3, 4, 5]. However, 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], early in the development process and the development of robust requirements free of such problems. However, the literature fails to provide comprehensive and detailed guidelines and clear evidence of 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 project, only few reports are available on evaluating the impact of an overall risk management method into 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 among the practitioners about the impact of risk management practice on software project.

Within the above context, the novel contributions of this paper are i) a goal-driven risk management method ; ii) 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 rationalize 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) in Bangladesh 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, to software project development projects.

The remainder of the paper is structured as follows. Section 2 outlines the state of the art about risk management methods and risk factors. Section 3 provides an overview of the goal-driven risk management model. 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. This section discusses the evaluation approach, study context, GSRM process and lessons learned from the study context by implementing GSRM. 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. Finally Sect. 9 concludes the paper and presents directions for future work.

2. Related works

This section focuses on existing work that relates to our approach. We first discuss the risk management framework then study results on identifying risk factors and their impact on software project and barriers in risk management practice.

2.1. Risk Management Framework

The theoretical foundation of putting risk management into a single framework is initially contributed by Boehm [13] risk-driven Spiral model. Later on, 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 customer/user, which is difficult to attain in real on-going project situation.

The Software Engineering Institute (SEI) provides a comprehensive framework to support a continuous risk management activities [14, 15]. The approach concerns identification, analysis, communication and mitigation strategies for software risk management. It depends on risk taxonomy [16] and consists of constructs used for organizing risk information. The taxonomy covers 194 questions of several areas including requirements, designing, coding, testing, integration, and engineering specialties under product engineering, development process, development system, management process, management methods, work environment under development environment and resources, contracts and program interfaces under program constraints. SEI also developed a process improvement model, Capability Maturity Model Integration (CMMI), provides close correlation between quality of software product and quality of the software development process [17]. CMMI make it compatible with ISO standard for software process assessment, i.e., SO/ IEC 15504 [18]. CMMI considers continuous risk management as an important feature with concepts like risk management strategy, identify and analyse risks and risk control. ISO 31000:2009 provides process, framework and a number of principles for an effective risk management practice [19]. Risk management is considered as an integral part of all organizational processes, including strategic planning and all project and change management processes.

Karolak proposed the Software Engineering Risk Model (SERIM) [2] by considering the Just-In-Time (JIT) software approach. SERIM attempts to minimize the amount of risks involved, while optimizing the contingency strategies for problematic situations. The approach considers three main risk elements, technology, cost and schedule from technological and business perspectives. These elements are interconnected with 81 risk factors. The risk factors are influenced by organization, estimation, monitoring, development methodology, tools, risk culture and usability. However, no empirical validation reports are available of the SERIM approach and it does not show when and where the risk management initiates during the development process and who will be involved into the process. 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 analyzing risk factors, risk events, risk reactions, risk effect sets and utility loss that would occur due to risk events. Riskit has some important limitations. There are no clear sources specified from where the goals are originated and how the identified goals are modeled. Risks are analyzed and prioritized 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 is 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. Determining element probability is a difficult task, which depends on project specific context and involved practitioner's belief. Roy's pro risk management framework [5] is an extension of the AS/NZS standard 4360:1999 [20]. The main attention of the framework is on business component in which the project is created and the operational domain where the project is actually carried out. Prikaldnicki et al. [21] proposed an integrated risk management process across three different organizational levels i.e., strategic, tactical and operational, in particular for the global software development (GSD) projects. It follows a reference model that focuses on two different dimensions, i.e., organizational and project and consists of four different phases including new project, project allocation, project development, evaluation and feedback.

2.2. Study Results

Several survey studies were performed on identifying risk factors and assessing their impact on software development projects and barriers to implement a formal risk management practice. The participants of these surveys are mostly experienced practitioners who express their view about risk factors and their impact in the software development. Some studies also focus on the challenges for implementation of risk management in software development projects.

2.2.1. Risk Factors

A well known *"top-ten"* list of risk factors is provided by Boehm [13]. After that several lists of risk factors have been published [22, 23, 24, 25, 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 based on the review of the existing Information System (IS) literature. The results of their survey are based on the questionnaire showing five influential factors: technological newness, application size, lack of expertise, application complexity and organizational environment for the most interpretable solution of software risk. Moynihan [27] focuses on the 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 client's apparent knowledge. Schmidt et al. [23] published a comprehensive list of risk factors by conducting Delphi survey study with the experienced project practitioners through three different panels located in three different countries. The study categorized the risk factors into several different areas such as corporate environment, sponsorship, relationship management, project management, scope, requirements, funding, scheduling, development process, staffing, technology, and external dependencies. Arshad et al. [25] published a list of important risk factors and ranked them by following the survey response. Iacovou et al. [28] summarized a risk profile by the top ten risk factors for offshore-outsourced development projects. The risk factors are ranked as very important, important and less important by focusing on three main areas: communication, client's internal management and vendor capabilities. Nakatsu et al. [26] further investigated and compared the risk factors between offshore and domestic outsourcing. The results showed that many risk factors related to project management commonly appeared on the top of both domestic and offshore risk lists such as lack of top management commitment, inadequate user involvement and failure to manage end user expectation. Some are unique for the offshore context such as lack of communication, poor change controls, lack of business know-how and failure to consider all costs. Aspray et al. [29] provided an ACM task force report that considers risk related to both technical and nontechnical issues.

2.2.2. Risk Factors Impact

Some works have been undertaken to investigate the potential effects of risk factors on software development projects. Wallace et al. [24] focused on risk factors from six different dimensions and their effect on a project. The study considered low, medium and high risk projects and observed the impact of risk dimensions on the projects. The six risk dimensions are, user, team, requirements, planning and control, complexity and organizational environment. The results showed that risks associated with requirements, planning and control and organizational environment are more obvious for the high risk projects. Ropponen et al. [30] identified six components of software development risk which are scheduling and timing, system functionality, subcontracting, requirement management, resource usage and performance, and personnel management. The result showed that awareness of the risk management impact and systematic risk management practice has an effect on schedule, requirements and personal management risk. Jiang et al. [31] examined the risk impact on different system development aspects ,in particular, the study looked at the 12 most common software development risks proposed by Barki [22]. Empirical evidence on the relationship between risk and project effectiveness was considered. Two common risks observed were lack of expertise and clear role definition. According to that work, Practitioner expertise is further refined with the ability to work with uncertain objectives, top management and as a team, and ability to carry out tasks effectively. Member selection for the project must comprise clear definition of specific tasks, expected outcomes, rewards, timing, responsibilities,

and reporting relationship. The research concluded that controlling these two risk dimensions can effectively contribute to the project success.

2.2.3. Software Risk Management Barriers

Ropponen et al. showed that [11] 75 % of the project managers did not follow any detailed risk management practice and did not have adequate knowledge about software risk management. However for a successful project, it is difficult to prove that any part of the resulting product was influenced by the software risk management [12]. Kwak et al [32] observed that project managers and practitioners perceive risk management activities as extra work and expenses. Nyfjord et al. in a survey study [10] outline several problems to integrate risk management into the development. They categorise them as resource problems (i.e., expensive training cost, lack of resource and time), organizational problems (i.e., different roles, lack of competence, overloaded work), scope problems, and process problems (i.e., lack of coordination, integration and planning). Another recent survey study of experienced project managers [9] on their perception of software risk management outlines tangible development cost, lack of resources and efforts from organization or process change due to risk mitigation are top three identified barriers of software risk management. The result also emphasizes other barriers such as too much risk to control, reward for problem solving not for risk mitigation, overconfidence on individual measurements that pose challenges for the implementation of risk management in the development. Pfleeger [33] emphasizes false precision (i.e., lack of risk probability distribution data and obscure value), questionable science (i.e., relevancy and quality of the data being studied, ignorance or giving less credence to qualitative data), and confusion of facts with value (i.e., risk quantification without knowing the consequence).

Risk management frameworks, standard and study results discussed above are important but they also demonstrate a number of limitations. For instance, No empirical validation reports are available to understand how applicable Karolak's SERIM approach is in a real project context. Similar to GSRM, Kontio's Riskit is also a goal-driven approach but it is not clear from where the goal can originate and risk analysis is based on scenario which is difficult to formulate. Considering the risk management standard ISO 31000:2009, it provides generic high level guidelines for implementing risk management into organizational process using process, framework and principals. But how risk management should integrate into organization process and what techniques need to follow for performing the risk management activities are not clear from the standard. The existing risk management frameworks, standards, and survey results from practitioners emphasize on the integration of risk management from the early development process. However details to integrate risk management activities into software project are still missing. A little work has been undertaken to understand the impact of overall risk management framework into software development. In contrast, our work improves the current states of the art by i) introducing a goal-driven approach for risk management; we demonstrate the source for goal and risk factors and categorise the identified goals and risks factors using component-element-factor hierarchy; ii) explicitly integrating risk management activities into early development; we use artefact and process oriented view to systematically integrate GSRM into requirements engineering phase; iii) defining operationalised activities for the risk management process; GSRM precisely defines textual and visual artefacts under the activities. The process initially determines how risky the project is before starting the risk management activities; iv) empirically evaluating impact of a risk management method in particular GSRM into an active ongoing

software project.

3. Overview of the Goal-driven Software Development Risk Management Model

The Goal-driven Software Development Risk Management Model (GSRM) is a framework that supports assessment and and management of risks from the early requirements engineering phase. It is an extension of the KAOS goal modelling language with concepts related to risk management. It is worth stating that the focus of this paper is not to present in detail the GSRM framework but rather to present its applicability and its empirical usage in a case study. GSRM has partially been presented in other publications [34, 35, 36]. This section presents a brief overview of GSRM, which demonstrates some distinguished characteristics, i.e., i) the explicit integration of risk management into the early development stages, ii) the involvement of project stakeholders as an active contributor to the risk management process, and iii) the usage and modelling of goals to rationalize risk management.

For an effective and efficient software development and project success, we categorise development components into five dimensions : project execution, development process, product, human, and environment (internal & external) [37, 38, 39, 36]. Individual components provide an abstract view that is generally comprised of single or multiple elements. Elements are the essential parts that describe a component. The elements may further be characterized by single or multiple factors, and if necessary, also refined into sub-factors. Therefore, factors are the lowest level refinement of the development component and represent a concrete aspect of the development. Elements and factors together represent the components, by following development activities, project execution, product specification and quality, human and environmental issues, and the resulting artefacts. For example, the project execution component is described by elements, such as planning and control, project scope and tool support; where planning and control are further refined into factors such as budget, schedule and milestones, monitor, complexity and change management. GSRM defines this as a *component-element-factor* hierarchy. Generally, the elements are intertwined, interdependent, and contribute to attain single or multiple development goals. This hierarchy includes both technical and non-technical development issues, which support the consideration of a holistic view on software risk management.

3.1. Levels of Abstraction

The model supports different levels of abstraction from goal to obstacle and finally to treatment. Figure 1 gives an overview of the different levels of abstraction, depicting exemplary questions that symbolize the characteristics of the proposed model. We divide the levels of abstraction into three main areas. These levels build the bridge from the goals of a software project to the risks that obstruct these goals, and finally, the treatments that reduce the risks to satisfy these goals. On the top level, there are the goals, i.e., objectives, expectations, and constraints from the development components. These goals map with the project success indicators. In the middle two levels the risk factors, which directly or indirectly obstruct the fulfillment of the goals, are shown. The risk factors cause undesirable events that bring negative consequences to the goals. Risk events are then assessed to estimate the severity of the risk. At the bottom level, there are the control actions that obstruct the risks and associated consequences to attain the goals. This abstraction supports refinement of goals and obstacles and establishment of the obstruction and contribution links amongst

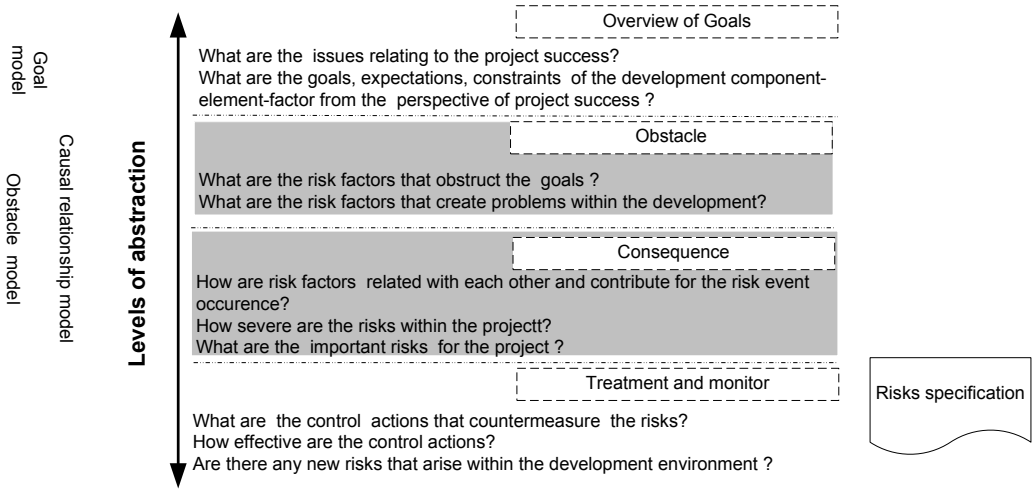

Figure 1: Overview of Risk Abstraction Levels

them. The refinement is fulfilled through vertical abstraction [40] so that goals are traced by the control action for their satisfaction. The more goals and risks are refined, the better the support of risk management at an early stage. The *Risk specification* is the main artefact produced by the model as shown in the right hand side of Figure 1. On the left side, the figure shows different models,i.e., goal model, obstacle model and causal relationship model. A goal model refines higher level goals to lower level finely grained sub-goals through AND or OR refinement so that sub-goals contribute to the main goals. An obstacle model is the refinement of risk factors to the risk event and obstruction link from obstacle to the goals. Therefore, an obstacle model is a goal-risk model as it links from the obstacles to the goals. Finally, a causal relationship model visualises the factors that are responsible for the occurence of a risk event.

3.2. Framework

The framework consists of four layers to support software development risk management. The advantage of a layer based modelling framework is that it includes suitable tasks, methods and techniques for performing specific activities under any layer.This section provides a brief overview of the layers.

Goal Layer. The concept of a Goal is the core concept of this approach. The goal layer focuses on the factors that contribute effectively to complete the project activities and directly link to the project success. Such goals are important as they describe what needs to be done for a successful project and who will be responsible for achieving the goal. Goals are project specific and focus on the economic benefit, project success criteria and boundary, knowledge gathering and reuse, user satisfaction, quality, vendor reputation, successful delivery, and other critical issues of the product. Goals can be stated at different levels of abstraction from higher level coarsely grained to lower level finely-grained goal assertions. The more the

various goals are refined, the easier it is to identify and analyze the risk factors that obstruct these goals. Goal refinement results in the construction of the goal model. Goals in software project can be of several types such as information , satisfaction, maintain, improve, and reduce.

Obstacle Layer. Obstacles are the main causes that reduce the ability to achieve a single or multiple goals. We treat risk factors as obstacles that directly or indirectly lead to a goal negation and create problems in the project. Obstacle categories should be aligned with and derived from the goal categories and model the situation about how several obstacles violate the identified goals. The obstacle layer identifies the potential software development risk factors and it formulates the obstruction to the goal dissatisfaction. Similar to the goal model, the obstacle model aims to provide a complete overview of the risk factors that exist in the project. The risk obstacle layer establishes the obstruction link from risk factors to sub-goals and from events to the main goal. Different software development risk factors support different categories of obstacles such as dissatisfaction, lack of adequacy, misinformation, wrong assumption and inaccuracy.

Assessment Layer. The Assessment Layer quantifies the risk events as a consequence of single or multiple risk factors. Risk quantification is an important first step in risk assessment [22]. This task is non-trivial due to the inherent subjective nature of the risks in the software engineering domain [4]. This layer precisely annotates individual risk events and it establishes the causal relationship model between risk factors and related risk events. It also focuses on the severity of the risk events' impact to goals. For high prioritized goals, obstacle identification and refinement should be extensive. It is worth stating that the same risk factor may lead to more than one risk event and the same risk event can obstruct more than one goal. Conversely, a goal is obstructed by multiple obstacles that relate risk events and associate factors. This representation allows us to model situations where an event is influenced by more than one risk factor and impacts on single or multiple goals. An obstruction link is established from the risk event to the specific goal that it obstructs. This approach supports the construction of a *goal-risk model* by refining risk factors to risk events and their combined obstruction to goals.

Treatment Layer. The treatment layer focuses on the control actions to counter the risks so that goals can be attained. It also monitors the effectiveness of the implemented control actions and identifies any new risks throughout the development. The main aim is to gain control of the risks as early as possible and preferable during the requirements engineering phase. Generally, there exists an alternative countermeasure to the obstacles but one should select the most cost effective one for the risk mitigation. This layer includes two different links: contribution link from the control action to the goal that it fulfills and obstruction link from the control action to the specific obstacle that it obstructs. The treatment layer allows modelling, reasoning, and tracing the adopted control action for the risk mitigation and goal satisfaction. It also includes a responsibility link from the control action to the agent so that the specific active agent would be responsible for implementing the control action in order to mitigate the risks.

Figure 2 shows the modelling framework of GSRM. GSRM uses the same notations for goals (parallelogram) and obstacles (reverse parallelogram) as used in the KAOS model. Goals are refined through AND and OR refinement into sub-goals. Obstacles are linked to

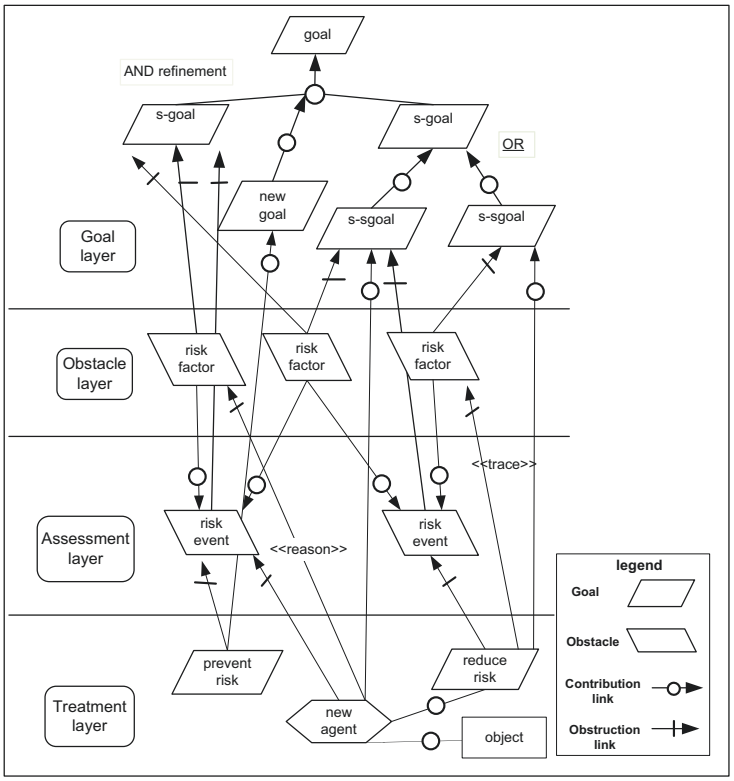

Figure 2: Framework of GSRM

the goals through obstruction links. The Treatment layer includes the agent who has the responsibility to implement selected actions to control risks. The framework is initiated with a goal in terms of project success and it ends with goals in terms of risks mitigation.

3.3. GSRM activities

The activities describe all the tasks and steps that are required for goal-driven risk management and create the risk specification artefact. In the literature several contributions and risk management standards agreed that to be successful, risk management must be run as an iterative process involving repeated risk assessment and project specific risk mitigation activities throughout the system life cycle. We follow these guidelines in order to describe the activities and tasks for GSRM. The goal-driven risk management activities are performed sequentially for the first iteration and continue further, if required are tailored, depending on the project context. Figure 3 gives an overview of the activities, tasks, and steps involved within the process model.

The first activity initialises goal-driven risk management during the requirements engineering phase. This activity defines the risk management scope, threshold & boundary, it allocates schedule and formulates risk management team. A Project's inherent riskiness

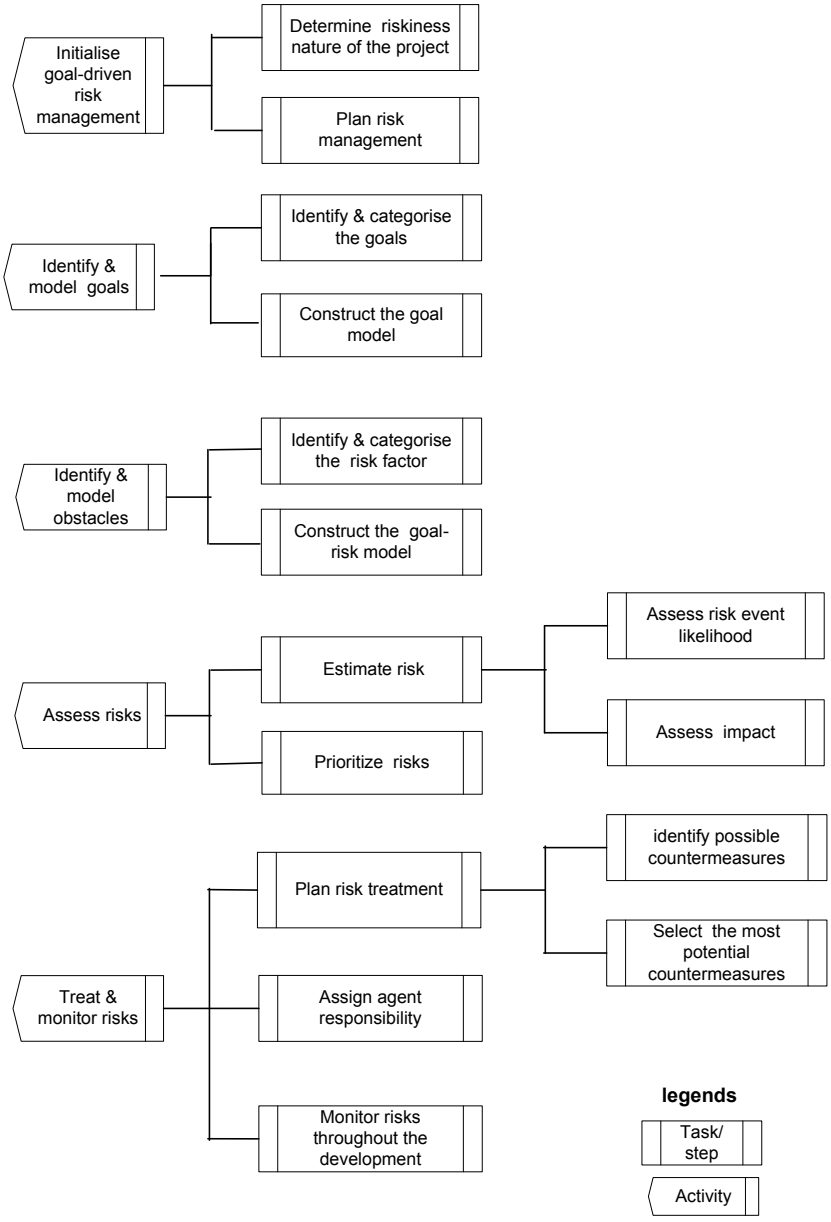

Figure 3: The Activities and Tasks for Goal-driven Risk Management

nature is important for this activity. In particular how risky is the project guides the treatment of the risks related to the project. Once the risk management plan is initialised, the next activity iinvolves the identification and modelling of goals, by focusing on the state of the

development components and mapping them with the project success indicators. It mainly identifies and categorises goals and refines high level goals to provide more concrete meaning in terms of their contribution to the project success. The Goal refinement activity results in the construction of a goal model from a parent goal to the sub-goals. Goal details is the artefact, produced by this activity, which triggers subsequent risk assessment and management activities. The next activity involves the identification and modelling of obstacles. Risk factors are identified as main causes of obstacles and responsible for the occurrence of risk event. We need to identify as many likely obstacles as possible so that the development team are aware of the problems from the beginning. Obstacle identification extends the goal model by including obstruction links from obstacles to goals. This extension support the construction of goal-risk models that visualize influential risk factors, which obstruct multiple goals. The following activity assesses risk by estimating the risk level and relevant priority. We start with the causal relationship model by following risk factors and associated risk events as a consequence. This allows us to focus on the relevant risk events for the risk level estimation, rather than considering all raw risk factors. Risk estimation considers risk event likelihood and its severity of impact on goal negation. The impact assessment builds the cause-consequence relationship from the risk event to the obstructed goals. The risk assessment finally prioritizes the risk so that high prioritized ones get immediate attention. The final activity of GSRM aims to control risks as early as possible and to monitor the effectiveness of the control action throughout the project. Risk treatment needs to be planned and mapped with the risk management scope. There may have been multiple control actions but depending on the project context, only potential ones should be selected. Once the selected control actions are implemented then this activity monitors risks throughout the project life cycle. The GSRM process constructs artefact type *risk specification*, which includes concepts such as *risk management plan*, *goal details*, *risk details* and *risk status report* and model elements such as *goal-risk model* and *causal relationship model*.

4. Integration of GSRM into requirements engineering

Risk management should be an inherent component of a software project [13] and it needs to be considered as early as possible. We advocate to consider it during the early requirements engineering phase. However, requirements engineering and software risk management are two different processes. The integration endeavor requires to consider the interactions among the underlying activities, tasks, methods, roles, and dependencies amongst the artefacts that are involved between the two processes [6]. To begin with, we examine two different perspectives, i.e. artefact and process oriented view, that allow to understand rationale in terms of the integration. We consider several integration points from the artefact and process oriented views. By *Integration point*, we mean anything that explicitly connects risk management into requirements engineering as criteria to integrate between two different processes. For instance, an artefact oriented view focuses on the dependencies among the requirements engineering or risk artefact types, its content items, and associated concepts [41, 42]. The process oriented view focuses on the interaction and dependencies among activities, tasks, and techniques along with the roles and responsibilities of requirements engineering and risk management [43].

4.1. Artefact Oriented View

Artefact oriented requirements engineering is a systematic methodology that describes the problem space of the system-as-is as comprehensively as possible towards complete requirements specification documents [41]. It combines both structure and content of the artefacts, expressed by the domain-specific concept model [42, 44]. The structure is described with a taxonomy reflecting the hierarchical structure of produced specification documents. The included content (the underlying concept model) reflects the concepts of an application domain in terms of precisely defining the used elements and their relations for specific description techniques [45]. In the business information system domain, generally two artefacts types are considered: business and requirements specification. These artefact types formulate goals, capabilities, constraints, system vision, and requirements (information system, organizational and integrational) [46]. Similar to the requirement specification artefact, GSRM also provides artefact type risk specification. The risk specification encompasses content item such as goals, obstacles, and treatment, which in turn consist of concept types such as risk management plan, goals details, risks details, and risk status report. Among the requirement and risk concepts, goals are one of the fundamental ones to support the integration. Thus, goals provide background foundation to elicit and analyze requirements and risks. The elicited requirement artefacts, in particular, business, user, and system requirements support the identification of risk factors. In fact, requirements are amongst one of the elementary inputs for risk identification. Quality of the elicited requirements is a highly influential factor in attaining project goals related to schedule, budget, product quality, and error free requirements. Reducing project risks is one of the critical requirements of a software project. Risk concepts contribute to reduce errors from the elicited requirement. Attributes such as risk level, priority, control action, and risk status help to reduce requirement errors. Commonly in requirements engineering, the elicited requirements are prioritized. Higher priority requirements get immediate attention. Attributes like requirements acceptance, time constraint and design decision are also supported by the risk status report. Figure 4 shows the interaction between requirement and risk artefact concept. We conclude that requirements and risk artefacts depend upon each other. One of the main focuses for integrating risk management into requirements engineering is to create and manage an error free, complete and robust requirement specification document and to control the human and organizational issues related to the project success.

4.2. Process Oriented View

The activities, tasks, and methods of the requirements engineering and risk management process are coherent and interrelated. Both the requirements engineering and risk management processes consist of sequence of activities, tasks, and steps in order to construct the artefact concepts. Requirements engineering is mainly comprised of eliciting, analysing, validating, and managing activities, which further contain fine-grained tasks and sub-tasks within these activities. These activities begin nearly in parallel to the activities of business modelling. As mentioned before, GSRM process consists of activities, tasks, and steps to support the risk specification artefact type. Activities of both requirements engineering and risk management are sequential and techniques used within the activities are partially similar. For instance, requirement elicitation commonly relies on the background study of specific type of artefacts, including pre-existing documents about the system as-is, such as organizational charts, policies, work procedure, business rules, data samples and scenario analysis

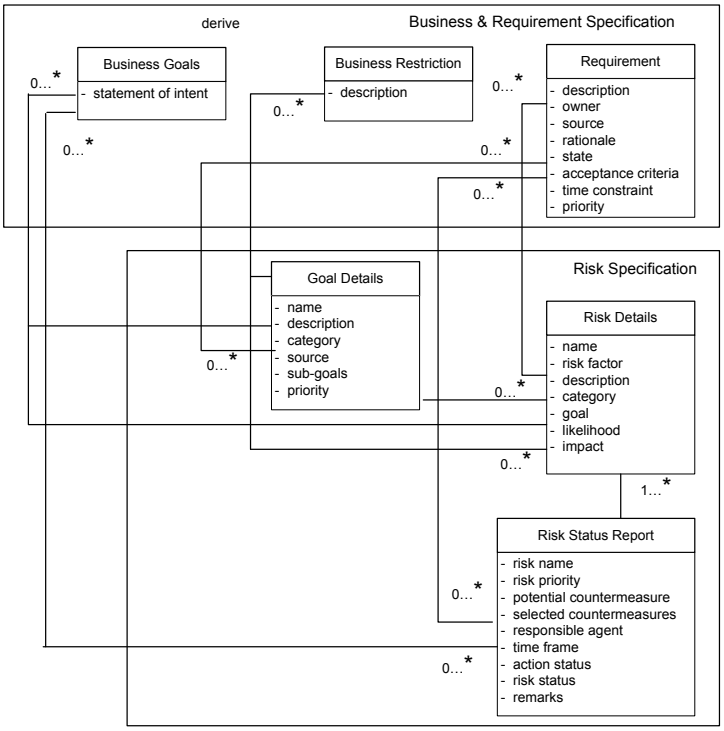

Figure 4: Attributes Dependency of Requirements and Risk Artefacts

of the interaction among the systems [8]. The method also focuses on the stakeholder-driven elicitation through structured and unstructured interviews and joint workshops. Goals and risks identification of the GSRM focus on the preliminary analysis of the system-as-is, such as project information, project domain analysis and requirement artefacts. The taxonomy based questionnaires and brainstorming sessions with stakeholders are very effective techniques for the risk identification [16, 47]. Therefore the techniques used and input artefacts required for the goal, requirement, and risk identification are similar. The activities and tasks under the process couple to one or more roles, which take the responsibilities for producing the related artefacts. Typical roles for the requirements engineering are customer/user representative, business analyst and requirement engineer. Requirement engineer is the key responsible person, who creates and manages the requirement specification by aligning the business needs to the software-to-be. He establishes the bridge among business analyst, architect, project manager, and customer/user. Risk manager is the person mainly responsible for performing risk assessment and management activities. He should have adequate knowledge of the project domain and sufficient skills to handle the risks in a specific project situation. Generally in small and medium projects, the project manager is concerned with the overall project execution and takes the additional role as risk manager. A successful project manager is always a good risk manager [48]. The requirement engineer active participation is more important within this context.

5. Evaluation and Data Collection

5.1. Study Design

We have chosen to employ an empirical research method to evaluate GSRM. Software development projects are a multidimensional undertaking, and validating a single method from a complex set of activities is a difficult task. Hence there exist a number of challenges when performing an empirical study within the software risk management domain. Firstly, software development projects generally contain long durations and always put the focus on time, budgetary, and quality control. In such setting, a comprehensive risk management practice is not always possible. Secondly, project managers and practitioners lack motivation to perform comprehensive risk management activities into software projects, as risk management is usually consider extra work. Lack of practitioners' motivation and involvement reduce the validity threats and reliability of the overall study. Finally, every project is unique and evolves due to changes of the project scope, market demand, and technology. A project may contain too many risks in various dimensions and it is not possible to control all of these risks. Risks are subjective by nature and past project risk values may not provide accurate estimation in the running project context. These challenges make it difficult to assess the precise effectiveness of a comprehensive risk management method in the project.

Nevertheless, despite all of these challenges, there exist a limited number of study results such as [49] about the impact of risk management on the overall software project. We previously conducted a case study for evaluating GSRM and its integration into requirements engineering [36]. However, the study has focused on identifying the issues related to integration of risk management into requirements engineering and GSRM was not fully employed due to tight budget and schedule pressure. However the obtained results provide important conclusions about the integration of risk management into requirements engineering phase. the current study focuses on further investigating GSRM by implementing it into an active on-going software development project. In this study, we combined case study methods with action research. Generally, an action research makes an effort to provide practical value to the study subject while simultaneously contributing to the acquisition of new theoretical knowledge [50]. This allows us on one hand to guide the development team for managing risks and to attain goals during the development and on the other hand to identify ways to improve the GSRM methodology. For our case, GSRM follows project documents and artefacts and interviews team members to identify goals and risks. Risk management results are effectively used to control the identified risks and to contribute to meet the project goals in order to improve the overall studied project situation. Our study combines theory, practice, case study, and action research so that in-depth understanding of the GSRM impact into the software projects can be demonstrated.

5.2. Data Collection and Analysis

Figure 5 depicts the details of the study. The study used project specific documents as input and carried out kick-off, brainstorming, and interview sessions to execute the various risk management activities. Closed questions were used during the interview session. Feedback about GSRM through a set of open questions used to evaluate the applicability of GSRM in a subjective way. The studied company's management was convinced to integrate a comprehensive risk management practice in the project due to the project inherent risky nature. Furthermore the first author has a long standing relationship with the company's

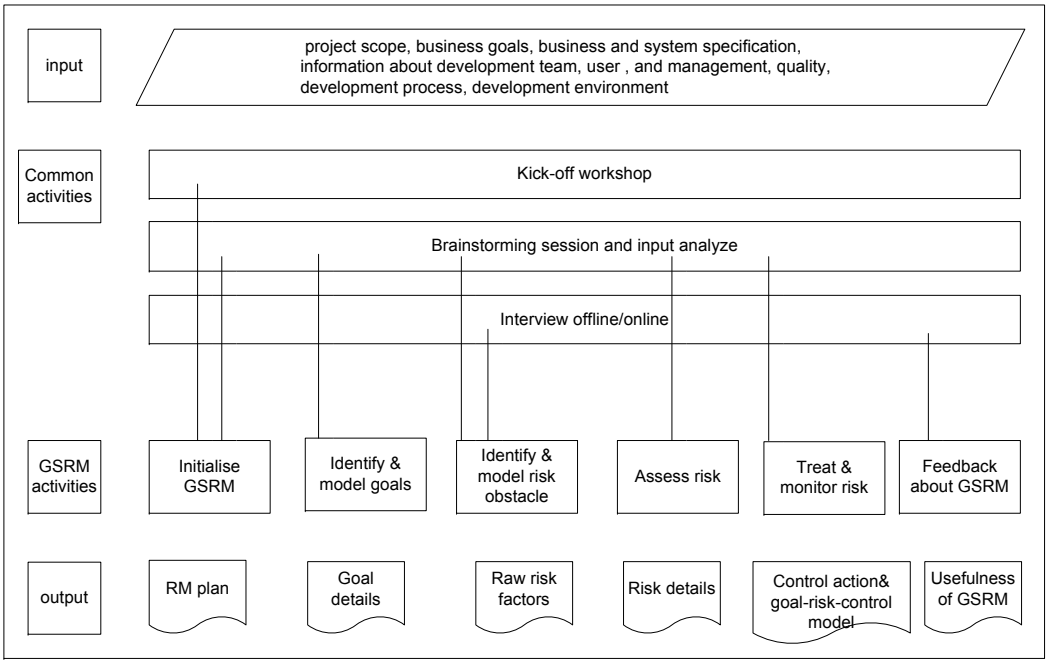

Figure 5: Case Study Details

management which has made it easier to obtain consent from management to implement GSRM into one of the running projects of the company.

We collected data mainly through interview and brainstorming sessions with the project participants, users, and sponsor representatives and by analyzing the project documents and development artefacts. The kick-off session was used amongst the members of the risk management and project team. Both qualitative and quantitative data were collected and analyzed to meet the study constructs(mentioned in the next section). Two different sets of questionnaires (closed and open questions) were used for data collection. Closed questions focused on the existance and frequency of problems and their causal dependencies to the risk events and consequences. The questions were arranged by following component-element-factor hierarchy and answers contained three possible values. The open questions are mainly descriptive and comparative questions used to identify participants' perception about the goal-driven risk management approach and its integration into requirements engineering. Open questions also answered specific practitioners' beliefs about risk management and its importance during the development process. In the beginning, a kick-off workshop was carried out amongst the members of the development team to provide an overview of GSRM, risk artefacts, and its integration into requirements engineering. The open and closed question responses and project artefacts were input to the brainstorming session. Three Master of Information Technology (MIT) students of the Institute of Information Technology (IIT), University of Dhaka, Bangladesh, former students of the first author, were employed for the implementation of GSRM. The students had adequate knowledge and

experience working in software projects. The collected data was analysed both qualitative and quantitative. In particular, quantitative analysis considered information like effort spent for performing the risk management activities, the number of risks factors and risk events, and the number of controlling actions and their effectiveness. Therefore, the responses and observations from the closed question sessions were used as input to analyse the data. Qualitative analysis considered issues like using goal-driven approach for risk management, risk management impact to the project, integration of risk management into requirements engineering, and adequacy of activities and artefacts of GSRM. Participant open questions responses, in addition to other observations, were used on this occasion.

6. Case Study

This section explains the implementation of GSRM into a software project. The activities of GSRM are systematically implemented in the studied project to evaluate the approach.

6.1. Study Context

6.1.1. Company Profile

The study reports on our empirical work of a software development project at Domain Software Technologies Ltd. (Domain Tech), a subsidiary company of Domain Technologies Ltd, (London, UK) and Domain Consulting (M) Sdn. Bhd. (Kuala Lumpur, Malaysia - Sister Company) [51]. Domain Tech started its operation, in 2002, in Bangladesh with multi dimension lines of business, such as software development, global IT support, and consultancy. Since then, several projects were successfully completed for clients located in UK, USA, and Malaysia. The company obtained adequate experience of offshore as well as inshore software projects. Currently the company has a total of 72 employees at the Bangladesh site.

6.1.2. Project Context

The project, considered in the case study, aimed to automate the Planning Commission Campus of the Ministry of Planning, as a part of the e-Governance project of the government of People's Republic of Bangladesh. The project was officially initiated Nov 2009 once Domain Tech obtained the work order from the ministry. The project context was the development of an application software that contained ten separate modules to automate day to day ministry operational activities. The modules were: Project Planning, Personnel Management, Payroll Management, Budgeting, Auditing and Accounting, File Tracking, Letter Dispatching, Meeting minutes, Library Management, Inventory and Vehicle Management. Every module have its own features, constrains and requirements and it relates with other modules. For instance, the personnel management system managed profiles of all planning commission employees including individual job records, leave management, and transfer records. The Personal management system features were: configuration, staff profile, leave management, and reports. The Payroll management system features were configuration, salary structure, festival bill, loan/advance, increment and reports, and budget system features are configuration, budget, bill, audit and reports. Two features were common to every module, i.e., configuration and reports. Configuration allowed users to customize individual modules for effective use of the software with minimum change. The purpose of the reports was to produce documented output such as office order and notice. Thus, the main project goals were:

- Automate the whole planning commission campus;
- Digitize old and new data and dynamic report generation;
- Interconnect with existing software in the commission campus;
- Train employees for the new system.

6.1.3. Project riskiness

This was the first government project by Domain Tech, where the main users were government employees. Initially a high level technical specification about the different project modules was developed by a consultant company on behalf of the ministry. Based on the specification, Domain Tech similar to other vendors, submitted an expression of interest and successfully obtained the work order. Apart from the office automation application software-to-be developed, project context also included a comprehensive user training. The development team consisted of 15 members with duration of 13 months. A total of around 1150 users were located in the main campus. The Domain Tech management and main project team members considered the project as high risk even though practitioners have experience to work in the similar project. The reasons were:

- Lack of experience to handle government employees;
- High level initial specification;
- Large numbers of users training who are government officials;
- Effective usage of the product and user satisfaction;
- High reputation regarding successfully completion of the project.

Despite the project being considered as a high risk project, the management decided to undertake it in order to accomplish the vision. The management vision is to successfully develop the office application and customize it for other government ministries and agencies. High reputation and user satisfaction may give Domain Tech a competitive advantage on the market as recently the local government decided to digitialize its ministry offices under the *Digital Bangladesh Vision 2021* project [52]. The project was important for the Domain Tech to capture the market for future business opportunities.

6.1.4. Study Construct

The main focus of this study was to specify the impact of goal-driven risk management model on the software development project. Study goals are to

1. Evaluate the advantages and limitations of goal-driven risk management in software development projects;
2. Improve our understanding of the issues involved in integration of risk management activities into requirements engineering

.

Project participants' positive and negative observations, risk management results, performance of the activities, and process integration were mainly used to evaluate the benefits and weaknesses of GSRM. Furthermore, as mentioned before, quantitative metrics, such as effort spent on GSRM in requirements engineering, number of risks and treatment actions over time were also considered to support the study goals.

6.2. Introduction of GSRM Process

6.2.1. Activity:1 Initialise Goal-driven Risk Management

A kick-off workshop by the risk management team was initially carried out to provide an overview of GSRM. The final part of the workshop was used for the GSRM initialization activity. The Project Manager (PM) elaborated the factors which brought the project as a risky one. The first task under this activity (i.e., determine the riskiness nature of the project) was considered and the main factors for high risky project were noted. Project scope, success criteria, business goals, and initial high level system specification were used to define the risk management context. The risk management scope was to control risks related to the application development from a holistic perspective, specifically project execution, user, product specification, quality, and user training. The risk management scope was biased by the project inherent challenges and project scope. However, risks related to project sponsor, fund support, and user internal organizational problems were not considered within the scope. The Risk Management(RM) team consisted of a PM as the team leader along with 3 students and 2 development team members. Therefor the PM was the main authority responsible for the successful implementation of risk management activities and for the communication of the results to management and user representatives. The risk management was scheduled at the first deliverable phase, i.e., requirements specification and design phase. The interview participants from both users and sponsor representatives were identified and schedules were also planned. However no schedule was considered for risk monitor and the PM categorised it as a demand-basis activity.

6.2.2. Activity 2: Identify and Model Goals

This activity was carried out by a brainstorming session amongst the members of the RM team. A Risk management plan was considered as input for this activity. Furthermore, project artefacts, such as project scope and execution, business scope, system specification, human and overall development environment and Domain Tech business vision were also considered for the identification and modeling of the goals. Four prioritized high level goals were agreed for the project. These were: *complete project within estimated budget and schedule*, *complete user's training*, *obtain positive reputation*, and *generalize the application for other government ministries*. The first two goals were considered through focusing on the project contract and needed to be achieved before the system goes to operation. The remaining two goals were critical for the future business vision of Domain Tech. It needed user positive feedback, overall product quality, and successfull deployment and maintenance of the software. High vendor reputation could give a competitive advantage to obtain a work order of a similar government project. Generalizing application by gathering knowledge from this project could significantly reduce the development cost from the vendor's perspective. These goals were refined during the session. Table 1 outlines the goals and sub-goals which were agreed by the RM team. The goals were related to each other and same sub-goal linked to more than one parent goal.

6.2.3. Activity 3:Identify and Model Obstacles

Initially interviews were carried out by student members with the selected practitioners of development team. The interviews continued further with 12 users and 1 sponsor member of the United Nations Development Program(UNDP). However it took more time than expected with the user members as they did not follow the agreed schedule and they demonstrated

Table 1: Identified Goals and Sub-goals

Goals	**Sub-goals**
Complete project in estimated budget and schedule	maintain estimated budget in development maintain estimated schedule in development maintain realistic estimation clear milestones competence practitioner reduce errors from requirements user active participation user positive motivation
Complete user's training	adequate training budget professional and competence training complete training and user manual user active participation user positive motivation improve effective communication & coordination improve practitioner motivation& productivity reduce user and practitioner conflict
Obtain positive reputation, Generalize the application	user satisfaction complete project in estimated budget and schedule quality product successful training details understanding of business process successful system usage complete elimination of existing manual system complete specification

lack of IT and project domain knowledge. The RM team also needed to provide an overview of goals, risks, and the main purpose of the interview before the actual interviews took place. 200 closed questions were used for the interview. A sample set of the questions is included in the appendix. Interview responses were complied to generate a raw list of risk factors. The raw list was very large and was refined by following the identified goals and brainstorming session.

Most of the identified risks were related to the project execution, product, and human perspectives. The development team only had a high level initial specification. As such, it was difficult to fully understand the necessary requirements. The Domain Tech management agreed to accept any user change at any stage and believed it will increase the company reputation, but it was indeed a great mistake. Users could not provide detailed information and lately numerous gaps were identified that needed to be considered. Users added several new things compared to the initial specification based on which schedule and effort estimation was calculated. Numerous changes were requested by the users, such as change of staff profile and lay out and addition of more functionalities under a specific module related to personal and payroll management, budgeting, auditing, and accounting. The collected

Table 2: Identified Risk Factors and Events

Risk factor	**Event**
Numerous change request	Budget overruns
Users passive involvement	Erroneous requirements
Users lack of project domain knowledge	Schedule overruns
User lack of IT competence	Ineffective training
Inadequate training budget	Unclear system vision
Large number of users requiring training	Ineffective communication
High training cost	User dissatisfaction
Incomplete & incorrect initial specification	Poor system use
Over-promise & Inaccurate estimation	Poor reputation
Error in project contract	Incomplete information
Lack of experience of handle government officials	Unable to generalize product
Bureaucracy nature of organization	Poor feedback
User lack of motivation	Deployment problems
Data conversion difficulties	High variation
Unwilling to sign interview document	
Complex interactions among the modules	
Political biasness	

information was sometimes ambiguous due to different interpretation of the same context as well as incomplete information. It was difficult to obtain appointments from the high officials and several meetings were needed to collect relevant information. But their remarks were important for the project acceptance. Some of the users had not adequate knowledge of the software application but played important roles to support the key business process. There were ambiguities in describing the key operational process and roles involved within the process. Once an interview was completed no user agreed to sign the interview documents. At the end, the requirements analysis consumed much more time than estimated.

Another problem appeared concerning the number of users who participated in the training. Initially Domain Tech agreed to train 500 users. However, during the requirements identification and on site observation, there were about 800 users identified who needed training for the new system environment. But the planning ministry and UNDP refused to increase the training budget as total project budget was already approved and there was no room for revision because several donor agencies supported the project fund. Domain Tech also had inadequate experience to handle the government officials. Existing data was documented with dissimilar structure, which made it difficult to convert the manual data into new electronic formats. Risk factors were summarized from the raw list. These factors were mainly due to unavailable, wrong and incomplete information, poor participation, errors from the existing system, and data conversion difficulties which resulted in incomplete specification and inaccurate estimation. Table 2 shows the risk factors.

6.2.4. Activity 4: Assess Risk

The RM team considered interview responses of the selected risk factors as initial input for the risk assessment. Moreover PM's and other members' experience and the pre-

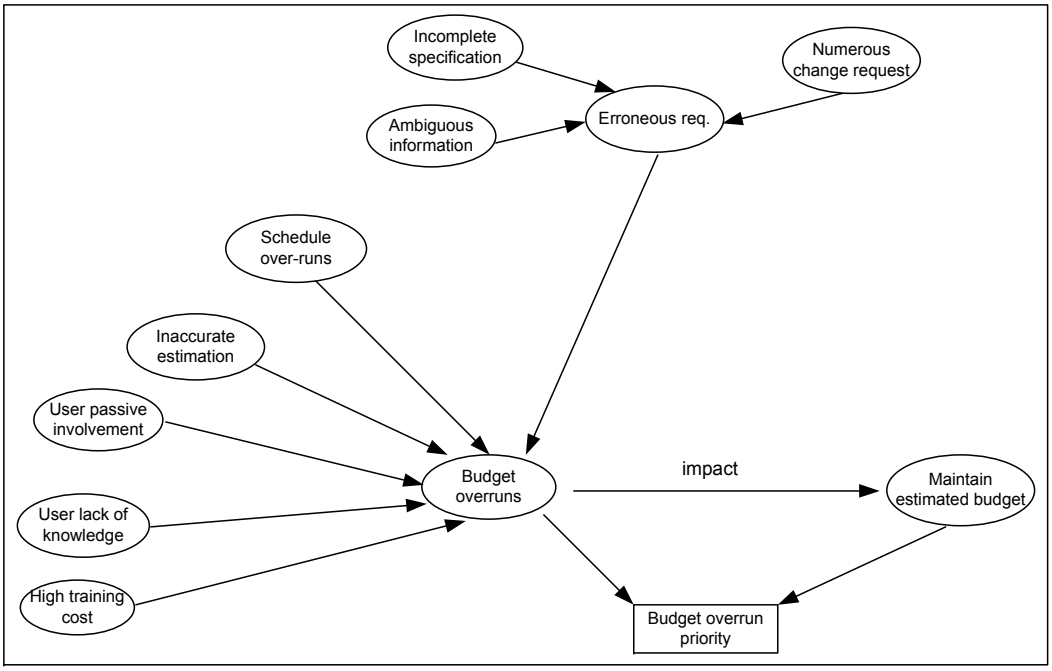

Figure 6: BBN Network for Budget Overruns

assumptions provided by GSRM about the causal link of the risk factors to the risk events and consequence to the goals were also considered for the estimation of the risk event likelihood and risk impact. Table 2 includes risk events caused by the identified risk factors. These events directly obstruct the goals, for example, budget overruns, erroneous requirements, poor training, and system use obstruct the goals like complete project in estimated budget and schedule or vendor high reputation. To construct a causal relationship model, risk factor, event, and priority were considered as target, observable, and decision nodes of BBN. The RM team agreed that budget overrun is the highest prioritized risk factor. Risk factors, such as inaccurate estimation, high training cost, erroneous requirements, and schedule over runs casually link to the budget overruns as shown in Figure 6. These factors are the most influential for the project context and some of them were beyond the control of the project manager and the project environment. The identified factors resulted in complete obstruction of the goal to maintain estimated budget in the development. Erroneous requirements and schedule overruns were considered as risk factors but these factors were again treated as risk events in different context. Finally, risk events were prioritized into the three scales, i.e., very important, important, and less important, so that severe ones get immediate attention. However for confidentiality reasons we cannot provide the detailed results.

6.2.5. Activity 5:Treat and Monitor Risk

The RM team initially planned to control the high and medium prioritized risks by completely eliminating or reducing the likelihood of risk event occurences and severity of the

impacts. The session identified the possible countermeasures and potential ones were selected so that the goals could be attained.

Erroneous requirements were one of the main events which negatively affect the possible outcome of goals such as complete project in estimated budget and schedule, reduce erroneous requirements, improve completeness in requirement specification, quality product, and generalize application. Users' factors were one of the main reasons for the risk events and a large number of users increased the overall total training cost and as a result the overall project budget. But these factors were beyond the PM's controls and difficult to eliminate completely. The RM team identified possible countermeasures: i.e., *train selected representatives from user groups*, *extensive user involvement*, *signed frozen requirements*, *complete understanding of users business processes and dependencies among the modules*, *obtain and incorporate key users feedback*, and *include a J2EE expert into the project*. The identified countermeasures were potential and would not incur additional cost except of the integration of a J2EE expert. However it was difficult to involve the key users even though they were available in the commission campus. And user lack of project domain knowledge and lack of IT competency could not be addressed by any means during the duration of the project. However, selected users training would reduce the overall training cost significantly. The RM team emphasized the professional and competence training and preparation of a comprehensive training manual before physically deploying the system. Still, there was no effective way to address the numerous change requests but the PM decided that once the feedback was integrated then the frozen requirements of a specific module would be signed by the key user. But at that stage, the project experienced a huge number of user change requests and several new things were added compared to the initial specification. The PM agreed that it was not be possible to maintain the estimated budget and schedule and management decided to accept 15% budget overrun. The management decided to keep their reputation high by any means and a complete understanding of the business processes was essential for the project as well as for the future customization. Domain Tech planned to arrange *workshop sessions* with the high government officials to inform them about the status of the project and the necessary further action. These sessions were effective to develop and implement the application into the planning commission campus.

6.2.6. Goal-risk model

Figure 7 illustrates the goal-risk model for the complete project in the estimated budget. The goal was refined into several sub-goals, such as maintain estimated budget and schedule, realistic estimation, clear milestones, user active participation and motivation, and reduce errors from requirements. Risk factors, such as numerous change requests, user passive involvement & lack of knowledge, high training costs, inaccurate estimation, errors in contract, and incomplete specification obstructed the sub-goals and lead to risk events such as erroneous requirements, schedule overruns and budget overruns. The bottom part of Figure 7 shows the treatment actions and the associated agents responsible for implementing the actions. E.g., comprehensive and competence training.

Figure 8 shows the model for obtaining positive reputation. The goal was rather subjective and refined into the user satisfaction, quality product, successful training, and system usage. Risk factors,such as poor training, ineffective communication, incomplete and complex product, and retain existing system obstructed the goal and lead to user dissatisfaction and low reputation as the main risk events. User motivation for the new system, effective cooperation between users and practitioners, complete and competence training are control

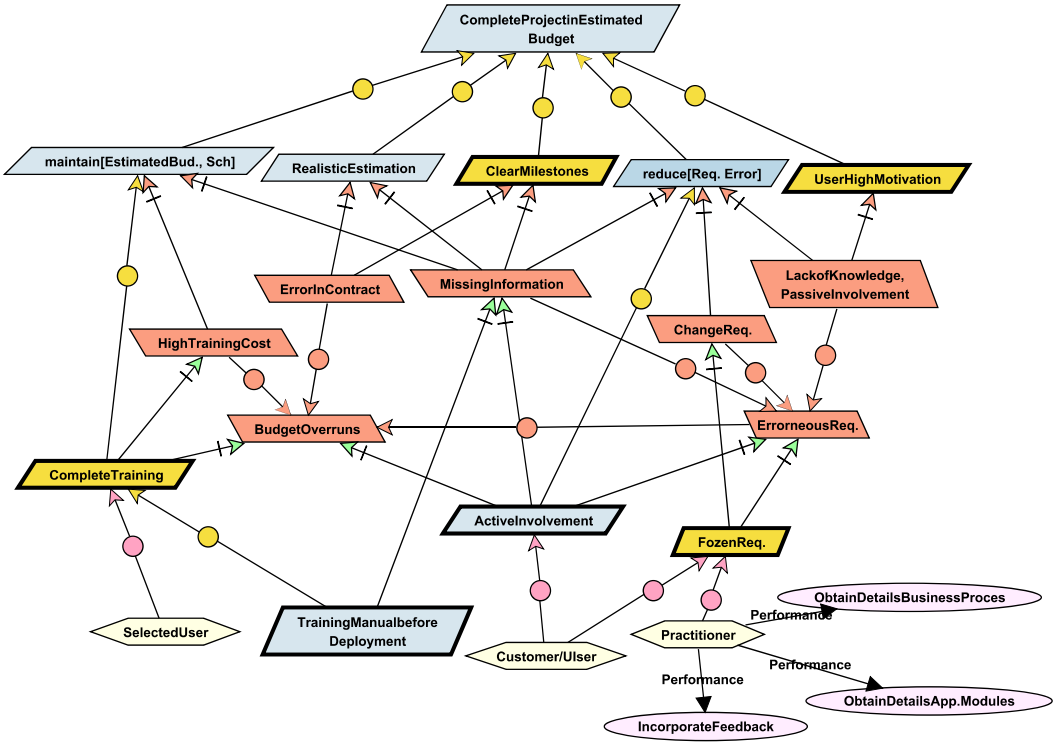

Figure 7: Goal-risk model for project completion

strategies, which can mitigate the risk factors. The Elimination of old manual systems was considered as a requirement within the control action. Project contracts did not cover the maintenance part which seemed to be an important issue for this project. The PM decided to convince the project sponsor to allocate the maintenance budget for the project. Domain Tech management also decided to arrange 2 or 3 common workshop sessions apart from training to motivate the users for the system-to-be deployed.

The difficult part regarding risk management was to convince the key government officials about the implementation of the selected control actions. The PM communicated the agreed control actions to the management and management planned a meeting with the planning commission secretary. The visual representation of the goal-risk model effectively helped to communicate the risk information with the management. In the meeting, the secretary of planning ministry agreed on the recommended actions and assigned 20 key users to be extensively involved during the development to help the development team. The secretary also circulated an office order at joint, deputy and assistant secretary level to collaborate with the project by giving necessary information. Furthermore, only 58 users were selected as champions for the training session conducted by Domain Tech. The PM decided to arrange a risk monitor meeting, initially twice a month and less frequently at the later stages of the development. At the end, student members carried out the interview session with the open

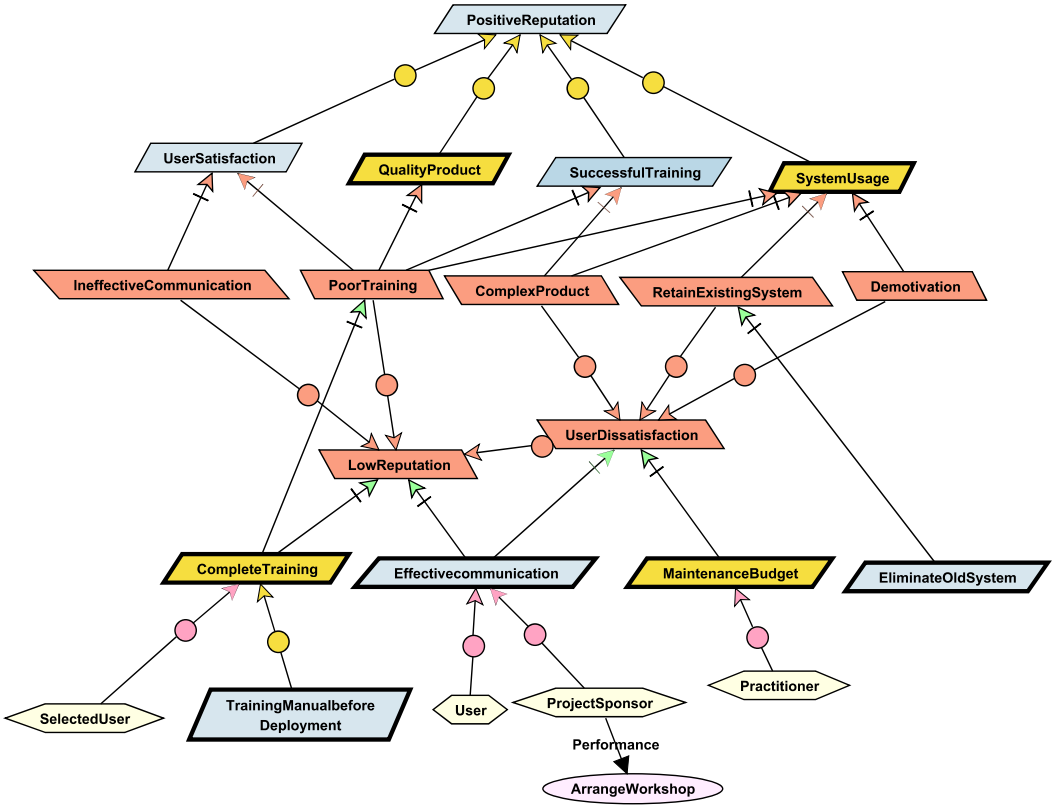

Figure 8: Goal-risk model for positive reputation

questions to obtain feedback about GSRM. The interview participants were the PM and other members of the RM team, three practitioners of the development team, one management representative of Domain Tech and two users.

7. Discussion

In this section, we report on the findings based on the observations from the project. Moreover, we compare our case study findings with other study results to generalize our findings. We also include a summary of the overall impact of GSRM into the software project.

goal_risk_reputation

7.1. Lessons Learned

7.1.1. Usefulness of the GSRM

Goal-driven approach for risk management. A Goal-driven approach for risk management is reasonably beneficial during the early development. A project contains goals. Goals make it easy to communicate with the users, project sponsors, and development team members.

Goal identification and modelling: The main users of the project were government officials who did not have adequate knowledge of the purpose of automation due to lack of IT expertise. However, they were able to demonstrate their expectations from the project. From the user perspective, several goals were identified such as user satisfaction, successful training, and system usage. From the project team perspective, goals were emphasized more on issues related to project completion, training, user participation and reputation. However, some participants observed that goal modelling was difficult to perform and required more experience.

Integration of GSRM into requirements engineering. Based on the open question responses, the PM and practitioners appreciated the integration of risk management into requirements engineering. It provided them with early warnings about the problems that existed in the project.

Artefact oriented view: There were dependencies among the requirements and risks specification artefacts. In particular goals were one of the fundamentals ones to support the integration. For instance, in the studied project, positive reputation was considered one of the high prioritised goals. Several risk factors such as poor training and ineffective communication were directly obstructing the goals. Elicited requirements were considered to identify risk factors, thus requirements were the input for the risk identification. Risk artefacts, such as risk level, priority and list of erroneous requirements directly contributed to reduce errors from the requirements. Therefore both the requirements and risks artefacts were dependent upon each other and supported the integration of two different processes.

Process oriented view: In the project, brainstorming was used to identify the goals, requirements and risk factors. The Project Manager and other members of the risk management team played main roles for risk management and requirements analysis. Furthermore, user active participation observed influential factors for both requirements engineering and risk management. Therefore, roles like project manager, requirements engineering, risk management, and user participation can effectively support activities under requirements engineering and risk management.

GSRM process. The activities and tasks under the GSRM process were considered as systematic and adequate to produce the risk specification artefact.

Activity definition: Based on the open questions response, the activities under the GSRM process were identified as fully operational as the approach provided adequate techniques to identify and analyse goals, risks and treatment actions. The initial activity explicitly allocated schedule and resource for risk management within the project. The composition of the risk management team from different roles was beneficial as different views of the goals and risks were identified and combined. Understanding the riskiness nature of the project at early stages was appreciated by the project manager and other members of the development team as it allowed them to understand the necessity of the risk management activities in the project.

Goals and risks identification and analysis: The GSRM process provides three techniques,(i.e., structured interviews with closed questions, brainstorming sessions, and analysis of project documents) to identify and analyse goals and risks. The information gained from the project brainstorming session was verified during the interview session. The combination of these techniques was treated as being systematic, reasonably applicable and

reducing the bias, in particular, for the risk identification. The component-element-factor hierarchy allowed the categorization of goals and risk factors and guided the structure of the interview.

Artefacts: GSRM provides both textual and graphical representation of artefacts produced by its activities. In particular, visual presentation is provided through goal-risk models and causal relationship models, which made it easy to communicate the risk information with the project team and management as we observed during the studied project. The central textual artefact of GSRM is a risk status report, which is the final output of the activities. Thus the risk status report provides complete information about the goals, risk factors, risk events and treatment actions and status of the risks.

Applicability of GSRM into the studied software project. Two kick-off workshops and five brainstorming sessions, each approximately 4 and 6 hours respectively, and interview sessions (2 hours for the closed questions by the practitioner and 3 hours and 30 minutes by the users and 1 hour and 30 minutes for open questions by the practitioner) were used to perform the activities of the GSRM process. Furthermore, the student members along with the other member of the risk management team analyzed, as input data for the risk management process, the project documents and in particular artefacts, such as the initial specification, the nature of change requests, the detailed requirements specification, the project scope, and the success criteria. They also compiled the interview responses to identify risks and to obtain feedback about GSRM. Among the top 7 risk events and 15 influential risk factors only 3 risk events and 11 risk factors were fully or partially controlled through 8 potential countermeasures. It took approximately four complete working days for the analysis. Goals and risks identification and modelling activities consumed the highest effort. Project complexity and a large number of goals certainly increased the overall risk management effort. A total of approximately 26% effort of initial deliverable phase including the student memebers effort used for risk management. The effort estimation excludes the risk monitor activity but the PM agreed that it was reasonable enough within the first deliverable phase of the project.

Limitations of GSRM. The studied project also allowed us to identify limitations of GSRM. The information about the limitations was collected based on the participants observations about GSRM during the interview with open questions and the feedback from the risk management team. First, when the project focuses on a large number of goals, then efforts for developing and maintaining the artefacts are considered too high. In particular, GSRM needs more effort for analysing the goals and constructing goal-risk and causal relationship model. Secondly, constructing a casual relationship model through BBN is a complex undertaking, specifically when several risk factors are considered as intermediate nodes, which increase the size of the probability density table. Thirdly, from the studied project, the risk monitoring activity was not properly performed at later stages due to the pressure to handle user changes and later deliverables. Finally, based on the studied project observation, it was difficult to answer some of the closed questions, in particular, quantify the state of the factors under the development components into three different scales at an early development stage. Despite of the implementation of GSRM into the studied project, not all identified risks were controlled. Risks related to the control budget and schedule overruns, i.e., requirements changes, user motivation for the system-to-be, and inadequate user knowledge, severely impacted the project. This was because most of the risk factors originated from the

user perspectives, which were beyond the project manager's control. Some risk factors were fully or partially controlled, such as high training cost, overall budget overruns, erroneous requirements, users participation, and detailed system specification. It is therefore always important in a software development project to have the risk factors and/or relevant potential treatments under the control of the project manager and project environment.

7.2. Comparison with Other Study Results

We compare the result of our study with other study results found in the literature, specifically those concerned with software development risk factors and their influence to the project outcomes. Such comparison allows us to generalize our findings and to identify contextual factors from the current study context.

Commonality in risk factors. There is a substantial commonality fully or partially amongst our results and those discussed in existing literature. For instance, Schmidt et al. [23] identified a comprehensive list of risk factors and top factors such as lack of adequate user involvement and cooperation, lack of frozen requirements, change scope, and unclear/misunderstood scope/objective are similar to our findings. The most noticeable areas, which included distinctive risk factors by our study, are user, user's organizational, and requirements context, which match important risk components such as schedule and budget, requirements management, and personnel management demonstrated by Ropponen et al. [30]. Our results also show similarities with the Wallance et al. [53, 24] findings on requirements, user, and complexity risk dimensions. Procaccino et al. [39] study results emphasized the factors related to the customer/user and requirements, such as user involvement, realistic expectation, complete and accurate requirements, and well defined project scope, which highly influences project success. Linberg [54] also summarized factors such as effective leaders, technologically realistic requirements, and realistic schedule and effort estimation relevant to the project success. Keil [55] identified a changing project scope and the lack of frozen requirements as critical risk factors of IS projects. Our results fully or partially match with these findings.

Risk management barriers. In the case study, the PM realized the need for risk management and he was the main authority for the task. However, practitioners were not motivated to implement a complete risk management process. A similar situation is observed by other study results. this situatio is treated as the main barrier that obstructs the implementation of a formal risk management practice. Our interview responses summarized a large number of risk factors from the project. A recent survey study result shows, that intangible benefit, lack of resource, and too many risks to control are the main perceived barriers by the experienced project manager to a successful implementation of software risk management [9]. Nyfjord et al. [10] and Kwak et al. [32] emphasize the organizational problems, such as variation of risk perception by different roles, lack of competence and process problems, such as lack of plan and coordination are additional barriers besides resource problems to integrate risk management into development. We observed that formal risk management practice was missing in the Domain Tech case study. However, implementation of GSRM and its result certainly advocate for a formal risk management practice in the upcoming projects.

Factors related to current study context. It is worth mentioning that some important factors derived from case studies found in the literature, were not applicable to the Domain Tech case study. These included no planning or inadequate planning, lack of management support, problems related to the development process, and development team. Moreover, several risk factors, seem to have a partial or no match compared to the other studies, such as user unwillingness to provide information and sign the interview document, bureaucracy nature of the organization, political biasness, user lack of IT knowledge, numerous change/update requests, errors in contracts, training for large number of users, and over promise made by the vendor. These factors dominate our case and mostly originate from the user context.

7.3. Overall Observation

Our experience indicates that GSRM is reasonably suitable for any complex project. The studied project was complex, where GSRM was well-integrated to identify and control the software development risks. The project manager agreed that by using GSRM it was possible to identify and tackle the problems in a structured way from the beginning of the development.

Analyzing the observations from the studied project helps us to better integrate our goal-driven risk management method into requirements engineering. The process and artefact oriented views allow explicitly integration of risk management activities into the requirements engineering phase. In general a project does not have a risk manager except in large or complex cases. We have seen that a project manager or requirements engineer having basic knowledge of risk management are able to perform the risk management activities. A tutorial session about goal modelling and risk management helps to understand the basic concepts of GSRM. Our observation is that when treatment of risk factors is beyond the control of the project manager and development environment then it is difficult to control the risk. The studied project failed to be completed within the estimated budget and schedule. The project suffered approximately 25% budget overruns even though some of the risk factors were mitigated. the risk management team agreed on 15% acceptable limits for budget overruns, which was also exceeded. The project users were government officials and the project was funded by UNDP, therefore there was no room for budget increase. The Domain Tech management considered the project as a loss project. But Domain Tech obtained the next work order from the planning ministry, which implied that the government officials were happy with the final outputs from the project. The project manager believed that without integration of risk management, budget overruns could be even more, government officials would not consider to significantly participate in the project and the requirement errors would not be contollable from the early stage. The project is planned to be deployed in ministry campus and selected users are currently under training. Domain Tech already obtained the phase 2 of the project, which includes three more modules in the existing application software. Therefore, one goal is attained i.e., obtain good reputation. The upcoming project concerns the annual development program, project and foreign aid monitor systems of the planning ministry, and maintenance. The Domain Tech management hopes that phase 2 will compensate the loss suffered by the developed project.

One further observation is that it is not necessary to always consider budget and schedule factors at the highest priority. Software development projects are complex undertakings. There exist other issues such as requirements, users, change management, user satisfaction, and system usage, which directly or indirectly influence the budget and schedule constraints. These factors need early attention in the development. Early determination

of the nature of project riskiness is very effective to plan the risk management activities. Our studied concluded that the project scope affects all dimensions of risk but for high risk projects, risks associated to requirements specification, users, change management, project execution are more obvious.

8. Study Validity

Case studies are generally criticized for being biased, less valuable, or unable to generalize the findings [56]. Threats to case studies are related to the difficulties of collecting reliable results and on generalising the findings. We systematically planned the study, considered study goals, and addressed the issues related to the threat from the beginning of the study. Data was collected and analyzed in a consistent way from multiple sources to overcome the study validity threats.

8.1. Internal Validity

We tried to reduce the expectation bias on the case study result. The interview responses were commonly analyzed in the brainstorming session by the risk management team. None of the principle investigators of GSRM and related research were directly involved in the case study, in order to reduce the bias of the findings. Data was collected not only from the development team but also from the customer and project sponsor representatives. Furthermore, project documents were also analyzed to understand the goals and risk factors. Therefore, several sources were used to collect data and this limits the effects of the interpretation of one single data source. Interview responses and results of the project documents inspection were further analyzed and discussed in the brainstorming sessions. The PM and other team members were not fully motivated for a formal risk management practice in the project. This attitude changed later in the project by observing the outcome of GSRM activities. This demonstartes the importance of GSRM and the fact that it assisted in quickly creating visual and critical insights. Maturation effects also intimidate the internal validity, in particular, when the participants react differently about the perception of risk during the course of development. However, in our work activities of GSRM took place at the requirement engineering phase, and therefore that threat can not significantly affect our study.

8.2. External Validity

The case study context is located in a single geographical region, and as such there is possibility for cultural bias. Our findings are from multiple data sources which allow for stronger conclusions. The study results are compared with other results from the literature in order to generalize our findings. The comparisons confirmed several commonalities in terms of goals and risk factors, however there are also some unique factors from the local context. Therefore, factors related to goals and risks are influenced by the local context.

8.3. Construct Validity

We considered benefits and limitations of the GSRM and correctly measured quantitative metrics, such as effort required to undertake risk management activities into software project. The quality of the case study questionnaires was improved by following the feedback from our previous study. The participants answered most of the questions apart from some which are observed as difficult at an early stage. The threat of the questions being the wrong

ones to ask was mitigated. The project manager and the development team members had adequate experience on several software development projects. User representatives also had adequate experience in the government service. The student members conducted a kick-off workshop for the practitioners and explained details about the interview purpose to the users. We believe that the participants understood the terms being used.

9. Conclusion

In this paper, we presented the GSRM method and reported our experience from its usefulness through an empirical evaluation by combining a case study with action research. Our results show that a goal-driven approach is suitable for risk management and risk management is well integrated into the requirements engineering. The results indicate that GSRM is a practical and reasonable risk management method that can be employed in an industrial context. The goal oriented view made easy to understand and communicate the concepts of GSRM and a tutorial session was adequate for this purpose. We have noted our experience and insight gained and lessons learned from the case study. To generalize our result we compared the study results with similar study results. We believe this study is useful to improve GSRM and to inform and motivate practitioners about the effectiveness of integrating risk management activities from the early phase of the development. However developing more case studies is necessary so that the validity of the results can be more generalized and the risk management method can be further improved based on the case study observation. We are currently working on defining comprehensive application guidelines for the GSRM, so that it can provide better support for risk analysis into a software project. We are also planning to develop a goal-risk taxonomy. A goal-risk taxonomy identifies and classifies goals and risks that are associated with certain project characteristics and links with possible potential control actions in controlling specific types of risks. Project managers can reuse the taxonomy for any upcoming project. Some factors within the course of the product life cycle such as deployment, operation, and maintenance do not get adequate attention at an early stage. But later these factors can cause a real danger for the project. These factors need further investigation. We would like to integrate continuous risk management activities for analysing these factors.

Appendix: Sample Set of Closed Questions

Project Planning, Control and project scope

(Q1) Are all project deliverables and related schedule realistically estimated & agreed with the customer?

☐ No ☐ Not fully realistic ☐ Estimated & agreed

(Q2) How frequent are both estimated schedule and cost planned to be revised?

☐ Never ☐ End of every deliverable ☐ Monthly or more frequent basis

(Q3) Are the essential technical issues (e.g. tools, language & hardware) identified, available and familiar for the project?

☐ Some ☐ More than some ☐ Yes all of them

(Q4) Is the project success criteria realistic and achievable?

☐ Partially ☐ Yes but not all ☐ Yes

(Q5) How risky is the project by considering scope, users, complexity, constraints, reusability, specification, resource, expertise and knowledge?

☐ High ☐ Medium ☐ Low

Development Process

(Q6) Does the development process support adequate activities, steps and methods?

☐ Partially ☐ More than partially ☐ Completely

(Q7) Are the current development processes suitable and adequate for the project?

☐ Partially ☐ More than partially ☐ Completely

(Q8) Is there any coding standard followed for the project?

☐ No ☐ Partially ☐ Yes

Specification

(Q9) Are the business goals and objectives identified for the project?

☐ Partially ☐ More than partially ☐ Completely

(Q10) Does the project scope clearly support the business needs and add business value to the overall customer's business environment?

☐ No ☐ Partially but not sure ☐ Yes

Requirements Faults

(Q11) Do the requirements provide different ambiguous interpretations or lack of support for rationale?

☐ Highly ☐ Partially ☐ Rarely

(Q12) Are requirements traceable to its source such as business specification, project scope and user expectation?

☐ No ☐ Some of them ☐ All

(Q13) Are the right customer or user representatives involved in the RE process?

☐ No ☐ Some ☐ Yes

(Q14) Do you follow any standard (template, notations and checklist) for the specification?

☐ No ☐ Partially ☐ Yes

Quality and Testing

(Q15) Is quality assurance and management planned and implemented for the project?
☐ No ☐ Partially ☐ Completely

(Q16) Is the overall product performance (i.e., throughput, real time response, recovery time and access) identified, quantified and agreed with the user?
☐ Partially ☐ More than partially ☐ Yes

(Q17) Does the project consider a complete user manual with the software?
☐ Incomplete user manual ☐ Un-verified user manual ☐ User manual will be developed, tested & delivered

Operation

(Q18) Do the users maintain any documented process to implement and deploy the software into their premises?
☐ No ☐ Partially ☐ Yes

(Q19) Are there any acceptance criteria for the system operation?
☐ No ☐ Yes but not clearly ☐ Yes

(Q20) Will the technical or operational documentation and supporting training manual planned to be available before the actual training takes place?
☐ No ☐ Not sure ☐ Yes

(Q21) Do there any checklist plan to assess the proper implementation of the system within the user's operating environment?
☐ No ☐ Partially ☐ Yes

Maintenance

(Q22) Does the project scope include maintenance?
☐ Yes ☐ Partially ☐ No

(Q23) Will the overall complexity of the software be increased (i.e., more complex structure of individual component, or several new components) due to the maintenance?
☐ Yes ☐ Not sure ☐ No

(Q24) Will the project document and user manual be updated once the maintenance operation is being completed?
☐ No ☐ Partially ☐ Yes

Motivation & Domain Knowledge

(Q25) How is the overall relevant domain knowledge of the development team?
☐ Not much ☐ Less than adequate ☐ Adequate

(Q26) What is the level of technical expertise between the development team and user?
☐ Lower by development team ☐ Almost same ☐ Higher by development team

Customer/user

(Q27) What is the level of involvement of customer / user until now?
☐ Passive ☐ Occasional ☐ Active

(Q28) What is the level of confidence of customer/user over the development team?
☐ Low ☐ Medium ☐ High

(Q29) Do you think customers/ users have realistic expectations about the project?
☐ No ☐ Partially ☐ Yes

Management

(Q30) How much capable is the project manager to lead the project successfully?
☐ Unreliable ☐ Partially reliable ☐ Reliable

(Q31) Does the management support the continual improvement of the development processes and development facilities?
☐ No ☐ Less frequently ☐ Frequently

(Q32) Does the management always try to achieve customer satisfaction goal?
☐ No ☐ Not always ☐ Yes always

Organizational Stability

(Q33) Does the management agree to trade additional budget and adequate support for handling project risks?
☐ No ☐ Not willingly ☐ Yes

(Q34) Is the organization always change its operational environment?
☐ Yes ☐ Change in certain interval ☐ Yes but for effective output

(Q35) Does the organization support the project?
☐ No ☐ Partially ☐ Yes

(Q36) Is the organization highly politically biased ?
☐ Yes ☐ Medium ☐ No

References

[1] B. Boehm, A. Egyed, J. Kwan, D. Port, A. Shah, R. Madachy, Using the winwin spiral model: A case study, Computer 31 (7) (1998) 33–44. doi:http://dx.doi.org/10.1109/2.689675.

[2] D. W. Karolak, Software Engineering Risk Management, IEEE Computer Society Press, 1995.

[3] S.-W. Foo, A. Muruganantham, Software risk assessment model, in: Proceedings of the IEEE International Conference on Management of Innovation and Technology (ICMIT 2000), 2000.

[4] J. Kontio, Software engineering risk management: A method, improvement framework and empirical evaluation, Ph.D. thesis, Helsinki University of Technology (2001).

[5] G. Roy, A risk management framework for software engineering practice, in: ASWEC '04: Proceedings of the 2004 Australian Software Engineering Conference, IEEE Computer Society, Washington, DC, USA, 2004, p. 60.

[6] S. Islam, Software development risk management model-a-goal-driven approach, Ph.D. thesis, Institute für Informatik, Technische Universität München, Germany (2011).

[7] R. L. Glass, Software Runaways: Monumental Software Disasters, Prentice-Hall, 1998.

[8] A. van Lamsweerde, Requirements Engineering: From System Goals to UML Models to Software Specifications, Wiley, 2009.

[9] E. E. Odzaly, P. D. Greer, Software risk management barriers: An empirical study, in: ESEM '09: Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement, IEEE Computer Society, Washington, DC, USA, 2009, pp. 418–421. doi:http://dx.doi.org/10.1109/ESEM.2009.5316014.

[10] J. Nyfjord, M. Kajko-Mattsson, Integrating risk management with software development: State of practice, in: Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 (IMECS), 2008.

[11] J. Ropponen, Risk assessment and management practices in software development, in: In: Willcocks, L.P., Lester, S. (Eds.), Beyond the IT Productivity Paradox;john Wiley & Sons, Chichester. pp. 247-266., 1999.

[12] P. L. Bannerman, Risk and risk management in software projects: A reassessment, The Journal of Systems and Software 81 (12) (2008) 2118–2133. doi:http://dx.doi.org/10.1016/j.jss.2008.03.059.

[13] B. W. Boehm, Software risk management: Principles and practices, IEEE Software 8 (1) (1991) 32–41. doi:http://dx.doi.org/10.1109/52.62930.

[14] F. Sisti, S. Joseph, Software risk evaluation method version 1.0., Tech. rep., SEI,Carnegie -Mellon University (1994).

[15] C. J. Alberts, A. J. Dorofee, R. Higuera, R. L. Murphy, J. A. Walker, R. C. Williams, Continuous Risk Management Guidebook, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA., 1996.

[16] M. Carr, S. Konda, I. Monarch, C. Ulrich, C. Walker, Taxonomy based risk identification (cmu/sei-93-tr-6, ada266992), Tech. rep., Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA. (1993).

[17] M. K. M.B. Chrissis, S. Shrum, CMMI Guidlines for Process Integration and Product Improvement, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003.

[18] ISO/IEC 15504 information technology – process assessment, International Organization for Standardization (ISO) /International Electrotechnical Commission (IEC).

[19] ISO 31000:2009 Risk Management Ů Principles and Guidelines, International Organization for Standardization (ISO) /International Electrotechnical Commission (IEC).

[20] Standard australia, as/nzs 4360:1999 risk management (1999).

[21] R. E. R. Prikladnicki, M. H. Y. J. L. N. Audy, Risk management in distributed it projects: Integrating strategic, tactical, and operational levels, International Journal of e-Collaboration 2 (2006) 1–18.

[22] H. Barki, , S. Rivard, J. Talbot, Toward an assessment of software development risk, Journal of Management Information Systems 10 (2) (1993) 203–225.

[23] R. Schmidt, K. Lyytinen, M. Keil, P. Cule, Identifying software project risks: An international delphi study, Journal of Management Information Systems 17 (4) (2001) 5–36.

[24] L. Wallace, M. Keil, A. Rai, Understanding software project risk: a cluster analysis, Information and Management 42 (1) (2004) 115–125.

[25] N. H. Arshad, A. Mohamed, Z. M. Nor, Risk factors in software development projects, in: SEPADS'07: Proceedings of the 6th WSEAS International Conference on Software Engineering, Parallel and Distributed Systems, World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA, 2007, pp. 51–56.

[26] R. T. Nakatsu, C. Iacovou, A comparative study of important risk factors involved in offshore and domestic outsourcing of software development projects: A two-panel delphi study, Information and Management 46 (1) (2009) 57–68. doi:http://dx.doi.org/10.1016/j.im.2008.11.005.

[27] T. Moynihan, How experienced project managers assess risk, IEEE Software 14 (3) (1997) 35–41. doi:http://dx.doi.org/10.1109/52.589229.

[28] C. L. Iacovou, R. Nakatsu, A risk profile of offshore-outsourced development projects, Communication of the ACM 51 (6) (2008) 89–94. doi:http://doi.acm.org/10.1145/1349026.1349044.
[29] W. Asprayn, F. Mayadas, M. Y. Vardi, Globalization and offshoring of software: A report of the acm job migration task force., Tech. rep., ACM, NY (2006).
[30] J. Ropponen, K. Lyytinen, Components of software development risk: How to address them? a project manager survey, IEEE Transactions on Software Engineering 26 (2) (2000) 98–112. doi:http://dx.doi.org/10.1109/32.841112.
[31] J. Jiang, G. Klein, Software development risks to project effectiveness, The Journal of Systems and Software 52 (1) (2000) 3–10. doi:http://dx.doi.org/10.1016/S0164-1212(99)00128-4.
[32] Y. H. Kwak, J. Stoddard, Project risk management: lessons learned from software development environment, Technovation 24 (11) (2004) 915 – 920. doi:DOI: 10.1016/S0166-4972(03)00033-6.
URL http://www.sciencedirect.com/science/article/B6V8B-487656Y-1/2/b745c965f078f80208d21435690db82a
[33] S. L. Pfleeger, Risky business: what have we yet to learn about risk management, The Journal of Systems and Software 53 (3) (2000) 265–273. doi:http://dx.doi.org/10.1016/S0164-1212(00)00017-0.
[34] S. Islam, Software development risk management model: a goal driven approach, in: ESEC/FSE Doctoral Symposium '09: Proceedings of the doctoral symposium for ESEC/FSE on Doctoral symposium, ACM, New York, NY, USA, 2009, pp. 5–8. doi:http://doi.acm.org/10.1145/1595782.1595785.
[35] S. Islam, S. H. Houmb, D. Mendez-Fernandez, M. M. A. Joarder, Offshore-outsourced software development risk management model, in: In Proc. of the 12th IEEE International Conference on Computer and Information Technology (ICCIT 2009), Dhaka , Bangladesh, DOI: 10.1109/ICCIT.2009.5407292, 2009, pp. 514–519.
[36] S. Islam, S. H. Houmb, Integrating risk management activities into requirements engineering, in: Proc. of the 4th IEEE Research International Conference on Research Challenges in Information Science(RCIS2010), Nice, France, 2010.
[37] S. McConnell, Rapid Development , Taming wild software schedules, Microsoft Press, 1996.
[38] T. Saarinen, An expanded instrument for evaluating information system success, Information and Management 31 (2) (1996) 103–118. doi:http://dx.doi.org/10.1016/S0378-7206(96)01075-0.
[39] J. D. Procaccino, J. M. Verner, S. P. Overmyer, M. E. Darter, Case study: factors for early prediction of software development success, Information and Software Technology 44 (1) (2002) 53 – 62.
[40] B. Schätz, A. Pretschner, F. Huber, J. Philipps, Model-based development of embedded systems, in: OOIS '02: Proceedings of the Workshops on Advances in Object-Oriented Information Systems, Springer-Verlag, London, UK, 2002, pp. 298–312.
[41] E. Geisberger, M. Broy, B. Berenbach, J. Kazmeier, D. Paulish, A. Rudorfer, Requirements engineering reference model (rem), Technical report, Technische Universität Müunchen (2006).
URL http://www.in.tum.de/forschung/pub/reports/2006/TUM-I0618.pdf.gz
[42] D. M. Fernández, M. Kuhrmann, Artefact-based Requirements Engineering and its Integration into a Process Framework, Forschungsbericht TUM-I0929, Technische Universität München (nov 2009).
[43] D. Firesmith, Requirements engineering tasks, Journal of Object Technology 5 (8).
[44] M. Broy, A. Fleischmann, S. Islam, L. Kof, C. Leuxner, K. Lochmann, D. Mendez-Fernandez, B. Penzenstadler, W. Sitou, S. Winter, Towards an integrated approach to requirement engineering, Tech. rep., Technical Report, TUM-I0935,Technische Universität München (December 2009).
[45] B. Schätz, Model-based development of software systems: From models to tools., habilitation, Technische Universität München, 2008.
[46] D. M. Fernández, B. Penzenstadler, M. Kuhrmann, M. Broy, A meta model for artefact-orientation: Fundamentals and lessons learned in requirements engineering, in: MoDELS (2), 2010, pp. 183–197.
[47] B. Freimut, S. Hartkopf, P. Kaiser, J. Kontio, W. Kobitzsch, An industrial case study of implementing software risk management, SIGSOFT Software Engineering Notes 26 (5) (2001) 277–287. doi:http://doi.acm.org/10.1145/503271.503247.
[48] C. Chittister, Y. Haimes, Risk associated with software development: A holistic framework for assessment and management, IEEE Transactions on Systems, Man and Cybernetics 23.
[49] G. G. J. Kontio, D. Landes, Experiences in improving risk management processes using the concepts of the riskit method, SIGSOFT Software Engineering Notes 23 (6) (1998) 163–174. doi:http://doi.acm.org/10.1145/291252.288301.
[50] S. Easterbrook, J. Singer, M. Storey, D. Damian, Selecting Empirical Methods for Software Engineering Research, Springer, 2007.
URL http://www.cs.toronto.edu/~sme/papers/2007/SelectingEmpiricalMethods.pdf
[51] Domain software technologies ltd, http://www.domtech.co.uk/ (last access October 2010).
[52] Digital Bangladesh Vision 2021, http://www.boi.gov.bd (last access October 2010).
[53] L. Wallace, M. Keil, Software project risks and their effect on outcomes, Communications of the ACM 47 (4)

(2004) 68–73. doi:http://doi.acm.org/10.1145/975817.975819.
[54] K. R. Linberg, Software developer perceptions about software project failure: a case study, The Journal of Systems and Software 49 (2-3) (1999) 177–192. doi:http://dx.doi.org/10.1016/S0164-1212(99)00094-1.
[55] M. Keil, P. E. Cule, K. Lyytinen, R. C. Schmidt, A framework for identifying software project risks, Communications of the ACM 41, issues 11 (11) (1998) 76–83. doi:http://doi.acm.org/10.1145/287831.287843.
[56] P. Runeson, M. Höst, Guidelines for conducting and reporting case study research in software engineering, Empirical Software Engineering 14 (2) (2009) 131–164. doi:http://dx.doi.org/10.1007/s10664-008-9102-8.