The Hong Kong Polytechnic University

Department of Computing

A Risk Management Methodology with Risk Dependencies

KWAN Tak Wah

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

August 2009

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

______________________________ (Signed)

KWAN Tak Wah (Name of student)

Abstract

Due to the dynamic changes of business environments and the advancements of technologies, information technology (IT) projects are facing lots of challenges, and there is a need of applying systematic approaches to deal with the risks to ensure the project's success. A common characteristic of current risk management approaches is that they consider risks as independent events. In fact, risks are not always independent. As current practices do not clearly manage dependencies between risks, project managers may inappropriately estimate risks and thereby leave risk effectively unmanaged. We believe that explicitly identifying and managing risk dependencies would be important in both initial and ongoing risk analysis and prioritization, and help to develop better risk management strategies and make more effective risk planning decisions.

This research formally models the risk dependency and proposes a management methodology to address risk dependencies. The essence of this effort is that we propose methods to re-estimate each identified risk by taking account of risk dependency effects, and we enhance a set of risk management practices to manage the re-estimated risk (named *Posterior Risk*). As the risk dependency effects can either increase (i.e. *non-favorable* effect) or reduce (i.e. *favorable* effect) the probabilities of those affected risks, we further propose a set of novel practices to evaluate, react, monitor and control the risk dependencies. In addition, we develop a set of metrics to measure the risk levels from both project and program perspectives with due considerations of the dependencies between risks. From the case studies of three IT projects, we confirm that risk dependencies do exist in projects and programs, and can be identified and systematically managed. We also observed that, as project teams needed to deal with risk dependency issues, communications between projects were improved, and there were synergetic effects in managing risks and risk dependencies among projects.

List of Publications

Journal Paper

1. Kwan, T.W. and Leung, H.K.N. (2009), "A Risk Management Methodology for Project Risk Dependencies", revision resubmitted to *IEEE Transactions on Software Engineering*

Conference Paper

1. Kwan, T.W. and Leung, H.K.N. (2009), "Measuring Risks within a Program Consisting of Multiple Interdependent Projects", *Proceedings of the International Conference on Computational Intelligence and Software Engineering*, Wuhan, China, Dec 11-13

2. Kwan, T.W. and Leung, H.K.N. (2009), "Estimating Project Risk Dependencies", *Proceedings of the 13th IASTED International Conference on Software Engineering and Applications*, Cambridge, USA, Nov 2-4

3. Kwan, T.W. and Leung, H.K.N. (2008), "Improving Risk Management Practices for IT Projects", *Proceedings of the 3rd IASTED International Conference on Advances in Computer Science and Technology*, Phuket, Thailand, Apr 2-4, pp.443-448

4. Kwan, T.W. and Leung, H.K.N. (2005), "An Enhanced Risk Taxonomy for Information Technology Projects", *Proceedings of the 9th IASTED International Conference on Software Engineering and Applications*, Phoenix, USA, Nov 14-16, pp.285-291

5. Kwan, T.W. and Leung, H.K.N. (2004), "Project Perspective of Software Acquisition Practices", *Proceedings of IASTED International Conference on Software Engineering and Applications*, Cambridge, USA, Nov 9-11, pp. 456-464

Acknowledgements

I wish to express my sincere gratitude to my supervisor Dr. Hareton Leung, who has provided me many valuable advice and guidance in my studies.

Most importantly, I would also like to express my deepest appreciation to all my family members for their continuous support, encouragement, patience and understanding through all the years.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Background

With the dynamic changes of business environments, Information Technology (IT) projects are facing many challenges and uncertainty factors. In addition to the advancements of technologies, IT projects are evolving to be more sophisticated in complexity. Managing such projects has to deal with various aspects resulting from the integration of software, technologies, infrastructures, and business processes. On the other hand, there is an increasing demand from the industry to effectively manage IT projects since project success becomes critical to the success of organizations (Lientz and Rea, 2001).

According to the latest studies from Standish Group (Standish, 2009), the success rates of IT projects were still low. Many project failures were caused by the poor project management, such as planning, estimating, scheduling and controlling; but there were also increasing number of failures due to other factors, such as poor morale, no employee commitment, no functional management commitment, poor productivity and poor stakeholder relations (Kerzner, 2004). Another key factor is failure to manage risks. Sherer (2004) pointed out that most of the failed projects were due to the inadequate identification and management of project risks.

As IT projects are becoming increasingly complicated and important, there is a need of applying systematic approaches to deal with the risks in order to ensure IT project's success. Improving the adoption of risk management practices in organizations can help to reduce the number of project failures.

1.2 Motivation and Objectives

As no IT project can ever be risk-free, many methodologies have been applied to quantify the likelihood and estimate the impact of risks that a project may encounter, say in schedule, scope, or resources. This is a business-critical issue (Lientz and Rea, 2001) as it is possible for IT projects to fail completely due to inadequate identification and management of project risks (Sherer, 2004). An effective evaluation process can indeed help to identify different sources of risk, so that appropriate risk management strategies can be developed and specific responsibilities are allocated to appropriate managers in the organization (Sherer, 2004). Although many studies have focused on project risk identifications, and a number of risk management processes and guidelines have also developed, risk management is still the least mature among all knowledge areas of project management (Grant and Pennypacker, 2006).

A common practice of current risk management approaches is to consider risks as independent events. This, however, is counter-intuitive as it is more likely that risk in one area (say, schedule) would impact risk in another area (say, costs) and we believe that the explicit identification of risk dependencies would be important in both initial and ongoing risk analysis and prioritization. Therefore, effectively managing project risk dependencies is one of the improvement areas of project risk management (Kwan and Leung, 2007).

The risk management processes and practices currently adopted in the industry basically followed the risk management paradigm developed by Software Engineering Institute (Van Scoy, 1992). The paradigm is an elaboration of the classic “plan-do-check-act” cycle and specifies a set of cyclic steps (i.e. Identify, Analyze, Plan, Track and Control) throughout a project (Williams *et al*., 1997). It emphasizes the risk management as a continuous process, in which each risk goes through these steps sequentially and independently. However, the current approaches of managing project risks are dangerously simplistic in that, because they do not explicitly address the dependencies between risks, they may inadequately evaluate and prioritize risks, and subsequently select improper risk response strategies, and thereby leave the risk effectively unmanaged. There may be a number of reasons why current risk management practices do not take risk dependencies into account but the primary reason is probably that there have not been, to date, any effective, accurate, and relatively simple means to estimate or evaluate risk dependencies.

In view of the limitations of current risk management approaches, the main objective of this research is to develop a practical risk management methodology, which helps to effectively manage risk dependencies and their effects in projects and programs (a group of related projects). To achieve this objective, our studies focus on the following four major areas:

1. Modeling the risk dependency. A number of novel concepts related to risk dependency are presented.
2. Enhancing project risk management practices. By applying the risk dependency concept, the existing project risk management practices are modified and additional practices are also proposed.
3. Measuring risk from both project and program perspectives. With due consideration of the risk dependencies, new metrics are developed to measure the project and program risks.
4. Verifying whether the enhanced practices and associated measures can be applied in real-life IT projects.

1.3 Contributions

The contributions of this study are: (1) formally define the risk dependency and (2) propose an improved risk management methodology that follows the basic steps of SEI paradigm and enhances the project risk management practices to address the dependency issues. The essence of the enhancements is that we propose methods to re-estimate each identified risk by taking account of risk dependency effects. The re-estimated risk, which is called *Posterior Risk*, is then managed under the enhanced practices. As the risk dependency effects can either increase (i.e. *non-favorable* effect) or reduce (i.e. *favorable* effect) the probabilities of those affected risks, we further propose a set of novel practices to evaluate, react, monitor and control the risk dependencies. In addition, we develop a set of metrics to measure the risk levels from both project and program perspectives. In this thesis, we also present the results from applying the enhanced practices into a program with three IT projects. From the case studies, we confirm that risk dependencies do exist in projects and can be identified and systematically managed. We observed that, as project teams needed to deal with risk dependency issues, communications between projects were improved and there were synergetic effects in managing risks and risk dependencies among projects.

1.4 Thesis Organization

The remainder of the thesis is organized as follows. Chapter 2 summarizes the common definitions of risk and associate risk management practices and measures. Chapter 3 formally defines the risk dependency in relation to project risks, and

Chapter 4 presents the risk management methodology for coping with the existence of risk dependency. Chapter 5 provides the case studies in which the risk dependency concept was applied to IT projects, and discusses the difficulties we encountered and lessons learned. Chapter 6 offers our conclusion and an outline of future research.

Chapter 2 Literature Review

In this chapter, we will focus on the risk management practices for software engineering. We first present the relationship between program and project, in particular their objectives and the risks affecting the objectives. Afterwards, we will review the current practices of managing risks from project perspectives. As we will develop a set of metrics for risk and risk dependency in Chapter 4, this chapter will also review the related works of risk measures. Lastly, we review the modeling techniques that are used to analyze the dependency relationships between events.

2.1 Relationship of Program and Project

The Project Management Institute (2008a) defines that a project is “a temporary endeavor undertaken to create a unique product, service or result”. Each project is usually managed by a project manager. However, for multiple related projects with reasonable size and duration, in addition to the basic project management structure (that is each individual project managed by its respective project manager), these projects are often led by a program manager to consolidate management and reporting (Letavec, 2006). PMI (2008b) defines that a program is “a group of related projects managed in a coordinated way to obtain benefits and control not available from managing them individually”. Figure 2.1 illustrates the relationship between a program and its constituent projects, showing multiple interdependent projects managed collaboratively within a program (Milosevic *et al.*, 2007). Given the tight relationship between program and projects, a risk in a project may affect other risks in the program and other associated projects, or vice versa.

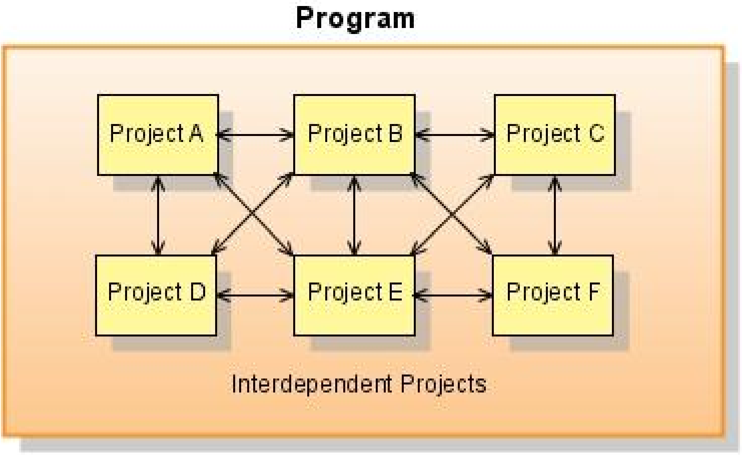


Figure 2.1: Multiple Interdependent Projects in a Program

2.1.1 Program and Project Management Objectives

Typical project management practices cannot help to guide the mission and strategy formulation of projects and cannot manage interrelated component projects as program management practices do (PMCC, 2002); however, jointly managing multiple interdependent related projects in a program can assist an organization to achieve a common goal. Table 2.1 summarizes the key characteristics of program management differing from project management (PMI, 2008a; PMI, 2008b; Reiss, 1996; Martinelli and Waddell, 2004).

Table 2.1: Project and Program Management

	Project Management	**Program Management**
Objective	- Aligned to the goals and objectives of a program	- Aligned to strategic objectives of business
Success Measure	- Budget compliance - Timeliness - Quality of products/services delivered	- Return on investment - Degree of benefits achieved

Management Focus	- Single project - Tasks and the works of producing the project deliverables - Effectively executing processes on a project - Minimizing demand for resources	- Overall leadership of related projects - Coordination and conflict resolution of related projects - Ensuring consistent use of common processes across projects - Maximizing utilization of resources
Knowledge Areas	- Integration management - Scope management - Time management - Cost management - Quality management - Human resource management - Communications management - Risk management - Procurement management	- Integration management - Scope management - Time management - Cost management - Quality management - Human resource management - Communication management - Risk management - Procurement management - Financial management - Stakeholder management - Governance management

In fact, program management can provide a platform that helps to close the gap between business strategies of an organization and the objectives of its related projects. PMI (2008b) defines program management as “the centralized coordinated management of a program to achieve the program’s strategic benefits and objectives”. By applying the program management practices, organizations have gained competitive advantages in resolving many business issues that cannot be addressed by managing projects individually; these issues include the alignment between business objectives and project objectives (Martinelli and Waddell, 2007). Figure 2.2 shows the relationship of objectives of a program and objectives of its related projects, in which multiple related projects are grouped under the same set of program objectives, and the program objectives at the higher level are cascaded down into one or more objectives of projects at the lower level. To achieve a particular program objective, it is necessary to achieve all its corresponding project objectives. For example, to achieve Program Objective 2, Project A Objective 1, Project A Objective 2 and Project B Objective 1 must be satisfied.

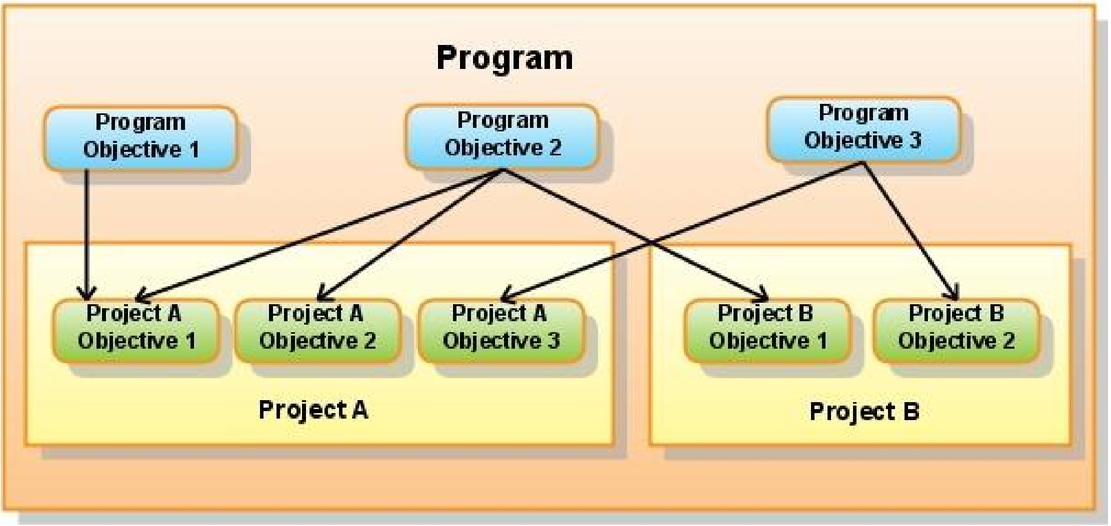


Figure 2.2: Relationship between Program and Project Objectives

2.1.2 Program and Project Risks

Before presenting the differences between program and project risks, we will first describe various ways that researchers have defined, quantified, and visualized the notion of risk.

2.1.2.1 The Definition of Risk

A risk is a potential event that will adversely affect the ability of a system to perform its mission should the risk event take place (Garvey, 2000). A risk has two basic attributes, *Probability P* and *Impact I*, where Probability stands for the likelihood that an event will occur. A risk R_x can thus be defined mathematically as a function of two attributes:

$$R_x = f(P_x, I_x)$$

As new risks are identified and mitigated over the duration of a project, the set of risks will vary. Assuming a set of n risks at time t, $R(t) = \{R_1, R_2, \ldots, R_n\}$, $R_x \in R(t)$

and $1 \leq x \leq n$, the value of R_x may change, and the number of risks n may also change. A common way to compute the risk value is the linear method which multiplies Probability and Impact together (Boehm, 1989):

$$R_x = P_x I_x$$

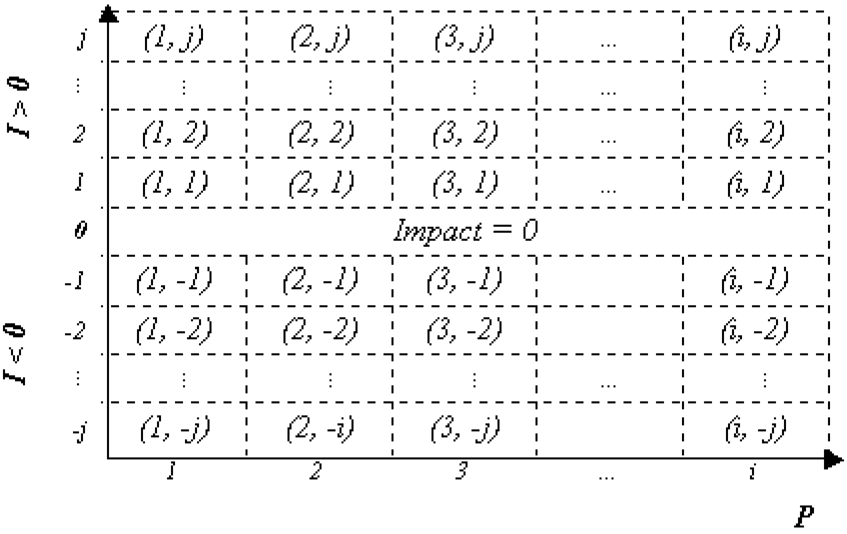
where P_x is the probability of R_x and I_x is the potential impact of R_x. Probability has a value between 0 and 1. Not all events are regarded as risky. There are three situations where they are not (White, 2006):

- the event will never happen ($P_x = 0, R_x = 0$);
- the event will certainly happen ($P_x = 1, R_x = I_x$);
- the event will not have any impact even if it does happen ($I_x = 0, R_x = 0$).

If $P_x = 0$, there will not be a risk as the event will not occur, i.e. $R_x = 0$. If P_x is 1, there will also not be a risk as it will certainly occur. I_x will not be equal to zero, as if $I_x = 0$, the event will have no impact, i.e. $R_x = 0$. A more precise computation of risk under the linear method is $R_x = P_x I_x$, $0 < P_x < 1$ and $I_x \neq 0$.

The recent risk management literature has broadened the definition of risk to include the notion that whereas a potentially negative event offers *risk*, a potentially positive event offers *opportunity* (COSO, 2004). Positive risk has also been considered by Kähkönen (2001) and according to the Project Management Body of Knowledge (PMI, 2008a), a project risk is an event that can have positive or negative effect on one or more project objectives and project risk management should seek to maximize the likelihood and impact of positive events as well as, as in the traditional view, minimize the likelihood and impact of negative events. This can be formulated as follows: for a negative effect $I_x > 0$, there will be a *risk* $R_x > 0$, and for a positive effect $I_x < 0$, there will be *opportunity* $R_x < 0$. In the following sections, R_x will represent both *risk* and *opportunity* and, unless otherwise stated, the term *risk* will include the meaning of *opportunity*.

Risk can also be ranked in a tabular format (Boehm, 1989; Charette, 1989; Dorofee *et al*., 1996) representing relative probabilities of occurrence and scales of impact. Assuming a relative scale from *1* to *i* for probability values, and another scale from *-j* to *j* for impact values, a risk R_x can be expressed as a pair of values, i.e. $R_x = (P_x, I_x)$, and a risk matrix generated, as in Figure 2.3. This risk matrix can be used in placing various combinations of Probability and Impact values into a risk assessment matrix (Figure 2.4). For a positive impact value, $I_x > 0$, R_x is classified as "High Risk" if both P_x and I_x are relatively high, and classified as "Low Risk" if both P_x and I_x are relatively low. In contrast, when the impact value is negative $I_x < 0$, R_x represents an *opportunity*, and it can be classified as shown in the bottom half of Figure 2.4.



	1	2	3	...	i
j	(1, j)	(2, j)	(3, j)	...	(i, j)
⋮	⋮	⋮	⋮	...	⋮
2	(1, 2)	(2, 2)	(3, 2)	...	(i, 2)
1	(1, 1)	(2, 1)	(3, 1)	...	(i, 1)
0	Impact = 0				
-1	(1, -1)	(2, -1)	(3, -1)		(i, -1)
-2	(1, -2)	(2, -2)	(3, -2)		(i, -2)
⋮	⋮	⋮	⋮	...	⋮
-j	(1, -j)	(2, -i)	(3, -j)		(i, -j)

Figure 2.3: Table-based Risk Matrix

Figure 2.4: Risk Assessment Matrix for Determining Risk Severity Level

The recent work viewed risks from two key perspectives: *software development process* and *software development project*. Chittister and Haimes (1993) proposed a framework for the assessment and management of risk associated with the software development process. They divided risks into three decompositions: (1) *Functional Decomposition* with seven attributes: requirement, product, process, people, management, environment and system development; (2) *Source-based Decomposition* with four sources of failures: hardware, software, organizational and human; and (3) *Temporal Decomposition* with risk sources related to development stages. Hyatt and Rosenberg (1996) presented a software quality model to relate quality attributes with risks, and focused their interests on the software development process and the associated deliverables. From the software development project perspective, Sherer (1995) conducted a survey and summarized risks in three dimensions: the *technical dimension* resulted from the uncertainty of tasks and procedures, the *organizational dimension* resulted from poor communication and organizational structure, and the *environmental dimension* resulted from changing environments and problems with external relationships. Conrow and Shishido (1997) aggregated risk sources from previous studies of software intensive projects into six risk issues: project level, project attributes, management, engineering, work environment, and others. Longstaff, Chittister, Pethia and Haimes (2000) later presented a framework for identifying the source of software risks in system integration. Their framework consists of seven areas: software development, temporal, leadership, environment, acquisition, quality and technology, and addresses risks from each area. Ropponen and Lyytinen (2000) conducted a survey and identified six software risk components: scheduling and timing risks, functionality risks, subcontracting risks, requirements management, resource usage and performance risks, and personnel management risks. They examined how risk management and environmental factors influence those risk components. Murthi (2002) recently revealed that most risk taxonomies cannot cover the external risks that affect real projects, and categorized risks in requirements, technology, business, political, resources, skills, deployment and support, integration, schedule, maintenance and enhancement, and design.

2.1.2.2 Definitions of Program and Project Risk

As discussed earlier, project risk is an event that can have positive or negative effect on one or more project objectives (PMI, 2008a). The success of a project is determined by whether the initial project objectives are met and a risk may affect the achievement of one or more defined project objectives (APM, 2004). This impact can also be measured using the risk assessment matrix, which shows the extent to which the risk affects those objectives. Assuming that z objectives are defined for a project, $OBJ = \{OBJ_1, OBJ_2, \ldots, OBJ_z\}$. A project risk R_x can have a set of relative impact values affecting one or more project objectives in OBJ, $\{I_x(OBJ_1), I_x(OBJ_2), \ldots, I_x(OBJ_z)\}$, where $-j_y \le I_x(OBJ_y) \le j_y$ and $1 \le y \le z$. $I_x(OBJ_y)$ is the impact value of R_x on the corresponding project objective OBJ_y. For a particular project objective, say OBJ_y, $I_x(OBJ_y) = 0$ means that R_x has no impact on OBJ_y.

Program risk is defined as an uncertain event that has positive or negative effect on one or more program objectives if the risk occurs (Hillson, 2008; PMI, 2008b). Although a program is constituted with multiple related projects whose risks may affect the project objectives of delivering specific solutions that aggregately fulfill the program objectives, program risks do not derive solely from project risks. Other than the risks considered in the project context, there are additional risks that may affect the program to achieve its objectives. Assume that there are a set of λ program objectives $Q\_OBJ = \{Q\_OBJ_1, Q\_OBJ_2, \ldots, Q\_OBJ_\lambda\}$, the impact of a program risk can also be represented by a set of relative impact values affecting one or more program objectives in Q_OBJ.

In practice, program risks can be viewed from different perspectives, leading to different program risk categorizations. For instance, Brown (2008) described a risk environment of program management that includes risks from three levels: *Business Level*, *Program Level* and *Project Level*. Following Brown, as a program sits between its related projects and organizational strategies, Hillson (2008) identified three potential sources of program risks: (1) risks could be delegated from above (i.e. the organizational strategy level); (2) risks could be arisen at the program level; and (3) risks could be escalated or aggregated from below (i.e. projects or the components of the program). Zacharias (Zacharias *et al*., 2008) developed a Risk Breakdown Structure (RBS) for program management, which could be applied to any kinds of programs. There were four basic elements in the highest level of RBS including *Management*, *Project Implementation*, *External* and *Operational Program Planning*. In the second edition of *The Standard for Program Management*, PMI

(2008b) stated that program risks can come from its related projects and their interactions with each other, from technical complexity and other constraints, and from the broader environment in which the program is managed. Program risks are grouped into six categories: *Environment-level risks*, *Program-level risks*, *Project risks*, *Operational-level risks*, *Portfolio-related risks*, and *Benefits-related risks*.

The different categorizations of program risks can be used to identify risks within a program. However, they do not indicate who (program managers or project managers) should own the risks. Based on the ownership of risks, we divide program risks into two levels: *Project Level Risks* (*PJR*) that should be managed by project managers and *Program Level Risks* (*PGR*) that should be managed by program managers. By this grouping, the major advantage is that program managers can clearly distinguish the risks that could directly affect the success of a program from the risks that could affect the successes of the projects in the program. From the program management perspective, program managers could then be able to track the changes of their responsible risks, and evaluate the performance of their own implementation of risk management responses. Meanwhile, they could still monitor the status of the risks at the project level reported by project managers. Our grouping complements other categorizations of program risks, and provides a higher level view of the risks. Table 2.2 summarizes and presents the mapping of different categorizations of program risks.

Table 2.2: Different Categorizations of Program Risks

	Brown (2008)	**Hillson (2008)**	**Zacharias (2008)**	**PMI (2008b)**
Program Level Risks (*PGR*)	Business Level	Organizational Strategy level	Management	Benefits-related risks
				Portfolio-related risks
			External	Environment-level risks

	Program Level	Program Level	Operational Program Planning	Program-level risks
Project Level Risks (*PJR*)	Project Level	Projects or components	Project Implementation	Project risks
				Operational-level risks

2.2 Project Risk Management Practices

In this section we will describe some standard project risk management practices based on the underlying definitions and basic assumptions described in the last section. The classic risk management studies mainly focus on risk identification (Williams *et al.*, 1997), but the purpose of project risk management includes risk identification and risk management (i.e. analyze, track, control and communicate) (Higuera and Haimes, 1996). Risk management is also not a one-time effort that performed at the project start; it is a continuous process throughout the entire project, in which risks are repetitively identified, recorded for analysis and communicated across all related parties (Noor, 2001).

The Software Engineering Institute (SEI) has developed a risk management paradigm (Van Scoy, 1992), which is an elaboration of the classic “plan-do-check-act” cycle of project management (Williams *et al.*, 1997), and defines a systematic risk management process that is made up of a set of cyclic steps as continuous activities throughout an IT project. The steps include “Identify”, “Analyze”, “Plan”, “Track” and “Control”. Other than these five-step of managing risks, the paradigm also contains a “Communicate” component which lies at the center of the model. The communication facilitates the interaction among all the elements of risk management and ensures information is shared effectively among the appropriate organizational levels and across developers, customers and users

(Higuera and Haimes, 1996). The common risk management processes and management practices adopted in the industry (Kwan and Leung 2007) can be mapped to the SEI paradigm as shown in Table 2.3. In the following we consider each of these five practices in turn.

Table 2.3: Basic Project Risk Management Practices

SEI Paradigm	**Basic Risk Management Practices**
Identify	1. Identify project risks
Analyze	2. Evaluate & prioritize risks
Plan	3. Develop risk response plans
Track	4. Monitor status of risk & associated risk response actions
Control	5. Control risk response actions

2.2.1 Identify Project Risk

If a risk cannot be identified, it cannot be managed and mitigated. Current risk identification processes involve examining the major areas of a project, collecting input from personnel, learning from past experience, and applying analytical tools and techniques. Table 2.4 provides examples of some common risk identification approaches (SEI, 2006; ASC, 2003; PMI, 2008a). Most of these approaches identify and manage events independently and tend to identify risks rather than opportunities; so these approaches are often complemented with techniques such as SWOT Analysis, Constraints and Assumptions Analysis, and Force Field Analysis (Hillson, 2001).

Table 2.4: Common Project Risk Identification Approaches

Risk Identification Approaches	**Examples:**
Examining project particulars	- Reviewing and study key assumptions and constraints of the project - Examining major project deliverables

Collecting input from personnel	- Asking opinions of subject matter experts - Conducting brainstorm session with staff - Applying Delphi technique
Learning from past experience	- Learning from similar projects - Reviewing organization’s project repository - Reviewing risk knowledge base
Applying analysis tools	- Conducting assessment using risk taxonomy - Using organization self-developed checklists - Applying diagramming techniques, such as • Cause-and-effect diagrams • System (or Process) flow charts • Influence diagrams

Among all the approaches, one of the more powerful risk identification methods is the taxonomy methodology (Carr *et al*., 1993), which provides a framework mapping the characteristics of software development and software development risks. The taxonomy consists of three major *Classes* of risks, with each class divided into *Elements* and each element is further characterized by *Attributes*. The three risk classes represent three different aspects of software development projects. *Product Engineering* includes all the technical aspects of the work to be accomplished; *Development Environment* addresses all the methods, procedures, and tools used to produce the product; and *Program Constraints* covers the contractual, organizational, and operational factors within which the software is developed (Carr *et al*., 1993). Together with the risk taxonomy, a Taxonomy-Based Questionnaire (TBQ) is designed. The TBQ consists of questions for each attribute, with which organizations can identify software development project risks accordingly.

This risk taxonomy was originally developed and served as a guideline or checklist for software development project teams systematically identifying their development risks. It is a useful tool for project risk identification, but it is not comprehensive enough to satisfy the true needs of today’s IT projects. Instead of treating projects solely from the software engineering point of view, Kwan and

Leung (2005) further enhanced the taxonomy by identifying eleven new attributes and extending the scope of thirty original attributes from the perspective of systems engineering, which integrates all the disciplines and specialties to address both the business and the technical needs of customers (Sage, 1995). Moreover, the enhancement had also considered other related risk drivers from program management, and IT service management which is the integrated approach to deliver IT services to meet business and customer requirements (ISO20000-2, 2005).

2.2.2 Evaluate and Prioritize Risk

There are two major types of risk analysis (or evaluation) in project management, Qualitative Risk Analysis and Quantitative Risk Analysis (PMI, 2008a; COSO, 2004). The two methods commonly applied in the project risk analysis are Failure Mode Effect Analysis (FMEA) and Failure Mode Effect and Criticality Analysis (FMECA) (Lock, 2007). FMEA is used to identify the failure modes (e.g. risks) and their associated effects, and FMECA is used to rank the failure modes according to their criticality and their probability (Bouti and Kadi, 1994).

Risks are usually presented in table-based ranking with assigned relative scale values to Probability and Impact of the risks. For example, all these values could be represented in a simple numerical scale, such as from “4” to “1”. Probability and Impact values can be placed in the Risk Assessment Matrix and prioritized, as in Figure 2.5. The risk level also determines the priority of responding to the risk. In Figure 2.5, when the impact value $I > 0$, the corresponding portion of the figure is for assessing risks and when the impact value $I < 0$, it is for assessing opportunities.

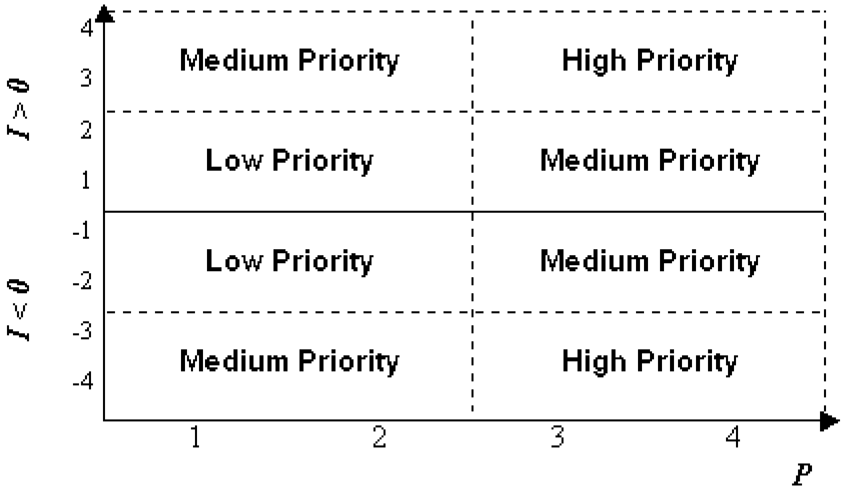


Figure 2.5: Risk Assessment Matrix for Prioritizing Response Actions

2.2.3 Develop Risk Response Plans

The table-based risk ranking approach allows organizations to select appropriate risk response strategies. A Risk Response Matrix (Figure 2.6) (COSO, 2004) helps to determine appropriate risk response actions to be taken for a particular risk or opportunity. Table 2.5 summarizes the typical risk response actions given in the Risk Response Matrix.

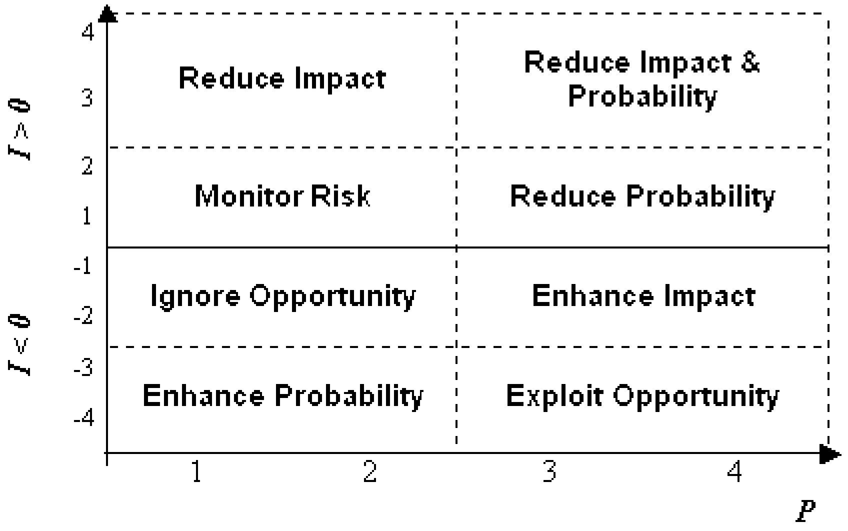


Figure 2.6: Risk Response Matrix

Table 2.5: Risk Response Actions

	Severity Level	**Probability**	**Impact**	**Purpose of Response Actions**	**Description**
Risk ($I > 0$)	High	High	High	Reduce Impact & Probability	Actions should be taken to either reduce the risk severity level (by reducing the risk likelihood and risk effect) or remove the risk
	Medium	High	Low	Reduce Probability	Actions should be taken to reduce the risk likelihood
	Medium	Low	High	Reduce Impact	Actions should be taken to reduce the risk effect
	Low	Low	Low	Monitor Risk	No actions will be taken to the risk, except monitoring
Opportunity ($I < 0$)	High	High	High	Exploit Opportunity	Actions should be taken to realize the opportunity
	Medium	High	Low	Enhance Impact	Actions should be taken to enhance the positive effect of the
	Medium	Low	High	Enhance Probability	Actions should be taken to enhance the likelihood of the opportunity
	Low	Low	Low	Ignore Opportunity	No actions are needed

The strategies for responding to Risks and Opportunities are very different. There are four common risk response strategies (COSO, 2004; PMI, 2008a; Hillson, 2001), *Avoid*, *Reduce*, *Transfer*, and *Accept*, as follows:

- *Avoid*: This strategy applies to high severity risks and involves eliminating and removing the risk by reducing both the probability and impact of a risk, for example, changing or reducing the project scope.
- *Reduce*: This strategy applies to medium and high severity risks, and involves reducing the probability and/or impact of a risk to an acceptable level. Examples include conducting additional tests, and strengthening or instituting more controls.

- *Transfer*: This strategy also applies to medium severity risks, and involves transferring or sharing the responsibility and/or the impact of a risk (i.e. reducing the risk impact) with other parties who could better manage the risks. Examples include paying a risk premium to another party to assume the risk.
- *Accept*: This strategy applies to low severity risks. Nothing will be done to deal with such risks but the project team may continuously monitor them or establish a risk contingency plan.

Similarly, there are four common response strategies for opportunities (PMI, 2008a; Hillson, 2001), *Exploit*, *Enhance*, *Share*, and *Ignore*:

- *Exploit*. This strategy applies to those high severity level opportunities, and its response actions involve assuring the identified opportunities to be realized by investing more efforts or eliminating any uncertainty associated with the opportunities. Examples include hiring better experts, employing more advanced technologies, or allocating appropriate resources.
- *Enhance*. This strategy applies to those medium and high severity level opportunities, and its response actions involve increasing the probability and/or impact of the opportunities by focusing on their key drivers and strengthening the causes of the opportunities in order to maximize the project benefits.
- *Share*. This strategy applies to those medium severity level opportunities. Opportunity sharing actions involve shifting responsibility to other parties to enable the best chance of realizing the opportunity (i.e. increasing the probability of the opportunity). Examples include seeking a contractor who

has specific skills that could help to maximize the chances of opportunities happening and/or increase the potential benefits.

- *Ignore*. This strategy applies to those low severity level opportunities. Nothing is done to the identified opportunities due to their chances and impacts are both low.

2.2.4 Monitor Status of Risk & Associated Response Actions

Risk monitoring is carried out continuously throughout the project life cycle. The main objective is to monitor any changes of identified risks, the effectiveness of risk responses, and the performance of the implementation of risk management practices (PMI, 2008a; ITGI, 2007; SEI, 2006):

- The status of identified risks should be monitored until either the risks or the project have been closed, as the risks may change due to project changes or other external factors during the project life cycle, or they may be mitigated by executing the risk response actions.
- The risk response actions may involve a series of activities taken to deal with risks. Like other project activities, those activities should also be monitored, and the effectiveness of risk responses should be evaluated.
- Key performance indictors are defined to monitor and measure the implementation of risk management practices and serve as a measure of progress towards project objectives.

To effectively monitor the risks or opportunities of a project, collecting well-defined data continually and consistently over time is needed to detect any occurred changes. In Section 2.3, a number of common metrics for monitoring project risks will be discussed.

2.2.5 Control Risk Response Actions

Risk control is also an on-going process for the life of a project. With the risk monitoring results, risk control involves re-assessing risks and selecting alternative risk response actions (PMI, 2008a; SEI, 2006). As there may be status changes of existing risks, new risks identified, or variances of planned against implemented risk response actions, all risks have to be re-evaluated and re-prioritized periodically so that appropriate decisions and risk response actions could be made. Based on the risk re-evaluation and re-prioritization results, the risk response plans should be reviewed and updated.

2.3 Program Risk Management Practices

Program risk management is a set of ongoing processes during the execution of a program to manage risks across a set of projects in order to achieve the overall goals of the program. As projects are the primary components of a program, a significant amount of program risk management efforts focus on the project level risks. The basic risk management processes performed by project managers on project level risks includes conducting project risk management planning, identifying project level risks, performing risk analysis, planning risk responses, and monitoring and controlling risks (PMI, 2008a).

However, program risks consist of project and program level risks. It is the responsibility of program managers to manage program level risks that are outside the authority of project managers (PMI, 2008b). The risk management processes for those program level risks are similar to those conducted by project managers at the project level. But, program managers should not directly involve in the management of project level risks. Project managers need to report the risks in accordance to defined program risk management plan, and Program managers oversee the risks at a higher level and coordinate all the project managers in risk responses in order to obtain synergetic effects.

Tables 2.6 compares the current practices (PMI, 2007; PMI, 2008a; PMI, 2008b) of program and project risk management to address the risks in a program.

Table 2.6: Management Practices for Program Risks

	Applied to Project Level Risks	**Applied to Program Level Risks**
Program Risk Management Practices	- Manage contingency reserve across entire program	
	- Identify and analyze inter-project risks - Oversee risks and responses at the project level within the program - Review risk response actions that could affect other projects - Provide solutions to risks that escalated by project managers - Implement response mechanisms that benefit projects within the program	- Conduct program risk management planning - Identify program level risks - Perform program level risk analysis - Plan program level risk responses - Monitor and control program level risks
Project Risk Management Practices	- Conduct project risk management planning - Identify project risks - Perform project risk analysis - Plan project risk responses - Monitor and control project risks	N.A.

2.4 Metrics for Analyzing Risk and Risk Response

Risk metrics can be applied at different times in a project to facilitate the analysis of overall project risk level and the effectiveness and performance of risk responses. In this section, we review some of the common metrics. These metrics are the Risk Score, the Risk Index, Top N Project Risk, and Project Risk by Objectives. Our discussions of these metrics will all assume that n independent risks $R(t) = \{R_1, R_2, \ldots, R_n\}$ are identified at time t for a project such that $0 \leq R_x \leq M$ where $R_x \in R(t)$ and M is the maximum possible value of R_x. The metrics to be introduced can also be applied to opportunities, only that $-M \leq R_x \leq 0$.

2.4.1 Risk Score and Risk Index

In order to get an overall picture of how risky a project is, we can use Risk Score $RS(t)$ and Averaged Risk Score $ARS(t)$. This involves adding and averaging all identified risks of the project at time t with the assumption that they are independent (Ferguson, 2004). Tracking the total risk value periodically during a project allows management to monitor project risk, and evaluate the effectiveness and performance of risk management actions.

$$(2.1) \quad RS(t) = \sum_{x=1}^{n} R_x$$

$$(2.2) \quad ARS(t) = \frac{1}{n} RS(t) = \frac{1}{n} \sum_{x=1}^{n} R_x$$

As risks will vary over the life of a project, $RS(t)$ will also change with t. The value of $RS(t)$ ranges from 0 (when all risks in $R(t)$ are at the lowest risk value)[1] to nM (when all risks in $R(t)$ are at the highest risk value with the assumption that the impacts of risks will not overlap). The value of $ARS(t)$ ranges from 0 to M.

The risk score can be computed without knowing the value of M. It is useful to analyze the trend of overall project risk level over time. However, if we can obtain

---

[1] Initially, a risk should not be 0 when it is first identified; however, if the risk is mitigated or the risk event does not happen, the risk level can then be set to 0.

the maximum possible value *M* of a risk (it is only meaningful if the ranking approach is used when evaluating risks in which each risk is assigned with a relative scale number), we can define another metric for measuring the overall project risk, the Risk Index, *RI(t)*:

$$(2.3) \quad RI(t) = \frac{RS(t)}{n\ M} = \frac{1}{n\ M} \sum_{x=1}^{n} R_x$$

The index value will be at its minimum *0* when all risks are at the lowest risk level, and it will reach its maximum value of *1* when all risks are at the highest risk level. The risk index can not only support analysis of the risk trend within a project, but also can facilitate the comparison of risk trends between projects.

2.4.2 Top N Project Risk

In practice, it is not necessary to track every risk and senior management usually will focus on the top 10 risks (Boehm, 1991). As suggested by Ferguson (2004), however, in larger projects it may be more appropriate to monitor the top 20 risks. It is indeed possible to modify the risk score and risk index to represent only the top risks in a project.

If there is a set of *N* risks $S(t) \subseteq R(t)$, and $|S| = N$, such that $\forall R_a(R_a \geq R_b;\ R_a \in S(t),\ R_b \in (R(t)–S(t))\ )$, *S(t)* contains the top *N* risks of a project at a given time. Taking the sum of all risks in *S(t)* obtains the Risk Score for the top *N* risks, $RS_N(t)$:

$$(2.4) \quad RS_N(t) = \sum_{x \in S(t)} R_x$$

Similarly, to calculate the risk index for the top *N* risks $RI_N(t)$ at a given time, $RS_N(t)$ is averaged out and divided by the maximum value *M* of risk level:

$$(2.5) \quad RI_N(t) = \frac{RS_N(t)}{N\ M} = \frac{1}{N\ M}\sum_{x \in S(t)} R_x$$

The value of $RI_N(t)$ will range from 0 to 1; a lower index value means that the top N risks have a lower risk level, while a higher index value means that the top N risks have a higher risk level. As $RI_N(t)$ tracks the top N risks at a given time, it can also be used to analyze trends in project risk.

2.4.3 Project Risk by Objectives

As the potential impact of an event can be represented by the strength of its potential impact on project objectives, it is useful to have a set of risk indexes for each project objective. This can be obtained by segregating the risk index to express effects on different project objectives.

Assume that there are a set of z project objectives $OBJ = \{OBJ_1, OBJ_2, \ldots, OBJ_z\}$. $R_x = f(P_x, I_x)$, where $R_x \in R(t)$, and $I_x = \{I_{x1}, I_{x2}, \ldots, I_{xz}\}$ is a set of the impact values of R_x affecting the corresponding project objectives $\{OBJ_1, OBJ_2, \ldots, OBJ_z\}$ in OBJ. Therefore, a risk could have different values of risk levels for different project objectives, i.e. $R_x = \{R_{x1}, R_{x2}, \ldots, R_{xz}\}$. The Risk Score, $RS^j(t)$, for a particular project objective OBJ_j, where $1 \leq j \leq z$ and $OBJ_j \in OBJ$, at a given time t can be obtained by taking the sum of all risks with respect to the objective OBJ_j:

$$(2.6) \quad RS^j(t) = \sum_{x=1}^{n} R_{xj}$$

Similarly, to calculate the risk index, $RI^j(t)$, with respect to a selected project objective OBJ_j at a given time t, the $RS^j(t)$ is averaged and divided by the maximum value M of risk level:

$$(2.7) \quad RI^{j}(t) = \frac{RS^{j}(t)}{n\,M} = \frac{1}{n\,M}\sum_{x=1}^{n} R_{xj}$$

The index measures the risk level of a selected project objective at a particular time. The value of $RI^{j}(t)$ will range from 0 to 1; a lower index value means that there is a higher chance of achieving the project objective OBJ_j, while a higher index value means that there is a lower chance of meeting OBJ_j. If we calculate all the risk indexes of each project objective, we can obtain a complete risk picture of all the project objectives at a given time. In order to produce an overall risk index against all z project objectives, we can add up and average the individual indexes to obtain the Average Risk Index $ARI(t)$.

$$(2.8) \quad ARI(t) = \frac{1}{z}\sum_{j=1}^{z} RI^{j}(t)$$

In this case, the sum of all indexes will range from 0 (when each $RI^{j}(t)= 0$) to z (when each $RI^{j}(t)= 1$). Therefore, the value of $ARI(t)$ will range from 0 to 1; at time t, a lower index value means that the project will have a higher chance of meeting all its objectives, while a higher index value means that the chance will be lower.

The importance of each project objective may be different. For example, if a project is to develop an Internet banking system, the data security issue can be one of the most concerned areas, and providing a secure environment for banking transaction may become one of the most important project objectives. We can further prioritize the risk response actions based on the extent to which the risk impacts the most important project objectives. Therefore, we may choose not to deal with the risks associated with less important project objectives, but can assign resources to eliminate risks that affect the more important project objectives.

2.5 Dependency Analysis Models

Current project management practices do not clearly address how dependencies between risks are managed. In this section, we present several dependency analysis models which have been used to represent the dependency of one event on another.

2.5.1 Tree-based Analysis

In this section, three tree-based analysis techniques are discussed; they are Fault Tree Analysis, Event Tree Analysis, and Cause-Consequence Analysis.

2.5.1.1 Fault Tree Analysis

Fault tree is a logical diagram used in the Fault Tree Analysis (FTA) (IEC61025, 2006) to represent the possible causes of an undesired event. The root (or the top node) of the tree represents the undesired event, and the other events (i.e. the causes) that lead to the root are modeled by independent leaf nodes with a series of logical expressions. For example, a fault tree can present the relation between the failure of a system and failures of the system components (Aven, 1992). In this case, if each leaf node is assigned a failure probability, the system failure probability can then be calculated. Other than quantitative analysis, the fault tree can also be applied for qualitative analysis (Sutton, 1992). There is another analysis tree that is similar to the fault tree; it is called the Management Oversight and Risk Tree (MORT) (Johnson, 1973). MORT analysis is performed in the same way as FTA, but it was specifically developed for safety analysis, involving approximately 1500 safety elements as nodes of the fault tree.

2.5.1.2 Event Tree Analysis

Event Tree Analysis (ETA) (IEC60300-3-9, 1995) is a method to illustrate the sequence of possible outcomes (or consequences) after the occurrence of an undesired event. Similar to a fault tree, an event tree starts from an undesired event (i.e. the top node of the event tree), and the event is linked to its outcomes towards the final consequences with a probability of occurrence assigned to each tree branch. The event tree can be used for both qualitative and quantitative analysis.

2.5.1.3 Cause-Consequence Analysis

Cause-consequence analysis (CCA) (Nielsen, 1971) combines the FTA and ETA, and is performed with a cause-consequence diagram, which starts from an undesired event and develops backwards to identify its causes (presented by a fault tree) and forwards to identify its consequences (presented by an event tree). CCA can help to identify the chain of events from the initiators of an undesired event to its final consequences. With the probabilities of occurrence attached to all the associated events in the cause-consequence diagram, the probabilities of the different consequences of the undesired event can then be calculated.

2.5.2 Markov Analysis

Markov analysis (IEC61165, 2006) provides a mathematical method to analyze the reliability and availability of systems, which are well specified and have strong component dependencies. In this analysis, a system is modeled as a number of discrete states with possible transitions among the states. The states are graphically presented as nodes in a directed graph, where the edges represent the probabilities of going from one node to another node. According to the probability distribution, the system transits from its current state to the next state. In this way, the future states only depend on the current state, and are independent of the past states. Statistical calculations are performed to estimate the sequence of states and analyze the trend of which state that will be followed by another state. In contrast with tree-based analysis, Markov analysis does not require component independence and an acyclic structure.

2.5.3 Bayesian Network

Bayesian network (Pearl, 1988) is a directed acyclic graph (DAG) in which each node represents a variable (which can be discrete or continuous) and each arc represents causal or probabilistic influential relationships between variables. A link between two variables represents a probabilistic dependency between them. A Bayesian network can be analyzed qualitatively or quantitatively. When a Bayesian network is analyzed qualitatively, it provides the relations of causes and effects between nodes. If a Bayesian network is analyzed quantitatively, it is a representation of a joint probability distribution, in which each node is associated with a conditional probability distribution reflecting its parent nodes.

Although a Bayesian network is a model that can represent probabilistic dependencies and independencies, the links between variables do not normally carry any meaning. However, if the links are interpreted as direct causal influences between variables (i.e. a variable is a cause of another variable), the network is then called a Causal Network (Pearl, 1988).

Bayesian network is often used, particularly when it is applied with probability theory, to manage uncertainty by explicitly presenting the conditional dependencies between different knowledge components. However, the computations involved in a Bayesian network with a reasonable number of variables are very complex and cannot be easily done manually. With the assistance of appropriate tools, like AgenaRisk (Fenton and Neil, 2004), the effects of both forward and backward inferences can be computed, and various types of ‘what-if’ and sensitivity analysis can then be performed (Fenton and Neil, 2004).

2.5.4 Goal-Risk Model

A goal model, represented as a directed graph, is used to refine the goals of a target system by decomposition (by the means of AND / OR refinement relationships) into measurable sub-goals (Navarro *et al.*, 2007). Tropos goal model (Giorgini *et al.*, 2003) is a goal model framework for requirement analysis by refining stakeholders’ goals. This framework allows analysts to model the influence of the satisfaction of a goal to the satisfaction of other goals. The influence can be expressed as strong positive “+ +”, positive “+”, strong negative “– –” or negative “–” contribution relations.

The Goal-Risk Model (Asnar and Giorgini, 2006) is a risk modeling and reasoning framework that further extends the Tropos goal model into three layers: Goal, Event (including risk and opportunity), and Treatment. The analysis starts by identifying a relevant event that can influence any goals in the goal layer. The event is decomposed with contribution relations until all of its leaf-events are mutually exclusive. Once the events have been analyzed, corresponding treatments are identified and analyzed. This framework is useful to explicitly model the risks with the relations between stakeholders’ goals in the upper layer, other risks in the event layer, and the associated treatments in the lower layer. Figure 2.7 gives an example of goal-risk model from (Asnar and Giorgini, 2007).

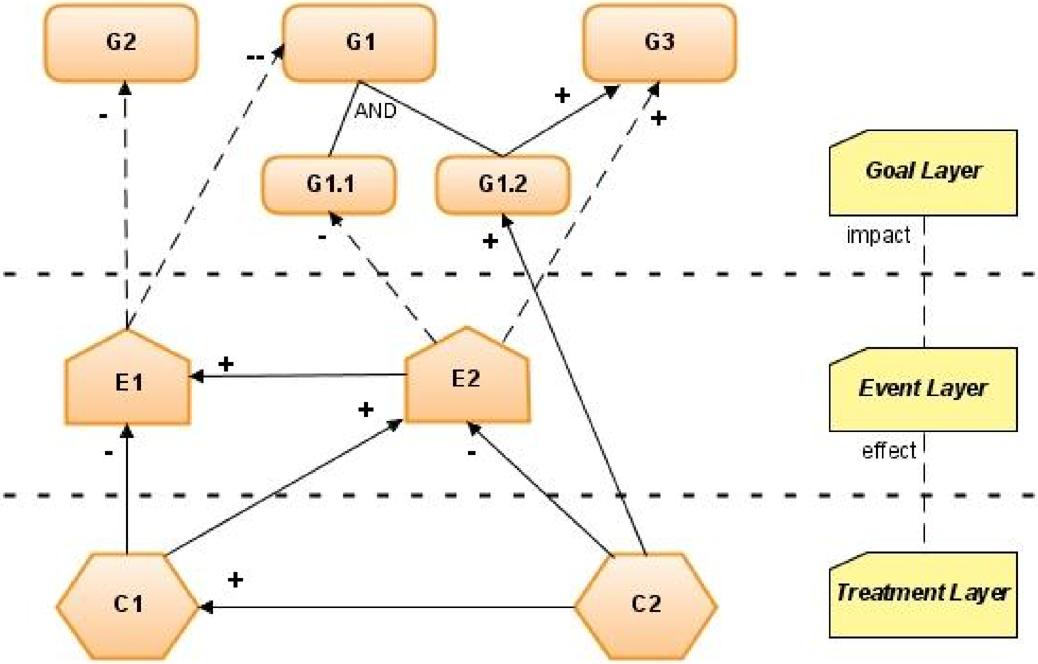


Figure 2.7: An Example Goal-Risk Model

2.6 Summary

In this chapter, we have reviewed how risks will affect program and project objectives, and how the current practices manage risks from program and project perspectives. All of the practices only view risks as independent events and manage them individually. We have also reviewed the modeling techniques that are used to analyze the dependency relationships between events. The techniques help to analyze dependency relationships and are applied in different disciplines, but none of them can satisfy the needs of modeling risk dependency in real practice; the risk dependency model needs to: (1) support a cyclic structure, (2) support for both qualitative and quantitative analysis, and (3) adequately be embedded in the risk management practices.

Chapter 3 Risk Dependency Modeling

While it is common for risks to be identified and managed independently, some projects risks in fact can be mutually dependent. For example, there may be a dependency relationship between a risk where on the one hand a vendor may not be able to recruit enough subject experts on time and on the other there is the risk that a technical design specification may be poorly developed. The dependency relationship in this case is that any increase in the likelihood of the first risk event makes the second risk event correspondingly more likely.

In this chapter, we introduce a number of novel concepts related to risk dependency. We first explain how to represent a risk dependency and discuss the methods to obtain a revised risk value with the risk dependency factor. We then propose several ways to estimate the combined risk dependency effect when there are more than one risk dependency affecting a particular risk. Since a risk dependency effect may either increase or lower the probability of a risk, we also define the favorability of risk dependency effect. Finally, we consider how to model the risk dependencies of a project with risk dependency graphs and propose a risk index to evaluate the extent of dependencies among risks within a project.

3.1 Risk Dependency

The risk dependency is referring to an effect due to the occurrence of a risk and this effect can either increase or decrease the probability of occurrence of other risk(s). For any two identified risks in a given set of risks, R_a and R_b, if the occurrence of risk R_a has an effect on risk R_b, we write $R_a \rightarrow R_b$. In this case, R_b has a *Risk Dependency* relationship with R_a; R_b is called a *Dependent Risk* or *Direct Successor* of R_a, and R_a is called the *Direct Predecessor* of R_b. There should not be any dependency relationship by a risk on itself. For any two risks, R_a and R_b, there could be three possible relations between them:

- R_b is a dependent risk of R_a, i.e. $R_a \rightarrow R_b$
- R_a is a dependent risk of R_b, i.e. $R_b \rightarrow R_a$
- R_a and R_b are dependent on each other, i.e. $R_a \rightarrow R_b \wedge R_b \rightarrow R_a$

Thus, there can be three types of risk dependencies between any two risks, R_a and R_b, as shown in Figure 3.1:

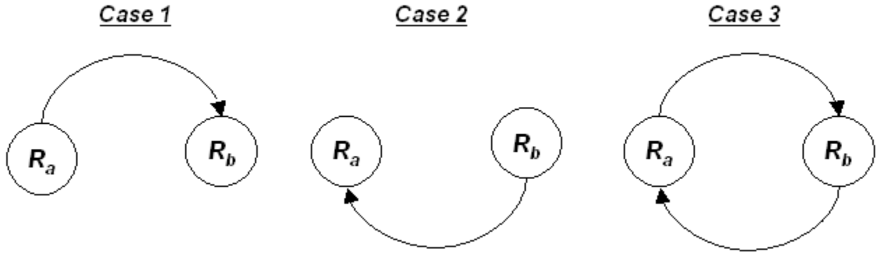


Figure 3.1: Possible Risk Dependencies between Two Risks

As each risk has two tuples, the Probability P and the Impact I, for the case of $R_a \rightarrow R_b$, if R_a occurs, the risk dependency may have an effect on either P_b or I_b of R_b. At a given time, if the impact estimations are done correctly, the impact of R_b should have already considered the effect of other risks. It will not need to be changed even if we add the risk dependency. Therefore, the risk dependency can be viewed as *Probability Dependency* as it will only have an effect on P, where the effect can be either positive or negative. There are two ways to represent the effect on the probability from one risk to another risk, namely the *Risk Dependency Value* and the *Risk Dependency Multiplier*.

Definition 1: Given a set of identified risks $R(t)$ at a given time t and $|R(t)| = n$, for each $R_x \in R(t)$, $R_x = f(P_x, I_x)$ where $1 \leq x \leq n$ and $P_x \in P$ which is the set of possible Probability values. If R_b has only one *Direct Predecessor* R_a, $R_a \rightarrow R_b$ where $R_a, R_b \in R(t)$ and $R_a \neq R_b$, there exists a Risk Dependency Value D_{ab} between R_a and R_b such that:

$$
\begin{aligned}
(3.1) \qquad R_b^{+a} &= f(P_b^{+a}, I_b) \qquad \text{where } P_b^{+a} \in P \\
&= f(P_b + D_{ab}, I_b)
\end{aligned}
$$

R_b^{+a} is called the *Posterior Risk* of R_b, which has considered the effect of risk dependency from R_a. P_b^{+a} is called the *Posterior Probability*, which is the result of the change of P_b caused by D_{ab}, where $D_{ab} \leq (1 - P_b)$.

D_{ab} has the following properties:

- If $D_{ab} = 0$, $R_b^{+a} = R_b$. It means that R_a does not have any risk dependency effect on R_b.
- If $D_{ab} \neq 0$, R_b is Risk Dependent (or Probability Dependent) on R_a.
- If $D_{ab} > 0$, the occurrence of R_a can increase the likelihood of occurrence of R_b.
- If $D_{ab} < 0$, the occurrence of R_a can decrease the likelihood of occurrence of R_b.
- If $D_{ab} = -P_b$, $R_b^{+a} = 0$; it means that the occurrence of R_a can make R_b disappear as it will not occur.
- There will not be a case that $D_{ab} < -P_b$. If $D_{ab} < -P_b$, P_b^{+a} will become

negative, but it is impossible to have a negative probability.

- There will also not be a case that $D_{ab} \geq 1$, as $P_b^{+a} = D_{ab} + P_b$, P_b^{+a} and P_b are both within the range between 0 and 1.

Definition 2: Given a set of identified risks $R(t)$ at a given time t and $|R(t)| = n$, for each $R_x \in R(t), R_x = f\,(P_x, I_x)$ where $1 \leq x \leq n$ and $P_x \in P$ which is the set of possible Probability values. If R_b has only one *Direct Predecessor* R_a, $R_a \rightarrow R_b$ where $R_a, R_b \in R(t)$ and $R_a \neq R_b$, there exists a Risk Dependency Multiplier DM_{ab} between R_a and R_b such that:

$$(3.2) \qquad R_b^{+a} = f\,(P_b\, DM_{ab}, I_b) \quad \text{where } P_b\, DM_{ab} \in P$$

The Risk Dependency Multiplier DM_{ab} has a similar effect as D_{ab} where $DM_{ab} \leq (\,1\,/\,P_b\,)$. DM_{ab} has the following properties:

- If $DM_{ab} = 1$, $R_b^{+a} = R_b$; it means that there is no dependency between R_a and R_b.
- If $DM_{ab} = 0$, then $R_b^{+a} = 0$; it means that the occurrence of R_a can eliminate R_b.
- If $1 > DM_{ab} > 0$, then $R_b > R_b^{+a}$; it means that the occurrence of R_a can decrease the likelihood of occurrence of R_b.
- If $DM_{ab} > 1$, then $R_b < R_b^{+a}$; it means that the occurrence of R_a can increase the likelihood of occurrence of R_b.
- There will not be a case that $DM_{ab} < 0$. If $DM_{ab} < 0$, P_b^{+a} will become negative, but it is impossible to have a negative probability.

The dependency multiplier DM_{ab} is related to the risk dependency value D_{ab} and P_b. From (3.1) and (3.2), we get $P_b+D_{ab} = P_b\, DM_{ab}$; therefore,

(3.3a) $D_{ab} = P_b\,(DM_{ab} - 1)$

or (3.3b) $DM_{ab} = 1 + \frac{D_{ab}}{P_b}$

There are two common ways to obtain a risk value (Boehm, 1989; Charette, 1989; Dorofee et al., 1996), namely *Linear Method* and *Ranking Method*. We next consider the relationship between D_{ab}, DM_{ab}, P_a and P_b for these two methods.

3.1.1 Relating D_{ab} and DM_{ab} under the Linear Method

In the linear method, a risk is computed by multiplying its Probability and Impact values (Boehm, 1989), i.e. $R_b = f\,(P_b, I_b) = P_b\, I_b$, in which P is a real number between *0 and 1*. Thus, if $R_a \rightarrow R_b$, R_b^{+a} in (3.1) will become (3.4):

(3.4) $R_b^{+a} = f\,(P_b+D_{ab}, I_b) = (P_b+D_{ab})\, I_b = P_b\, I_b + D_{ab}\, I_b = R_b + D_{ab}\, I_b$

As $P_b+D_{ab} \in P$ and $0 \leq P_b+D_{ab} \leq 1$, therefore, $-P_b \leq D_{ab} \leq (1 - P_b)$.

Similarly, (3.2) will become (3.5):

(3.5) $R_b^{+a} = f\,(P_b\, DM_{ab}, I_b) = DM_{ab}\, P_b\, I_b = DM_{ab}\, R_b$

As $P_b\, DM_{ab} \in P$ and $0 \leq P_b\, DM_{ab} \leq 1$, therefore, $0 \leq DM_{ab} \leq (1 / P_b)$.

According to the definition of Conditional Probability, $P(b|a) = P_{ab} / P_a$, where P_{ab} is the joint probability of R_a and R_b, and Definition 1 stated above, $P(b|a) = P_b^{+a} = P_b + D_{ab}$, therefore from (3.3a),

$P_{ab} = P_a\,(P_b+D_{ab}) = P_a\,(P_b + P_b\,(DM_{ab} - 1)) = P_a\, P_b\, DM_{ab}$

In other words, the dependency multiplier DM_{ab} is also equal to the joint probability of R_a and R_b divided by the multiplication of their two independent probabilities, i.e.

(3.6) $$DM_{ab} = \frac{P_{ab}}{P_a P_b}$$

3.1.2 Relating D_{ab} and DM_{ab} under the Ranking Method

Another method to compute the risk value is the ranking method. A risk value is determined by applying a predefined table-based ranking. Each risk consists of a tuple (P_x, I_x). Assuming that a relative scale, 1 to i, is assigned to the probability values, thus the probability can take on a value between 1 and i. In this case, (3.1) will become (3.7) as shown below. If $R_a \rightarrow R_b$,

(3.7) $$R_b^{+a} = (P_b + D_{ab}, I_b)$$

As $P_b + D_{ab} \in P$ and $1 \leq P_b + D_{ab} \leq i$, we get

(3.8) $$(1 - P_b) \leq D_{ab} \leq (i - P_b)$$

Other than the general properties for D_{ab}, there are two additional properties for this ranking method:

- If $D_{ab} > i - P_b$, $R_b^{+a} = (i, I_b)$, because i is the largest possible probability value; the risk dependency cannot increase the probability value any more when the probability value has already reached its maximum.
- If $D_{ab} < -P_b$, $R_b^{+a} = 0$. The occurrence of R_a has already made R_b disappear; the risk dependency cannot decrease the probability value any further.

Under the ranking method, (3.2) will become (3.9):

(3.9) $$R_b^{+a} = (\lfloor P_b DM_{ab} + 0.5 \rfloor, I_b)$$ *where the value of $P_b DM_{ab}$ is rounded to the nearest integer*

As $\lfloor P_b DM_{ab} + 0.5 \rfloor \in P$, $1 \leq \lfloor P_b DM_{ab} + 0.5 \rfloor \leq i$. As $D_{ab} = P_b (DM_{ab} - 1)$ from

(3.3a), (3.8) then becomes $(1 - P_b) \le P_b (DM_{ab} - 1) \le (i - P_b)$, or

(3.10) $(1 / P_b) \le DM_{ab} \le (i / P_b)$

Other than the general properties for DM_{ab}, there are two additional properties for this ranking method:

- If $DM_{ab} > i / P_b$, $R_b^{+a} = (i, I_b)$, because i is the largest probability value; the risk dependency cannot increase the probability value any more when the probability value has already reached its maximum.
- If $DM_{ab} < 1 / P_b$, $R_b^{+a} = 0$. The occurrence of R_a has already made R_b disappear; the risk dependency cannot decrease the probability value any further.

3.2 Combined Risk Dependency

In the last section, we only consider the case when there is only one risk dependency. However, there can be more than one direct predecessor risks that affect a particular risk. We propose three approximation methods (Pang, 2004), namely the Conservative Method, the Optimistic Method and the Weighted Method, to calculate the Combined Risk Dependency Value, δ, and the Combined Risk Dependency Multiplier, λ, in the case of multiple direct predecessors.

Let $R_x = f (P_x, I_x)$, and R_x has k direct predecessors, say $R_1, R_2, \ldots, R_k$, where $x \neq 1, \ldots, k$. The posterior risk $R_x^+ = f (P_x^+ , I_x)$, where $P_x^+ = P_x + \delta_x$ or $P_x^+ = P_x \lambda_x$. The set of the Risk Dependency Values is $D_x = \{ D_{1x}, D_{2x}, \ldots, D_{kx} \}$ and the set of Risk Dependency Multipliers is $DM_x = \{ DM_{1x}, DM_{2x}, \ldots, DM_{kx} \}$.

The three methods for computing the combined risk dependency effect are presented next:

3.2.1 Conservative Method

This method picks the largest value from among all the Risk Dependency Values or all the Risk Dependency Multipliers of direct predecessors as the dependency effect on the probability of a risk or an opportunity; the implicit assumption is that the project will put a higher priority in mitigating risks but lower priority in exploiting opportunities. To adopt this approach for risks, the project should be critical to an organization, or the risk should have a high impact on the project objectives, as this method will maximize the dependency effect to a risk and hence may require more resources. On the contrary, for opportunities, as this method will minimize the dependency effect to an opportunity, use of this approach assumes that the project has limited resources on managing opportunities, or the opportunities may not add much value to the project objectives.

(3.11a) $\delta_x = Max\ (D_{1x}, D_{2x}, \ldots, D_{kx})$

(3.11b) $\lambda_x = Max\ (DM_{1x}, DM_{2x}, \ldots, DM_{kx})$

3.2.2 Optimistic Method

This method picks the smallest value among all the Risk Dependency Values or all the Risk Dependency Multipliers of direct predecessors, and minimizes the dependency effect to a risk or maximizes the dependency effect to an opportunity. Comparing with the Conservative approach, for risks, it assumes that the project should be less critical with fewer resources, and the risk should have a low impact on the project objectives; for opportunities, it assumes that the project allows more resources on managing opportunities, or the opportunities may add great value to the project objectives.

(3.12a) $\delta_x = Min\ (D_{1x}, D_{2x}, \ldots, D_{kx})$

(3.12b) $\lambda_x = Min\ (DM_{1x}, DM_{2x}, \ldots, DM_{kx})$

3.2.3 Weighted Method

The Weighted Method assigns a relative weighted value to each of the dependencies in order to calculate the combined dependency effect. As the weighting can be based on expert judgment or past experience, comparing with the Conservative and Optimistic approaches, this method can be more accurate in estimating the combined effect. Although this method can be applied to many situations (including the situations described in the Conservative and Optimistic methods), it requires the project to put extra efforts to evaluate each of the dependencies and determine the appropriate weighted values. To have better estimation, project experiences on managing dependencies should be captured and shared with other similar projects.

Given risk dependency values $D_{1x}, D_{2x}, \ldots, D_{kx}$ (or D_{ix} where $1 \leq i \leq k$), each D_{ix} is assigned its corresponding weighted value w_i, where $\sum_{i=1}^{k} w_i = 1$. The weighted Risk Dependency Value is computed as:

(3.13a) $\delta_x = \sum_{i=1}^{k} w_i D_{ix}$

Similarly, for Risk Dependency Multipliers, we have DM_{ix}, $1 \leq i \leq k$, and each DM_{ix} is assigned its corresponding weighted value w_i, where $\sum_{i=1}^{k} w_i = 1$. The weighted Risk Dependency Multiplier is:

$$(3.13b)\quad \lambda_x = \sum_{i=1}^{k} w_i \, DM_{ix}$$

3.3 Dependency Favorability

As the effects of risk dependencies can either increase or reduce the probabilities of those affected risks, it is needed to assess the favorability of the dependency effect for a particular risk before we can develop appropriate risk dependency response actions; that is to determine whether the dependency effect is favorable or non-favorable.

Definition 3: Given a set of identified risks of a project at a given time t, $R(t) = \{R_1, R_2, \ldots, R_n\}$, $R_x = f(P_x, I_x)$ and $R_x \in R(t)$, and a dependency effect that makes R_x become R_x^+. If $R_x^+ > R_x$, the dependency effect is a *Non-Favorable Effect*; otherwise, if $R_x^+ < R_x$, the dependency effect is a *Favorable Effect*. The difference between R_x and R_x^+, i.e. $|R_x - R_x^+|$, is called the *Degree of Dependency Effect*.

For risks ($I_x > 0$), a non-favorable effect will increase the probability of a risk and a favorable effect will lower its probability. On the contrary, for opportunities ($I_x < 0$), a non-favorable effect will lower the probability of an opportunity and a favorable effect will increase its probability. The degree of dependency effect of a risk represents the change of severity level due to the dependency effect applied to the risk.

3.4 Risk Dependencies in Program

In this section, we formally define the program risk and then identify several types of risk dependencies within a program.

3.4.1 Definition of Program Risk

According to our model, we divide program risks into two levels, *Project Level Risks* (*PJR*) and *Program Level Risks* (*PGR*). We next give a formal definition of the program risks:

Definition 4: Given a program $Q(t)$ which is consisted of q related projects, $Q(t) = \{Z_1(t), Z_2(t), \ldots, Z_q(t)\}$ with a set of Program Risks $QR(t)$ at time t, $QR(t) = PJR(t) \cup PGR(t)$ such that:

- $PJR(t)$ is a set of Project Level Risks and $PJR(t) = PJR(Z_1(t)) \cup PJR(Z_2(t)) \cup \ldots \cup PJR(Z_q(t))$, where $PJR(Z_i(t))$ represents a set of risks identified with project $Z_i(t), 1 \leq i \leq q$;
- $PGR(t)$ is a set of Program Level Risks and $\forall r \in PGR(t), r \notin PJR(t)$.

Every risk in a program is either a project level risk or a program level risk. A risk identified within any projects in a program is called a project level risk; all other risks are called program level risk. As illustrated in Figure 3.2, those risks identified in a project and affecting any objectives within the same project are project level risks; examples include the risks that may impact project schedule, project cost and so on. On the other hand, those risks identified outside the scope of project context but affect program objectives are program level risks; examples include the risks that will change the external business environment impacting any objectives of a program.

Although project level risks will only affect project objectives, they can still affect program objectives indirectly, as failing in meeting project objectives may affect the achievement of program objectives. In addition, as risk dependencies may exist between some risks, it is possible that the probability of occurrence of a risk is either increased or decreased by another risk within a program. Although project level risks will not directly affect any program objectives, it is possible that program level risks, which directly affect program objectives, are affected by risk dependencies of any risks (including project level risks) in a program.

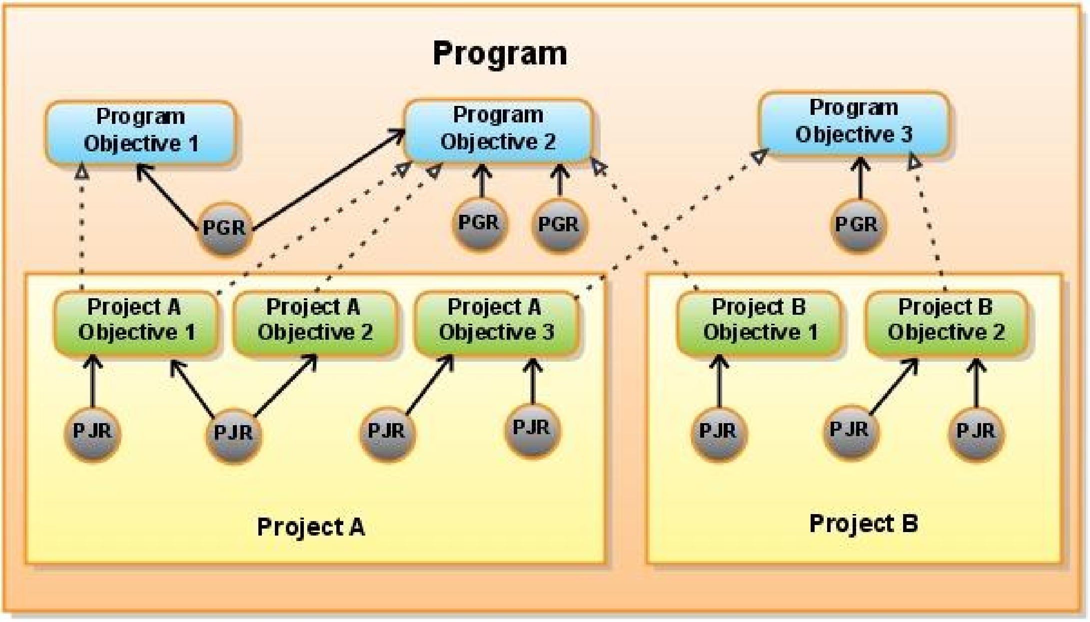


Figure 3.2: Program Risk Effects

3.4.2 Risk Dependency Types

The independent risks in a program are those risks that are not affected by any other risks; on the contrary, dependent risks are those risks that are affected by at least one risk within the program. Risk dependencies exist not only within a project but can exist between related projects. Also, program level risks may affect other project level or other program level risks within a program. As shown in Table 3.1, there can be four possible types of risk dependency in a program.

Table 3.1: Risk Dependency Types in Program

Type	Description	Owner
I	Risk dependency within a project	Project Manager
II	Risk dependency between projects	Program Manager
III	Risk dependency between a project level risk and a program level risk	Program Manager
IV	Risk dependency among program level risks	Program Manager

Among all possible risk dependencies, Type I involves only project level risks within the same project; Type II involves project level risks across different projects; Type III involves both project level and program level risks; and Type IV involves only program level risks. For the extreme case when there is only a single project in a program, there will not be any Type II risk dependency. Figure 3.3 presents an example and illustrates the possible types of risk dependencies.

In Figure 3.3, *PJR1*, *PJR3*, *PJR4*, *PJR6*, *PJR8*, *PGR1* and *PGR3* are independent program risks; *PJR2*, *PJR5*, *PJR7*, *PGR2* and *PGR4* are dependent program risks. It means that, in this case, the program has totally 7 independent program risks (i.e. the sum of 5 independent project level risks and 2 independent program level risks), and 5 dependent program risks (i.e. the sum of 3 dependent project level risks and 2 dependent program level risks).

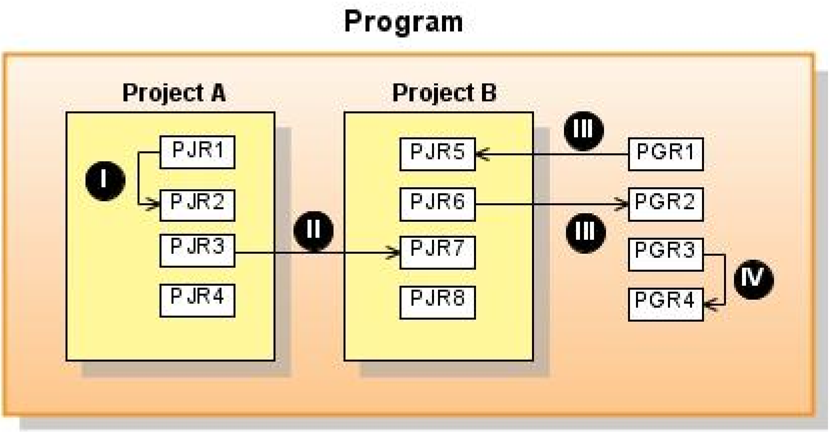


Figure 3.3: Risk Dependencies within a Program

3.5 Risk Dependency Graph

To represent risk dependency in a project or program, we define a *Risk Dependency Graph*, *RDG* to be a directed graph in which nodes represent risks and edges represent dependency between risks.

Definition 5: Given a set of identified risks $R(t) = \{R_1, R_2, \ldots, R_n\}$ at a given time t in a project, a risk dependency graph, $RDG = (N, D)$, where N is the set of nodes corresponding to the risks that have dependency, $N \subseteq R(t)$, and D is the set of edges representing the risk dependency relationships between the nodes, $D = \{(R_a, R_b) \mid R_a \rightarrow R_b, 1 \leq a, b \leq n, a \neq b\}$.

For a given risk, it may have dependency relationship (depending or being depended or both) with other risks or be independent of other risks. As the non-independent risks may not be necessary connected together, there can be more than one *RDG* in a project. Although *R(t)* refers to all identified risks in a single project at a given time, it can be extended to represent all program risks. Figure 3.4

shows an example of two *RDGs* in a project; one is formed by the seven risks of $\{R_1, R_2, R_4, R_5, R_7, R_8, R_9\}$ and another one is formed by the two risks $\{R_3, R_6\}$.

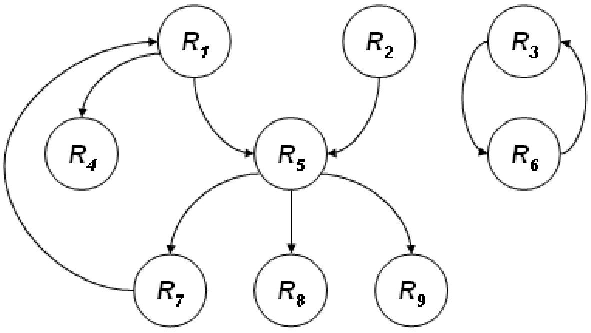


Figure 3.4: A Project with Two *RDG*s

Given a *RDG*, if there exist a path from R_a to R_b passing through one or more other risks, R_b is said to be an *Indirect Successor Risk* of R_a, and R_a is said to be an *Indirect Predecessor Risk* of R_b. For example, from the left *RDG* shown in Figure 3.5, R_4 and R_5 are the direct successors of R_1; R_4, R_5, R_7, R_8 and R_9 are all successors of R_1. Similarly, for risk R_9, R_5 is its only direct predecessor; R_1, R_2, R_5 and R_7 are all its predecessors. A *RDG* may contain cycles; the example formed by two risks $\{R_3, R_6\}$ shown in Figure 3.4 indicates that R_3 and R_6 are direct successors and direct predecessors of each other.

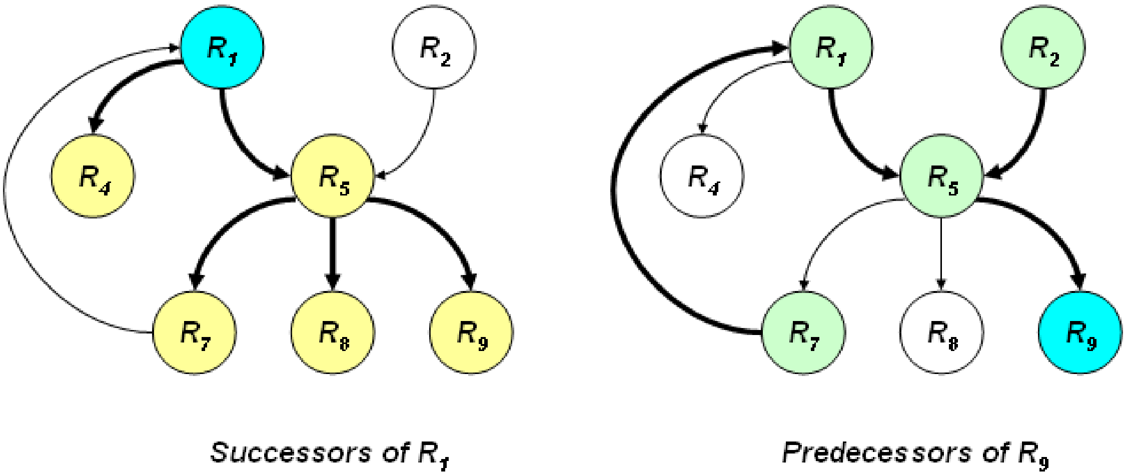


Figure 3.5: Successors and Predecessors in *RDG*

For any risk R_x in a *RDG*, the set of direct successors of R_x and the set of direct predecessors of R_x are defined as follows:

Definition 6: Given a Risk Dependency Graph $RDG = (N, D)$, for any $R_x \in N$, the Direct Successor of R_x, $Succ(R_x) = \{R_y \in N \mid (R_x, R_y) \in D\}$ and the Direct Predecessor of R_x, $Pred(R_x) = \{R_y \in N \mid (R_y, R_x) \in D\}$.

There are two risk dependency chains, *Successor Dependency Chain (SDC)* and *Predecessor Dependency Chain (PDC)*, which can help to locate all the successors and predecessors of a particular node in *RDG*. Both *SDC* and *PDC* are directed acyclic graphs constituted by the successors and predecessors of a particular risk respectively. The Breadth-first Search (BFS) algorithm (Cormen *et al*., 2001) can be utilized to find the dependency chains in a *RDG*. The algorithm firstly visits all the unexamined direct successors or direct predecessors of a selected risk in *RDG* and then traverses the graph along the selected dependency direction. Meanwhile, it also ensures that the dependency chains be acyclic by disregarding any potential feedback loops in *RDG*. The algorithm only visits the unexamined nodes when traversing the graph.

The following algorithm, *SDC-Gen*, shows how to find *SDC* for a risk R_x in *RDG*:

```
Algorithm SDC-Gen ($R_x$)
1.  Start with the node $R_x$ from RDG and put it as a root node in SDC
2.  Select the first unexamined node $R_i$ from SDC
    a.  Identify all the unexamined direct successors, $Succ(R_i)$, in RDG and put them into the SDC, if there are any
3.  If all nodes in SDC have been examined, then quit, else repeat from Step 2.
```

The following algorithm, *PDC-Gen*, shows how to find *PDC* for a risk R_x in *RDG*:

```
Algorithm PDC-Gen ($R_x$)
1.  Start with the node $R_x$ from RDG and put it as a root node in PDC
2.  Select the first unexamined node $R_i$ from PDC
    a.  Identify all the unexamined direct predecessors, $Pred(R_i)$, in RDG and put
        them into the PDC, if there are any
3.  If all nodes in PDC have been examined, then quit, else repeat from Step 2.
```

For the example shown in Figure 3.6, a *RDG* (including the risks of R_1, R_2, R_4, R_5, R_7, R_8 and R_9) is extracted from Figure 3.4. A $SDC(R_5)$ and a $PDC(R_5)$ for risk R_5 are obtained using algorithms *SDC-Gen* and *PDC-Gen,* respectively.

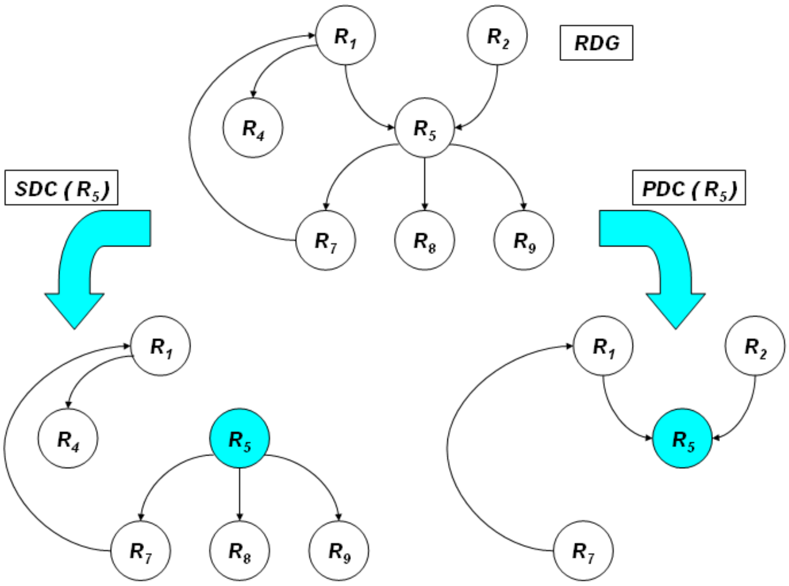


Figure 3.6: Risk Dependency Chains, *SDC* and *PDC*

As *SDC-Gen* and *PDC-Gen* apply the Breadth-first Search to find the dependency chains in a *RDG*, in the worst case, these algorithms need to traverse all possible nodes and all possible edges. Therefore, the time complexity of *SDC-Gen* and *PDC-Gen* is $O(|N|+|D|)$.

From the risk dependency graph, a number of useful metrics are defined for evaluating the extent of dependencies among identified risks at a given time. The first two metrics, the total number of Direct Successors (*NDS*) and the total number of Direct Predecessors (*NDP*), measure the dependency for a particular risk, while the next two metrics, the Total Risk Dependency Count (*TRDC*) and the Risk Dependency Index (*RDI*), measure dependency at the project level.

We will first consider the case of a single project. Later, we will present the case of multiple projects within a program.

Definition 7: Given a set of *n* identified risks of project *Z* at a given time *t*, $R(t) = \{R_1, R_2, \ldots, R_n\}$, for any $R_x \in R(t)$:

(3.14) $NDS(R_x) = |\,Succ(R_x)|$

(3.15) $NDP(R_x) = |\,Pred(R_x)|$

The total number of Direct Successors $NDS(R_x)$ of R_x reflects the number of risks that will be affected when R_x occurs. If $NDS(R_x) = 0$, it means that no other risks will be affected when R_x occurs. A high value of $NDS(R_x)$ indicates a high number of risks that are depending on R_x. In other words, if we could mitigate those risks R_x with a relatively high $NDS(R_x)$ value, it can help to reduce the total dependency effects of the project.

The total number of Direct Predecessors $NDP(R_x)$ of R_x reflects the number of risks that will affect R_x. If $NDP(R_x) = 0$, it means that no risks will affect R_x, and R_x can be independently considered when performing risk assessment. On the other hand, a high value of $NDP\ (R_x)$ means that many risks can affect the likelihood of occurrence of R_x.

Definition 8: Given a set of *n* identified risks of project *Z* at a given time *t*, $R(t) = \{R_1, R_2, \ldots, R_n\}$, for any $R_x \in R(t)$:

$$\text{(3.16)} \quad TRDC(Z) = \sum_{i=1}^{n} NDS(R_i) = \sum_{i=1}^{n} NDP(R_i) \quad \textit{where } 0 \leq TRDC(z) \leq n(n\text{-}1)$$

$$\text{(3.17)} \quad RDI(Z) = \frac{TRDC(Z)}{n\,(n-1)} \qquad \textit{where } 0 \leq RDI(Z) \leq 1$$

As a risk dependency involves two risks, the successor and the predecessor, the total number of direct successors of a project, i.e. $\sum_{i=1}^{n} NDS(R_i)$, and the total number of direct predecessors within the same project, i.e. $\sum_{i=1}^{n} NDP(R_i)$, should be the same. The *TRDC(Z)* is the total number of identified risk dependencies in project *Z*, which is equal to the total number of direct successors or the total number of direct predecessors in *Z*. If every risk is depended on every other risk, *TRDC(Z)* will reach its maximum value. If the total number of risks is *n*, there will be a maximum of *n-1* direct successors for any risk and thus the maximum of *TRDC(Z)* can be *n(n-1)*. On the other hand, if every risk is independent of other risks, *TRDC(Z)* will become zero.

For the case of multiple concurrent projects in a program, say *Q*, risk dependency may occur between a risk from one project to a risk from another project, or between a project level risk and a program level risk. In other words, the total number of direct predecessors of a project may not be equal to the total number of direct successors of the same project. To calculate the *TRDC* of a project *Z* within a program *Q*, only the direct successors within the project should be considered, i.e. *TRDC(Q, Z)* = the total number of direct successors in *Z* as shown in (3.18), as they will be affected due to the dependency. Similarly, the risk dependency index for the case of a project *Z* within a program *Q* is given in (3.19).

$$(3.18)\quad TRDC(Q,Z) = \sum_{i=1}^{n} NDS(R_i)$$

$$(3.19)\quad RDI(Q,Z) = \frac{TRDC(Q,Z)}{n\,(n-1)}$$ where *n* is the number of identified risk of *Z*

However, when considering all concurrent projects as a whole, the total number of direct successors is still equal to the total number of direct predecessors. *TRDC*, which varies from *0* to *n(n–1)*, can be used to analyze the trend of the number of risk dependencies in a project at different stages. However, as the number of risks at different project stages may vary, *TRDC* only counts the total number of dependencies, but cannot reflect the proportion of risk dependencies among all the risks at each project stage. Here, *RDI* helps to measure the trend of the degree of risk dependency of a project and compare the results between different project stages. Recall that *RDI* equals *0* when all the identified risks are independent, and equals *1* when every risk depends on every other risk. A high *RDI* value means that there are more dependencies among risks at a particular project stage, and this suggests more effort should be devoted in assessing the dependencies and selecting appropriate response strategies.

To calculate the *TRDC* and *RDI* at the program level, we can simply extend Definition 7 and 8, and let *R(t)* represent all program risks, where *n* is the total number of identified program risks. In this case, the total number of direct predecessors should be equal to the total number of direct successors within the same program, assuming that there will not be any risk dependency with a risk outside the program.

3.6 Summary

In this chapter, we have formally modeled the risk dependency and proposed methods to re-estimate risk by taking account of risk dependency effects. As there can be more than one risk affecting a particular risk, we have further proposed methods to estimate the combined effects. As the effects of risk dependencies can either increase or reduce the probabilities of those affected risks, we presented an assessment approach to determine whether an effect is favorable or non-favorable.

In addition, we have formally defined the program risk and identified different types of risk dependencies within a program, and a Risk Dependency Graph to represent risk dependencies in a project or program.

The risk dependency model that we proposed can address the needs of risk dependencies to be managed in practice. With the help of the model and various supporting methods, we can further enhance the risk management practices in the subsequent chapter.

Chapter 4 Risk Management Methodology with Risk Dependency

To effectively manage risk dependency, in this chapter, we enhance four basic risk management practices and propose five additional practices. The enhancements of basic practices involve how to evaluate and prioritize posterior risks (Section 4.2), develop posterior risk response plans (Section 4.2), monitor status of posterior risk and associated posterior risk response actions (Section 4.4), and control posterior risk response actions (Section 4.4). The enhanced practices are similar to the basic practices, except that the posterior risks are used for estimating the risk dependency effects.

Other than applying the concept of posterior risks, alternatively, we also propose additional novel risk management practices which directly deal with the dependencies between risks. The new practices are to identify risk dependencies (Section 4.1), evaluate and prioritize risk dependencies (Section 4.3), develop risk dependency response plans (Section 4.3), monitor status of risk dependencies and associated risk dependency response actions (Section 4.4), and control risk dependency response actions (Section 4.4).

With reference to the basic risk management practices listed in Table 2.4, the proposed changes are summarized in Table 4.1. The last column “Sect” identifies the subsequent sub-sections in which the enhanced and additional practices are discussed in detail.

Table 4.1: Enhanced Project Risk Management Practices for Risk Dependency

SEI Paradigm	Basic Risk Management Practices	Enhanced / *Additional* Risk Management Practices	Major Changes	Sect
Identify	1. Identify project risks	1a. Identify project risks	None	
		1b. *Identify risk dependencies*	New practice	4.1
Analyze	2. Evaluate & prioritize risks	2a. Evaluate & prioritize posterior risks	Use of posterior risks	4.2
		2b. *Evaluate & prioritize risk dependencies*	New practice	4.3
Plan	3. Develop risk response plans	3a. Develop posterior risk response plans	Use of posterior risks	4.2
		3b. *Develop risk dependency response plans*	New practice	4.3
Track	4. Monitor status of risk & associated risk response actions	4a. Monitor status of posterior risk & associated posterior risk response actions	Use of posterior risks	4.4
		4b. *Monitor status of risk dependencies & associated risk dependencies response actions*	New practice	4.4
Control	5. Control risk response actions	5a. Control posterior risk response actions	Use of posterior risks	4.4
		5b. *Control risk dependency response actions*	New practice	4.4

Lastly, in Section 4.5 and 4.6, we propose several metrics to be applied to analyze posterior risks and the effectiveness and performance of any associated response actions from project level and program level respectively.

4.1 Risk Dependency Identification

After all key risks have been identified, risk dependencies can be detected by examining each pair of risks within a project or across other concurrent projects in an organization, and determining whether there is any dependency relationship between them. Based upon the enhanced risk taxonomy for IT project (Kwan and Leung 2005), which consists of three major risk classes: *Product Engineering*, *Supporting Environments* and *Project Constraints*, Figure 4.1 shows six grouping of risk dependencies among different risk taxonomy classes.

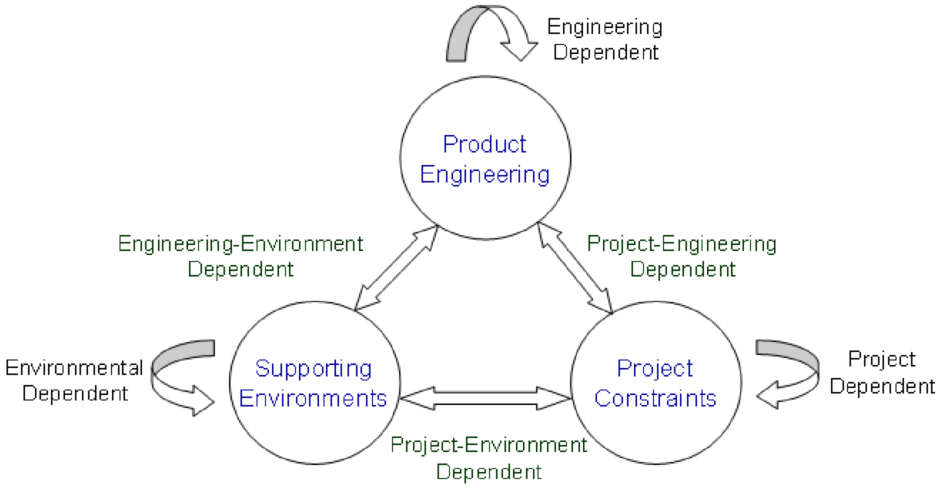


Figure 4.1: Risk Dependency Groups

The first three risk dependency groups refer to the risk dependencies existed within the same risk class, and the other three groups refer to the risk dependencies existed across two different risk classes. Table 4.2 illustrates the different risk dependency groups with examples of risk dependencies that may usually exist.

Table 4.2: Risk Dependency Groups and Examples

Risk Dependency Groups	**Descriptions**	**Examples**
Engineering Dependent	Dependency exists between two risks from the same risk class of "Product Engineering"	- Inter-system dependencies
Environmental Dependent	Dependency exists between two risks from the same risk class of "Supporting Environments"	- Sequential steps of development methodology - Use of development tools
Project Dependent	Dependency exists between two risks from the same risk class of "Project Constraints"	- Dependent activities in project plan - Shared resources for multiple tasks within a project - Shared resources for multiple projects

Project-Engineering Dependent	Dependency exists between a risk from the “Project Constraints” risk class and another risk from the “Product Engineering” risk class	- Deliverables involving efforts from multiple parties
Project-Environment Dependent	Dependency exists between a risk from the “Project Constraints” risk class and another risk from the “Supporting Environments” risk class	- Development involving internal or external parties
Engineering-Environment Dependent	Dependency exists between a risk from the “Product Engineering” risk class and another risk from the “Supporting Environments” risk class	- Shared development environments for different components of a system

There are two ways to summarize the risk dependencies. First, all identified risk dependencies can be represented using the *Risk Dependency Value Matrix*. As shown in Figure 4.2, the risk dependency value matrix is an *n* by *n* table for *n* identified risks; each cell in the matrix represents the risk dependency value, i.e. the D_{ab}, between the corresponding risks. If the value of a particular cell is equal to zero, it means that there is no dependency between the two corresponding risks at the given time. The diagonal values of the table will always be zero as they represent the self dependencies of each risk which should not exist by definition.

	R_1	***R_2***	***R_3***	**…**	***R_n***
R_1	***0***	D_{12}	D_{13}	…	D_{1n}
R_2	D_{21}	***0***	D_{23}	…	D_{2n}
R_3	D_{31}	D_{32}	***0***	…	D_{3n}
⋮	⋮	⋮	⋮	***0***	⋮
R_n	D_{n1}	D_{n2}	D_{n3}	…	***0***

Figure 4.2: Risk Dependency Value Matrix

Alternatively, the risk dependencies can also be represented by the *Risk Dependency*

Multiplication Matrix as shown in Figure 4.3. Each cell contains the value of the Risk Dependency Multiplier of the two corresponding risks. If there is no dependency between the risks at a given time, the value of the cell will be equal to 1. The diagonal values of the matrix are all 1s.

	R_1	R_2	R_3	…	R_n
R_1	**1**	DM_{12}	DM_{13}	…	DM_{1n}
R_2	DM_{21}	**1**	DM_{23}	…	DM_{2n}
R_3	DM_{31}	DM_{32}	**1**	…	DM_{3n}
⋮	⋮	⋮	⋮	**1**	⋮
R_n	DM_{n1}	DM_{n2}	DM_{n3}	…	**1**

Figure 4.3: Risk Dependency Multiplication Matrix

4.2 Posterior Risk Evaluation, Prioritization and Response

The matrices for prioritizing actions (Figure 2.5) and selecting the specific response actions (Figure 2.6) can be modified for assessing the posterior risks instead (as shown in Figure 4.4 and 4.5 respectively), i.e. the probability of a risk to be assessed is replaced by the probability of its posterior risk. The posterior risk assessment matrix (Figure 4.4) helps to determine the posterior risk level and prioritize the risk response actions. The posterior risk response matrix (Figure 4.5) helps to select appropriate response actions based upon the different combination of probability level and impact level of posterior risks. The same as before, Figures 4.4 and 4.5 have included the assessments for both risks (when $impact > 0$) and opportunities (when $impact < 0$).

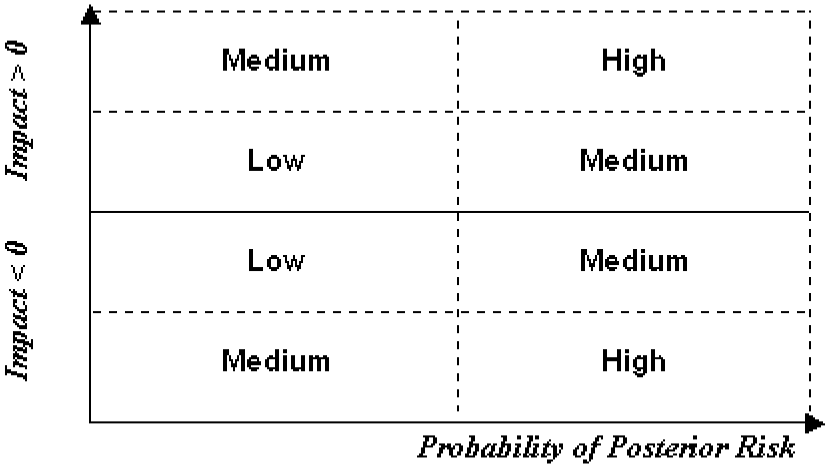


Figure 4.4: Posterior Risk Assessment Matrix

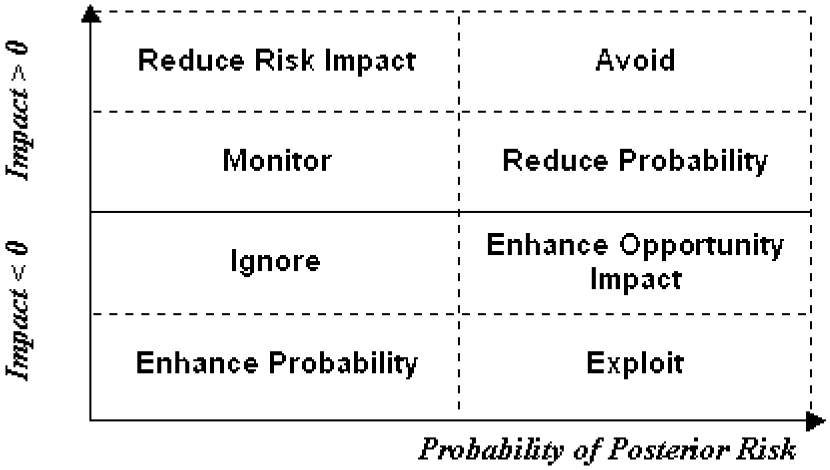


Figure 4.5: Posterior Risk Response Matrix

Here, the assessment focuses on the probability of posterior risk against the impact severity of the posterior risk.

4.3 Risk Dependency Evaluation, Prioritization and Response

Traditionally, project risk is assessed based upon their importance or urgency so that an organization can prioritize its resources to deal with the risks. As the risk dependencies can increase the probabilities of occurrence of risks, it is also worthy to develop strategies to directly deal with the dependencies, especially when the overall risk dependency index *RDI* is high, indicating that there are a large number of risk dependencies among risks in a project or program.

The major objective of risk dependency response plan is to reduce the effects of risk dependencies so that the probabilities of occurrence of those affected risks can be lower, and to enhance the effects of risk dependencies so that the probabilities of occurrence of those affected opportunities can be higher. Sometimes, reducing or enhancing the probabilities of occurrence of any direct predecessors of a risk can help to reduce or enhance the dependency effect to the risk. Based on the favorability of dependency effect and the degree of dependency effect, a Risk Dependency Response Matrix (Figure 4.6) is developed to help selecting the appropriate risk response strategies to a particular risk or opportunity.

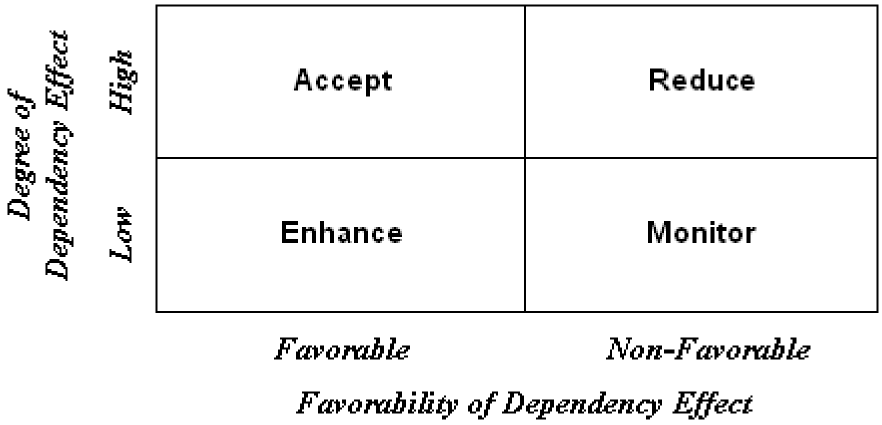


Figure 4.6: Risk Dependency Response Matrix

There are four response strategies:

- ***Reduce***: This strategy applies to the non-favorable but high degree of dependency effect, and its response actions should focus on reducing the dependency effect.

- ***Monitor***: This strategy applies to the non-favorable and low degree of dependency effect. As the dependency effect is small and further reducing the dependency effect may not be possible, the project team may just monitor for any effect changes.
- ***Enhance***. This strategy applies to the favorable but low degree of dependency effect, and its response actions should focus on increasing the dependency effect.
- ***Accept***. This strategy applies to the favorable and high degree of dependency effect. As it is the most desirable dependency effect, no action is required to respond to the effect.

However, it may not be possible or need to deal with all the dependencies in a project. Priority should be given to those dependencies with higher posterior risk severity levels due to the higher degree of risk dependency effect. The reason is that the risk dependency effect may not necessary cause a risk to become high risk, as the effect may decrease the likelihood of the occurrence of the risk; on the contrary, high risk may not be necessary caused by the risk dependency effect.

We determine the dependency response priority based upon the change of initial and posterior risk severity levels. Table 4.3 further expands the four response strategies (shown in Figure 4.6) with consideration of the different change combinations of risk severity level to determine the appropriate response priority to risk dependency.

Table 4.3: Priority of Response Actions to Risk Dependency

	Initial Severity Level	**Posterior Severity Level**	**Response Strategy**	**Response Priority**	**Description**

Risk *(I > 0)*	High	Medium	Enhance	Medium	The dependency effect is favorable and helps to lower the risk severity level; as the severity level of posterior risk is not low, some effort to enhance the dependency effect can be considered
	High	Low	Ignore	Low	The dependency effect is favorable and has already helped to reduce the risk severity level to low; there is no need to take any action
	Medium	High	Monitor	Medium	The dependency effect is small but it increases the risk severity level to high; some effort applied to the dependency effect can be considered
	Medium	Low	Enhance	Low	The dependency effect is favorable and has already helped to reduce the risk severity level to low; there is no need to take any action
	Low	High	Reduce	High	The dependency effect is large and significantly increases the risk severity level to high; there should be a highest priority to reduce the dependency effect
	Low	Medium	Monitor	Medium	The dependency effect is small and slightly increases the risk severity level to medium; some effort applied to the dependency effect can be considered
Opportunity *(I < 0)*	High	Medium	Monitor	Medium	The dependency effect is small but slightly reduces the severity of the opportunity to medium; some effort applied to the dependency effect can be considered
	High	Low	Reduce	High	The dependency effect is large and significantly reduces the severity of the opportunity to low; there should be a highest priority to reduce the dependency effect
	Medium	High	Enhance	Low	The dependency effect is favorable and has already helped to increase the severity of the opportunity to high; there is no need to take any action
	Medium	Low	Monitor	Medium	The dependency effect is small but it reduces the severity of the opportunity to low; some effort applied to the dependency effect can be considered
	Low	High	Ignore	Low	The dependency effect is favorable and has already helped to increase the severity of the opportunity to high; there is no need to take any action

	Low	Medium	Enhance	Medium	The dependency effect is favorable and helps to increase the severity of the opportunity; as the severity level of posterior risk is still not high, some effort to enhance the dependency effect can be considered

Table 4.3 does not show the cases when the initial severity level is equal to the posterior severity level. For those cases, dependency effect does not change the risk severity level; in other words, the dependency effect to the risks is minimal and no response action to the risk dependency is needed.

4.4 Monitor and Control of Posterior Risk and Risk Dependency

The enhanced practices of monitoring and controlling posterior risks and their associated response actions are similar to those basic risk management practices except that posterior risks are used. They are all on-going processes which are carried out continuously throughout the project life cycle to track any changes and identify any deviations from plans so that the most appropriate response actions can be selected.

The objectives of the additional practices for monitoring and controlling risk dependencies are also the same as those basic risk management practices except that the risk dependencies and their associated response actions are monitored and controlled: (1) monitoring the status of identified risk dependencies, and (2) monitoring and reviewing their associated risk dependency response actions. These additional practices apply equally to monitor and control of opportunities.

Due to changes of many project variables, such as project scope, objectives, environments, resources, or other external factors, the initially identified project risks and risk dependencies may change over time during the project life cycle. For different reasons, additional risks and risk dependencies may arise, and some existing risks and existing risk dependencies removed. Table 4.4 summarizes the possible changes and their main causes, and lists the additional actions that we propose to take in order to keep the information of project and program risks up to date.

Table 4.4: Possible Changes of Risks and Risk Dependencies

Cases	**Possible Changes to Risks**	**Main Causes of Changes**	**Actions**
1.	New risks identified	- Project or program changes	- If there are new risks, it is needed to review the relationship with other risks to see whether any new risk dependencies can be identified.
2.	Probabilities or impacts of risks changed	- Actions taken to deal with the risks - Predecessor risks have occurred	- If the probabilities of any risks changed, it may affect their direct successors as discussed previously. The probability values of their directly associated posterior risks may need to be re-calculated.
3.	Existing risks removed	- Actions taken to deal with the risks - Risks have occurred - Risks will not affect the project or program any more	- If any existing risks have been removed, all the dependencies between their direct successors or predecessors can be removed as well.
Cases	**Possible Changes to Risk Dependencies**	**Main Causes of Changes**	**Actions**
4.	New risk dependencies added	- New risk identified	- If any new risk dependencies are added, it is needed to determine the Risk Dependency Values (or the Risk Dependency Multipliers) of all their successors, and to find the posterior risks by re-calculating their probability values with the dependency effects.
5.	Risk dependency effect changed	- Actions taken to deal with risk dependencies	- Case 5 is similar to case 2. If the risk dependency effects (either positively or negatively) are changed, the probability values of their directly associated posterior risks may need to be re-calculated.

6.	Existing risk dependencies removed	- Concerned risks removed - Actions taken to deal with risk dependencies	- If any existing risk dependencies are removed, all associated posterior risks will reverse to their original risks (i.e. removing the effects from the risk dependencies).

4.5 Measuring Risk from Project Perspective

In Section 2.2.2, a number of metrics have been presented for monitoring the risks or opportunities. These metrics are enhanced to consider the risk dependency factor and the posterior risk from the perspective of single project in this section. In the next section, we will consider metrics for program.

We will assume that:

1. There are totally n risks identified at time t for project Z.
2. There is a set of m independent risks which do not have any direct predecessors, $R(t) = \{R_1, R_2, \ldots, R_m\}$ and $|R(t)| = m$.
3. There is another set of $n–m$ posterior risks which have one or more direct predecessors, $R^+(t) = \{ R^+_{m+1}, R^+_{m+2}, \ldots, R^+_n \}$ and $|R^+(t)| = n-m$.
4. The range of the risk levels of each independent risk or posterior risk is from 0 to M, i.e. $0 \le R_x \le M$ and $0 \le R^+_x \le M$, where M is the maximum possible value of R_x and R^+_x. For opportunities, the range will become ($-M, 0$). That is $-M \le R_x \le 0$ and $-M \le R^+_x \le 0$.
5. Let $R^*(t) = R(t) \cup R^+(t)$. Thus, $R_x \in R^*(t)$ for $1 \le x \le m$ and $R^+_x \in R^*(t)$ for $m+1 \le x \le n$. We will use R^*_x to represent either R_x or R^+_x.

4.5.1 Measuring Overall Project Risk

The total risk in $R(t)$ and $R^+(t)$ are $\sum_{x=1}^{m} R_x$ and $\sum_{x=m+1}^{n} R_x^+$ respectively. The project risks are measured by the posterior risk score $RS^+(t)$ and the averaged posterior risk score $ARS^+(t)$:

$$(4.1) \quad RS^+(t) = \sum_{x=1}^{m} R_x + \sum_{x=m+1}^{n} R_x^+ = \sum_{x=1}^{n} R_x^*$$

$$(4.2) \quad ARS^+(t) = \frac{1}{n} RS^+(t) = \frac{1}{n} \left(\sum_{x=1}^{n} R_x^* \right)$$

The risk index becomes posterior risk index $RI^+(t)$:

$$(4.3) \quad RI^+(t) = \frac{RS^+(t)}{n\ M} = \frac{1}{n\ M} \left(\sum_{x=1}^{n} R_x^* \right)$$

If there is no posterior risk in project Z at time t, the set of $R^+(t)$ will be empty and $m = n$. Equations (4.1), (4.2) and (4.3) can then be reduced and become equations (2.1), (2.2) and (2.3) respectively.

4.5.2 Measuring Top N Project Risk

Similarly, the risk score and risk index for the top N risks are enhanced to be the posterior risk score $RS_N^+(t)$ and posterior risk index $RI_N^+(t)$ for the top N risks. If there is a set of N risks $S^*(t) \subseteq R^*(t)$, and $|S^*(t)| = N$, such that $\forall R_a^* \, (R_a^* \geq R_b^* ;\ R_a^* \in S^*(t),\ R_b^* \in (R^*(t) - S^*(t)))$, $S^*(t)$ contains the top N risks of project Z at time t. $RS_N^+(t)$ and $RI_N^+(t)$ can be obtained from (4.4) and (4.5) respectively:

$$(4.4) \quad RS_N^+(t) = \sum_{x \in S^*(t)} R_x^*$$

$$(4.5)\quad RI_N^+(t) = \frac{RS_N^+(t)}{N\,M} = \frac{1}{N\,M}\sum_{x \in S^*(t)} R_x^*$$

If there is no posterior risk in the top *N* risks at time *t*, the set $R^+(t)$ will become empty and $S^*(t) \subseteq R(t)$. Equations (4.4) and (4.5) can then be reduced and become equations (2.4) and (2.5) respectively.

4.5.3 Measuring Project Risk by Project Objectives

The risk score and risk index that are segregated to express the different degree of effect on different project objectives can also be enhanced to become posterior risk score and posterior risk index. Assume that there are a set of *z* project objectives $OBJ = \{OBJ_1, OBJ_2, \ldots, OBJ_z\}$, and $I_x = \{I_{x1}, I_{x2}, \ldots, I_{xz}\}$ is a set of the impact value of R_x^* affecting the corresponding project objective $\{OBJ_1, OBJ_2, \ldots, OBJ_z\} \in OBJ$. That is, a risk could have different values of risk levels for various project objectives, i.e. $R_x^* = \{R_{x1}^*, R_{x2}^*, \ldots, R_{xz}^*\}$. The posterior risk score, $RS^{j+}(t)$, for a particular project objective OBJ_j, where $OBJ_j \in OBJ$, at a given time *t* becomes:

$$(4.6)\quad RS^{j+}(t) = \sum_{x=1}^{m} R_{xj} + \sum_{x=m+1}^{n} R_{xj}^+ = \sum_{x=1}^{n} R_{xj}^*$$

The posterior risk index, $RI^{j+}(t)$, with respect to a selected project objective OBJ_j at a given time *t* becomes:

$$(4.7)\quad RI^{j+}(t) = \frac{RS^{j+}(t)}{n\,M} = \frac{1}{n\,M}\left(\sum_{x=1}^{n} R_{xj}^*\right)$$

For a complete risk picture against all project objectives at a given time, we use the averaged posterior risk index $ARI^+(t)$:

$$(4.8) \quad ARI^{+}(t) = \frac{1}{z}\sum_{j=1}^{z} RI^{j+}(t)$$

If there is no posterior risk in project Z, the set $R^{+}(t)$ will become empty and $m = n$. Equations (4.6), (4.7) and (4.8) can be reduced and become equations (2.6), (2.7) and (2.8) respectively.

Note that project objectives may not always be independent and can be divided into sub-objectives (Asnar and Giorgini, 2006). In that case, the metrics presented in this section should be further extended.

4.6 Measuring Risks from Program Perspective

As the current risk metrics only focus on measuring risks within a project and lack the view of the risks at the program level, there is a need to develop risk metrics to measure risks from the program perspective. In this section, we will propose new risk metrics to measure the program risk levels taking into account the effect of dependencies. By expanding the risk dependency concepts and metrics used for projects, we propose a number of metrics to monitor program risks. The proposed metrics measure Overall Program Risk, Top *N* Program Risks, and Program Risk by Objectives.

We assume that:

1. At time t, a program $Q(t)$ contains q related projects, $Q(t) = \{Z_1(t), Z_2(t), \ldots, Z_q(t)\}$ and $|Q(t)| = q$.
2. There are totally β project level risks (i.e. the sum of all risks from all projects in the program) and δ program level risks identified at time t for program $Q(t)$, i.e. there is a total of $\beta + \delta$ risks in $Q(t)$.

3. There is a set of α independent project level risks in $Q(t)$. These risks do not have any direct predecessors. $Q\_PJR(t) = \{Q\_PJR_1, Q\_PJR_2, \ldots, Q\_PJR_\alpha\}$ and $|Q\_PJR(t)| = \alpha$.
4. There is another set of $\beta - \alpha$ posterior project level risks in $Q(t)$. These risks have one or more direct predecessors in $Q(t)$. $Q\_PJR^+(t) = \{Q\_PJR^+_{\alpha+1}, Q\_PJR^+_{\alpha+2}, \ldots, Q\_PJR^+_\beta\}$ and $|Q\_PJR^+(t)| = \beta - \alpha$.
5. $Q\_PJR^*(t) = Q\_PJR(t) \cup Q\_PJR^+(t)$. Thus, $Q\_PJR_x \in Q\_PJR^*(t)$ for $1 \leq x \leq \alpha$ and $Q\_PJR^+_x \in Q\_PJR^*(t)$ for $\alpha+1 \leq x \leq \beta$. We will use $Q\_PJR^*_x$ to represent either Q_PJR_x or $Q\_PJR^+_x$.
6. There is a set of γ independent program level risks in $Q(t)$. These risks do not have any direct predecessors. $Q\_PGR(t) = \{Q\_PGR_1, Q\_PGR_2, \ldots, Q\_PGR_\gamma\}$ and $|Q\_PGR(t)| = \gamma$.
7. There is another set of $\delta - \gamma$ posterior program level risks in $Q(t)$. These risks have one or more direct predecessors in $Q(t)$. $Q\_PGR^+(t) = \{Q\_PGR^+_{\gamma+1}, Q\_PGR^+_{\gamma+2}, \ldots, Q\_PGR^+_\delta\}$ and $|Q\_PGR^+(t)| = \delta - \gamma$.
8. $Q\_PGR^*(t) = Q\_PGR(t) \cup Q\_PGR^+(t)$. Thus, $Q\_PGR_x \in Q\_PGR^*(t)$ for $1 \leq x \leq \gamma$ and $Q\_PGR^+_x \in Q\_PGR^*(t)$ for $\gamma+1 \leq x \leq \delta$. We will use $Q\_PGR^*_x$ to represent either Q_PGR_x or $Q\_PGR^+_x$.
9. The range of the risk levels of independent risk or posterior risk is from 0 to M, i.e. $0 \leq Q\_PJR^*_x, Q\_PGR^*_x \leq M$, where $Q\_PJR^*_x \in Q\_PJR^*(t)$, $Q\_PGR^*_x \in Q\_PGR^*(t)$ and M is the maximum possible value of $Q\_PJR^*_x$ and $Q\_PGR^*_x$. For opportunities, the range will become $(-M, 0)$. That is $-M$

$\leq Q\_PJR_x^*,\ Q\_PGR_x^* \leq 0.$

4.6.1 Measuring Overall Risks in Program

To monitor risks in a project, as shown in (4.1) and (4.2), the posterior risk score and the averaged posterior risk score are used to represent an overall risk picture of a project. The calculations involve adding and averaging all risks in the project at a given time. However, the risk measures for a project do not reflect the overall program risk. Program managers need some metrics that can help them to monitor the overall program level risks that they are managing, and the overall project level risks that are managed by their project managers. We can add and average all the program level risks and all the project level risks to obtain the overall program level risk and the overall project level risk of a program respectively. Similarly, to monitor the risk level of the entire program, we can add and average all the program risks (including project level and program level risks) to obtain the overall risk level of a program.

Next, we propose metrics for measuring overall project level risk and overall program level risk in a program, and the metrics for the entire program.

4.6.1.1 Overall Project Level Risk in Program

The sum of project level risks in $Q\_PJR(t)$ and $Q\_PJR^+(t)$ are $\sum_{x=1}^{\alpha} Q\_PJR_x$ and $\sum_{x=\alpha+1}^{\beta} Q\_PJR_x^+$ respectively. The project level risks in program $Q(t)$ are measured by the posterior project level risk score $Q\_PJR\_RS^+(t)$:

$$(4.9a)\quad Q\_PJR\_RS^{+}(t) = \sum_{x=1}^{\alpha} Q\_PJR_x + \sum_{x=\alpha+1}^{\beta} Q\_PJR_x^{+} = \sum_{x=1}^{\beta} Q\_PJR_x^{*}$$

Alternatively, as each individual project level risk only belongs to one project within a program, the posterior project level risk score of a program should be equal to the sum of posterior risk score $RS_y^{+}(t)$ of all projects in *Q(t)*:

$$(4.9b)\quad Q\_PJR\_RS^{+}(t) = \sum_{y=1}^{q} RS_y^{+}(t)$$

The averaged posterior project level risk score $Q\_PJR\_ARS^{+}(t)$ and the posterior project level risk index $Q\_PJR\_RI^{+}(t)$ of program *Q(t)* will be:

(4.10)

$$Q\_PJR\_ARS^{+}(t) = \frac{1}{\beta} Q\_PJR\_RS^{+}(t) = \frac{1}{\beta} \sum_{x=1}^{\beta} Q\_PJR_x^{*} = \frac{1}{\beta} \sum_{y=1}^{q} RS_y^{+}(t)$$

(4.11)

$$Q\_PJR\_RI^{+}(t) = \frac{Q\_PJR\_RS^{+}(t)}{\beta\, M} = \frac{1}{\beta\, M} \sum_{x=1}^{\beta} Q\_PJR_x^{*} = \frac{1}{\beta\, M} \sum_{y=1}^{q} RS_y^{+}(t)$$

If there is only one project in program *Q(t)*, Equations (4.9b), (4.10) and (4.11) can be reduced and become Equations (4.1), (4.2) and (4.3) respectively.

4.6.1.2 Overall Program Level Risk

The sum of program level risks in *Q_PGR(t)* and $Q\_PGR^{+}(t)$ are $\sum_{x=1}^{\gamma} Q\_PGR_x$ and $\sum_{x=\gamma+1}^{\delta} Q\_PGR_x^{+}$ respectively. The program level risks in program *Q(t)* are measured by the posterior program level score $Q\_PGR\_RS^{+}(t)$:

$$(4.12)\quad Q\_PGR\_RS^{+}(t) = \sum_{x=1}^{\gamma} Q\_PGR_x + \sum_{x=\gamma+1}^{\delta} Q\_PGR_x^{+} = \sum_{x=1}^{\delta} Q\_PGR_x^{*}$$

The averaged posterior program level risk score $Q\_PGR\_ARS^{+}(t)$ and the posterior program level risk index $Q\_PGR\_RI^{+}(t)$ of program *Q(t)* will be:

$$(4.13)\quad Q\_PGR\_ARS^{+}(t) = \frac{1}{\delta} Q\_PGR\_RS^{+}(t) = \frac{1}{\delta} \sum_{x=1}^{\delta} Q\_PGR_{x}^{*}$$

$$(4.14)\quad Q\_PGR\_RI^{+}(t) = \frac{Q\_PGR\_RS^{+}(t)}{\delta\, M} = \frac{1}{\delta\, M} \sum_{x=1}^{\delta} Q\_PGR_{x}^{*}$$

Note that Equations (4.12), (4.13) and (4.14) are similar to Equations (4.1), (4.2) and (4.3), except that the program level risks are used in the calculations.

4.6.1.3 Overall Program Risk

As program risks consist of all project level risks and program level risks in a program, the posterior program risk score $Q\_RS^{+}(t)$ will be the sum of the posterior project level risk score $Q\_PJR\_RS^{+}(t)$ and the posterior program level risk score $Q\_PGR\_RS^{+}(t)$, i.e.

$$(4.15)\quad Q\_RS^{+}(t) = Q\_PJR\_RS^{+}(t) + Q\_PGR\_RS^{+}(t)$$

The averaged posterior program risk score $Q\_ARS^{+}(t)$ of program *Q(t)* will be the average of all project level and program level risks:

$$(4.16)\quad Q\_ARS^{+}(t) = \frac{Q\_RS^{+}(t)}{\beta + \delta}$$

The posterior program risk index $Q\_RI^{+}(t)$ of program *Q(t)* will be:

$$(4.17)\quad Q\_RI^{+}(t) = \frac{Q\_RS^{+}(t)}{(\beta + \delta)\, M} = \frac{1}{M} Q\_ARS^{+}(t)$$

4.6.2 Measuring Top N Risks in Program

In practice, due to the limited resources, program stakeholders may not always interest in all project level and program level risks, and often concentrate on those highest severity risks. As it is the responsibility of program managers to manage the program level risks and risk dependencies (Type II, III and IV) that are outside the authority of project managers, they need some metrics that can help them to track the top program level risks. In addition, program managers need to coordinate the risk responses among all the projects in a program, and it is useful that some metrics are available to monitor the top project risks among all projects. To consider a program as a whole, program managers also need some metrics that can monitor the top risks of the entire program. We next define metrics to measure the 3-tier top N risks: top N risks among all project level risks, top N risks among all program level risks and top N risks among all risks (including project level and program level risks) in a program.

4.6.2.1 Top N Project Level Risks in Program

If there is a set of N project level risks $Q\_PJS^{*}(t) \subseteq Q\_PJR^{*}(t)$, and $|Q\_PJS^{*}(t)| = N$, such that $\forall\ Q\_PJR_{a}^{*}$ ($Q\_PJR_{a}^{*} \geq Q\_PJR_{b}^{*}$; $Q\_PJR_{a}^{*} \in Q\_PJS^{*}(t)$, $Q\_PJR_{b}^{*} \in (Q\_PJR^{*}(t) - Q\_PJS^{*}(t))$), $Q\_PJS^{*}(t)$ contains the top N project level risks in program $Q(t)$. The posterior risk score and posterior risk index for the top N project level risks in $Q(t)$, $Q\_PJR\_RS_{N}^{+}(t)$ and $Q\_PJR\_RI_{N}^{+}(t)$, can be obtained from (4.18) and (4.19) respectively:

$$(4.18)\quad Q\_PJR\_RS_{N}^{+}(t) = \sum_{x \in Q\_PJS^{*}(t)} Q\_PJR_{x}^{*}$$

$$(4.19)\quad Q\_PJR\_RI_N^+(t) = \frac{Q\_PJR\_RS_N^+(t)}{N\ M} = \frac{1}{N\ M}\sum_{x\in Q\_PJS^*(t)} Q\_PJR_x^*$$

The calculations of posterior risk score and posterior risk index for the top *N* project level risks in a program are similar of those metrics for projects, i.e. (4.4) and (4.5), except that only top project level risks being considered in (4.18) and (4.19). For the case that all Top *N* project level risks come from a single project, the posterior risk score and posterior risk index for the top *N* project level risks of a program are the same as those for this particular project; it implies that this project should receive the most attention among all the related projects in the program.

4.6.2.2 Top N Program Level Risks

If there is a set of *N* program level risks $Q\_PGS^*(t) \subseteq Q\_PGR^*(t)$, and $|Q\_PGS^*(t)| = N$, such that $\forall Q\_PGR_a^* (Q\_PGR_a^* \geq Q\_PGR_b^*;\ Q\_PGR_a^* \in Q\_PGS^*(t)$, $Q\_PGR_b^* \in (Q\_PGR^*(t) - Q\_PGS^*(t)))$, $Q\_PGS^*(t)$ contains the top *N* program level risks in program $Q(t)$. The posterior risk score and posterior risk index for the top *N* program level risks in $Q(t)$, $Q\_PGR\_RS_N^+(t)$ and $Q\_PGR\_RI_N^+(t)$, can be obtained from (4.20) and (4.21) respectively:

$$(4.20)\quad Q\_PGR\_RS_N^+(t) = \sum_{x\in Q\_PGS^*(t)} Q\_PGR_x^*$$

$$(4.21)\quad Q\_PGR\_RI_N^+(t) = \frac{Q\_PGR\_RS_N^+(t)}{N\ M} = \frac{1}{N\ M}\sum_{x\in Q\_PGS^*(t)} Q\_PGR_x^*$$

The calculations of posterior risk score and posterior risk index for the top *N* program level risks in a program are similar of those metrics for projects, i.e. (4.4) and (4.5), except that only top program level risks being considered in (4.20) and (4.21).

4.6.2.3 Top N Program Risks

For the top N program risks, we assume that there is a set of N risks among all project level and program level risks in $Q(t)$, $Q\_S^*(t) \subseteq (\, Q\_PJR^*(t) \cup Q\_PGR^*(t)\,)$, and $|Q\_S^*(t)| = N$, such that $\forall\, Q\_R_a^*\, (\, Q\_R_a^* \geq Q\_R_b^*\, ;\ Q\_R_a^* \in Q\_S^*(t),\ Q\_R_b^* \in (Q\_PJR^*(t) \cup Q\_PGR^*(t) - Q\_S^*(t)))$. In other words, $Q\_S^*(t)$ contains the top N risks in program Q at a given time t. The posterior risk score and posterior risk index for the top N risks in $Q(t)$, $Q\_RS_N^+(t)$ and $Q\_RI_N^+(t)$ can be obtained from (4.22) and (4.23) respectively:

$$(4.22)\quad Q\_RS_N^+(t) = \sum_{x \in Q\_S^*(t)} Q\_PJR_x^* + \sum_{x \in Q\_S^*(t)} Q\_PGR_x^*$$

$$(4.23)\quad Q\_RI_N^+(t) = \frac{Q\_RS_N^+(t)}{N\ M}$$

The calculations of posterior risk score and posterior risk index for the top N program risks in a program are similar of those metrics for projects, i.e. (4.4) and (4.5), except that all top risks in a program being considered in (4.22) and (4.23). In the case that all Top N program risks come from project level risks, Equations (4.22) and (4.23) become (4.18) and (4.19) respectively. For the case that all Top N program risks come from program level risks, Equations (4.22) and (4.23) become (4.20) and (4.21) respectively.

4.6.3 Measuring Program Risks by Program Objectives

The impacts of project risks can be defined as a set of relative impact values affecting one or more project objectives. Similarly, the impacts of program risks can also be defined as a set of relative impact values affecting program objectives. Thus, the risk index can be segregated to express the effects on different program objectives. The program managers can then prioritize the risk response actions based on the extent to which the risk impacts the most important program objectives.

Among all the program risks, only program level risks will directly affect program objectives; project level risks can only affect program objectives indirectly by the Type III risk dependency. The posterior risk scores and posterior risk indexes of program level risks in a program can be segregated to express the different degree of effect on different objectives of the program. As the posterior program level risks are used in the calculations of the posterior risk scores and posterior risk indexes, the scores and indexes have already included the indirect effects from the project level risks.

Assume that there are a set of λ program objectives $Q\_OBJ = \{Q\_OBJ_1, Q\_OBJ_2, \ldots, Q\_OBJ_\lambda\}$, and $Q\_I_x = \{Q\_I_{x1}, Q\_I_{x2}, \ldots, Q\_I_{x\lambda}\}$ is a set of impact values of a program level risk $Q\_PGR_x^*$ affecting the corresponding program objective $\{Q\_OBJ_1, Q\_OBJ_2, \ldots, Q\_OBJ_z\} \in Q\_OBJ$.

A program level risk could have different values of risk levels with respect to various program objectives, i.e. $Q\_PGR_x^* = \{ Q\_PGR_{x1}^*, Q\_PGR_{x2}^*, \ldots, Q\_PGR_{x\lambda}^* \}$. The posterior program risk score, $Q\_RS^{j+}(t)$, for a particular program objective Q_OBJ_j, where $Q\_OBJ_j \in Q\_OBJ$, is then the sum of all risk scores of Q_OBJ_j at a given time t:

$$(4.24)\quad Q\_RS^{j+}(t) = \sum_{x=1}^{\gamma} Q\_PGR_{xj} + \sum_{x=\gamma+1}^{\delta} Q\_PGR_{xj}^{+} = \sum_{x=1}^{\delta} Q\_PGR_{xj}^{*}$$

The posterior program risk index, $Q\_RI^{j+}(t)$, with respect to a selected program objective Q_OBJ_j at a given time t becomes:

$$(4.25)\quad Q\_RI^{j+}(t) = \frac{Q\_RS^{j+}(t)}{\delta\ \ M} = \frac{1}{\delta\ \ M}\left(\sum_{x=1}^{\delta} Q\_PGR^{*}_{xj}\right)$$

For a complete risk picture against all program objectives at a given time *t*, we use the averaged posterior program risk index $Q\_ARI^{+}(t)$:

$$(4.26)\quad Q\_ARI^{+}(t) = \frac{1}{\lambda}\sum_{j=1}^{\lambda} Q\_RI^{j+}(t)$$

Similar to project objectives, program objectives may not always be independent and can be divided into sub-objectives. The metrics presented in this section should be further extended to address these cases.

Chapter 5 Case Study

In this chapter, we present an empirical study on employing the risk dependency concept in managing project and program risks. The objectives of this study are to demonstrate how the risk dependency can be applied in real life IT projects, and validate whether the dependency can help to evaluate and prioritize risks in a manner more reflecting reality.

We will first provide some background information about the program, especially the relationship of the objectives between the program and its three related projects, and present the risk management practices of the organization. Afterwards, we will present the program risks identified at different project stages by various parties, and the risk dependencies established within each project and across projects. Next, we obtain the posterior risks of each concerned risk. We will also present various posterior risk scores and discuss the risk trends. Finally, we conclude by summarizing the benefits achieved, challenges and lessons learnt.

5.1 Background

Program Q consisted of three parallel projects, namely, Project A, Project B, and Project C, which were managed by three different project teams. All three projects were outsourced projects, which aimed to enhance three existing systems, namely, System X, System Y, and System Z respectively. System X is a secure environment for various types of customers to submit, manage and maintain their documents, System Y captures the customer submitted documents from System X and publishes them automatically onto designated websites, and System Z provides flexible means to capture, categorize, package and distribute the subscribed information to customers through pre-defined message formats. These projects involved complicated system environments consisting of various installed system software, data interfaces and replication, network devices, business and user workflows. As the development of these systems were all contracted out to different vendors, effectively managing risks became critical to the program success.

The organization had budgeted HK$2.87M for Project A, HK$0.95M for Project B and HK$0.8M for Project C. The project teams had the responsibility to closely monitor the activities performed by the vendors, who were required to perform all major development activities including:

- ➢ Update project plans and project status;
- ➢ Prepare detailed functional requirements;
- ➢ Perform system designs;
- ➢ Develop the application systems;
- ➢ Conduct technical walkthrough sessions;
- ➢ Prepare system and integration test plans, test cases and test results;
- ➢ Establish acceptance testing environments and provide on-site support during tests;
- ➢ Establish production environments;
- ➢ Prepare release and fallback plans;
- ➢ Conduct training for users, administrators, operational staff and technical staff;
- ➢ Perform all system documentation updates.

According to the project natures, there was a project constraint that System X of Project A could not be launched before System Y of Project B and System Z of Project C were ready. However, the constraint did not cause any major difficulty, as the estimated duration of Project A was about 1 year, and that of Project B and C were only about 7 months each. Risk dependencies existed among the projects will be discussed in the subsequent sections.

The goals of Program Q were to phase out all the manual processes and cater to the needs of Chinese customers. The organization identified the three projects with specific project-level objectives to advance the program objectives, as shown in Figure 5.1. Although the projects were individually managed by three different project teams, they shared the same business objective of the program which was to provide a more effective way for customers to manage and distribute their company news and related information.

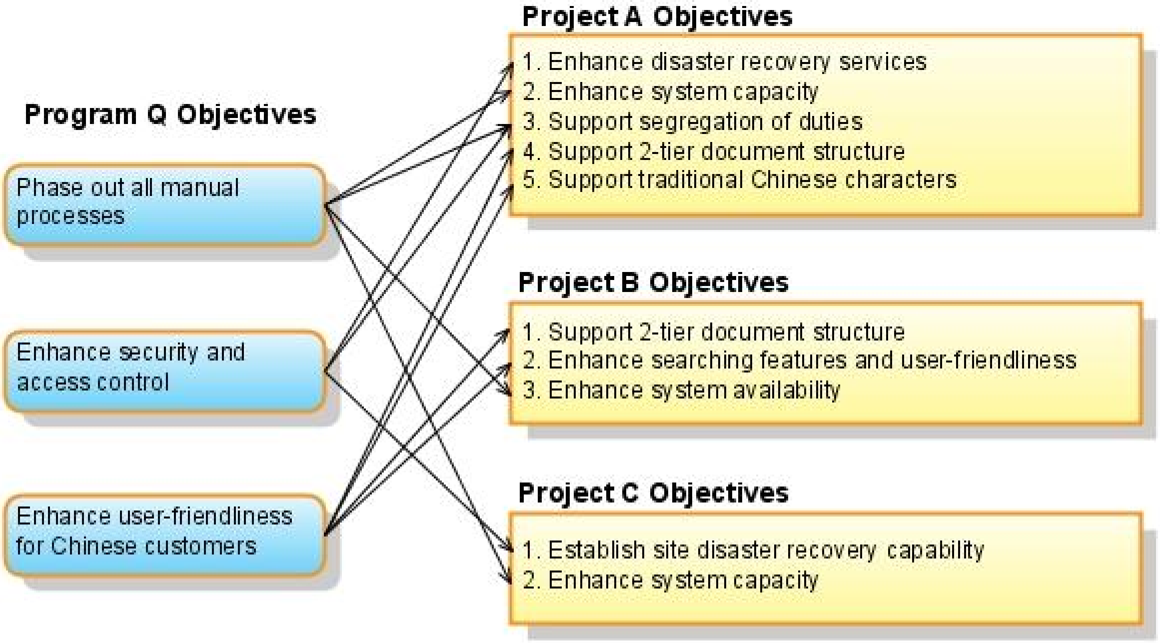


Figure 5.1: Mapping of Objectives between Program and Three Projects

5.1.1 Program and Project Organizational Structure

A Project Steering Committee (PSC), which consisted of project sponsors, system owners, IT support and business managers, was formed to oversee the entire program. On the IT side, IT Management Committee (ITMC), which was led by IT Head who was responsible to oversee all IT systems, had also requested Quality Assurance (QA) team to serve as an independent party to review the projects, and offer improvement recommendations. Figure 5.2 highlights the relationship between the three projects interfaced with IT supports, and Table 5.1 summarizes the members of each team and their major responsibilities.

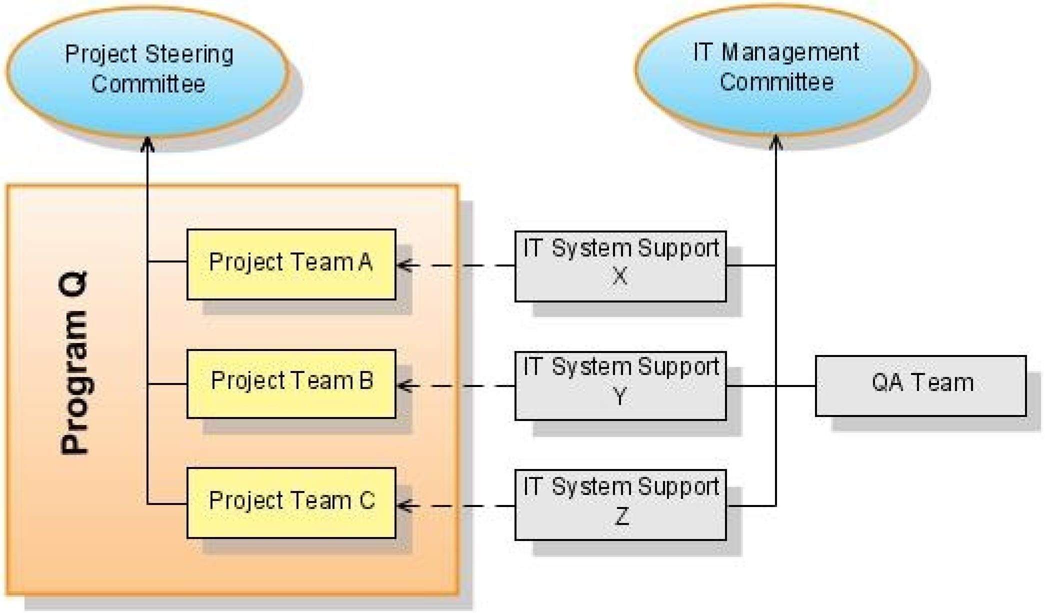


Figure 5.2: Program and Project Structure

Table 5.1: Project Team Members and Responsibilities

	Members	**Responsibility**
Project Steering Committee (PSC)	Project sponsors, system owners, IT support managers and business managers	- Oversee all three projects - Make final decisions and approvals
Project Team A	IT System X support managers and responsible business managers	- Manage Project A and report Project A status to PSC

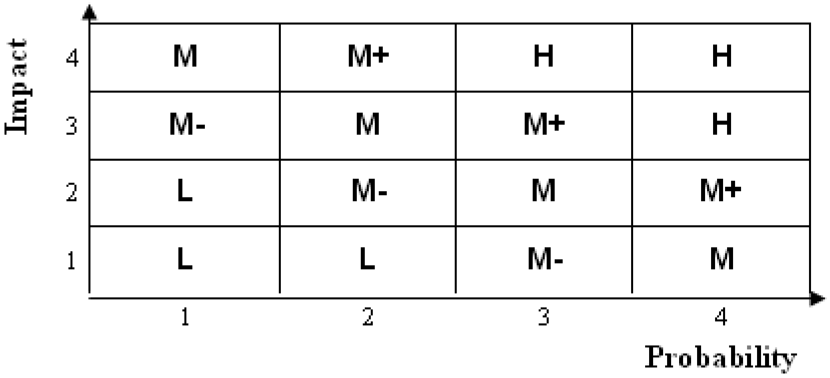
Project Team B	IT System Y support managers and responsible business managers	- Manage Project B and report Project B status to PSC
Project Team C	IT System Z support managers and responsible business managers	- Manage Project C and report Project C status to PSC
IT Management Committee (ITMC)	IT Head and all IT managers in various areas	- Oversee all IT projects
IT System Support X	IT System X support managers and concerned technical staff	- Manage all projects related to System X and report any issues to ITMC
IT System Support Y	IT System Y support managers and concerned technical staff	- Manage all projects related to System Y and report any issues to ITMC
IT System Support Z	IT System Z support managers and concerned technical staff	- Manage all projects related to System Z and report any issues to ITMC
Quality Assurance (QA) Team	Quality assurance analysts	- Review and identify risks and make recommendations to project teams

5.1.2 Organizational Risk Management Practices

All three projects adopted the same common practices of risk identification and management:

- At the beginning of the program, risks were initially identified at the program level by all related project sponsors and system owners. If the risks were not related to specific projects, they would be managed at the program level; otherwise they would be passed to each individual project teams to follow up.
- At the project initiation stage T1 of each project, more risks were identified by individual project sponsors and system owners.
- At the project planning and system design stage T2, the system development stage T3 and the system testing stage T4, risks were identified by IT system support managers and quality assurance analysts.

Once identified, all risks were initially evaluated and added into a centralized risk register. The evaluations were performed by the risk originators (i.e. the one who identified the risks) using a Ranking Method, which was based on the different combination of assigned probability and impact values. According to the pre-defined risk assessment matrix used by the organization, each risk was assessed by assigning appropriate probability and impact values, both ranged from *1* to *4*, where a higher probability value representing a higher chance that the risk would occur, and a higher impact value representing a higher negative effect that the risk would impact the project. Based on the different combination of assigned probability and impact values, a risk's severity level and its response priority were determined before any response actions could be planned. According to the pre-defined risk assessment matrix used by the organization as shown in Figure 5.3, each risk was then classified as Low Risk (*L*), Medium Low Risk (*M-*), Medium Risk (*M*), Medium High Risk (*M+*) or High Risk (*H*); in addition, as for calculating risk scores, each risk classification was further assigned a pre-defined score from *1* (low risk) to *5* (high risk).



Impact				
4	M	M+	H	H
3	M-	M	M+	H
2	L	M-	M	M+
1	L	L	M-	M
	1	2	3	4
				Probability

Figure 5.3: Risk Assessment Matrix

As the QA Team of the organization and an independent party of the projects, we reviewed each risk and determined whether there were any risk dependencies. The results of this exercise and any recommendations being made were feedbacked to the project teams for consideration. If the project teams accepted the recommendations and changed their initial evaluation results of the affected risks, we could then confirm the existence of the risk dependencies. In addition, for each risk identified within a project, the respective project teams would determine whether the risk might impact any other systems. The specific risks were then communicated to the concerned project teams for their review. Based on that information, we identified the risk dependencies between risks across different projects. After dependencies were determined between risks, those affected risks were re-estimated and were re-evaluated based on the new values of posterior risks. Afterwards, the respective project teams planned and executed risk response actions accordingly. Besides, the PSC was the focal point to plan, manage and coordinate the risks at the program level. If the planned risk response actions were related to the responsibilities of any project teams, the actions would be executed by the specific project teams; otherwise, the committee would look for other resources outside the project teams.

5.2 Identified Risks and Risk Dependencies

Totally 40 program risks were initially identified at the beginning of the program, in which 34 project level risks were passed to individual project teams for follow-up, and 6 program level risks were managed at the program level. The program level risks mainly related to the support of organizational infrastructure, the external and internal operational faults, and the supporting resources for initial launch of all involved systems. Table 5.2 lists the number of risks identified within the program at different project stages. We observed that program level risks were identified only at T1; it was because the risks identified at program level at the other project stages were all project specific and had been counted as project level risks. Table 5.3 lists the number of risk dependencies that were identified at different project stages within each project, as well as the number of risk dependencies between projects.

Table 5.2: Number of Identified Program Risks

Program / Project	**Risk Categories**	**T1**	**T2**	**T3**	**T4**	**Total**
Project A	Product Engineering	5	9	8	1	23
	Supporting Environments	4	4	-	5	13
	Project Constraints	6	1	-	-	7
	Total	**15**	**14**	**8**	**6**	**43**
Project B	Product Engineering	3	8	5	1	17
	Supporting Environments	5	4	-	5	14
	Project Constraints	3	1	-	-	4
	Total	**11**	**13**	**5**	**6**	**35**
Project C	Product Engineering	3	10	3	1	17
	Supporting Environments	4	4	-	5	13
	Project Constraints	1	5	-	-	6
	Total	**8**	**19**	**3**	**6**	**36**
Project Level (*Project A + Project B + Project C*)		34	46	16	18	114
Program Level		6	0	0	0	6
Program Q Total (*Project Level + Program Level*)		**40**	**46**	**16**	**18**	**120**

Table 5.3: Number of Identified Risk Dependencies

	Program / Project	**Risk Dependency Groups**	**T1**	**T2**	**T3**	**T4**	**Total**
Type I	**Project A**	Engineering Dependent	-	2	2	-	4
		Environmental Dependent	-	-	-	1	1
		Total	**0**	**2**	**2**	**1**	**5**
	Project B	Environmental Dependent	-	-	-	2	2
		Project-Engineering Dependent	-	1	-	-	1
		Total	**0**	**1**	**0**	**2**	**3**
	Project C	Environmental Dependent	-	-	-	1	1
		Project-Engineering Dependent	1	1	-	-	2
		Total	**1**	**1**	**0**	**1**	**3**
Type II	**Projects A & B**	Engineering Dependent	-	2	-	-	2
		Environmental Dependent	-	2	1	-	3
		Project Dependent	1	-	-	-	1
		Total	**1**	**4**	**1**	**0**	**6**

	Projects B & C	Engineering Dependent	-	3	2	-	5
		Environmental Dependent	2	2	-	-	4
		Project Dependent	1	-	-	-	1
		Total	**3**	**5**	**2**	**0**	**10**
	Projects A & C	N.A.	-	-	-	-	0
		Total	**0**	**0**	**0**	**0**	**0**
Program Q		**Total**	**5**	**13**	**5**	**4**	**27**

Figure 5.4 presents a complete picture of the risk dependency graphs of the program, where the nodes representing the risks identified in the projects and the arrows representing the dependencies between risks. Some risks are direct successors and direct predecessors of each other. For example, two risks resided in two different projects had the same concern of estimating the system capacity during peak hours.

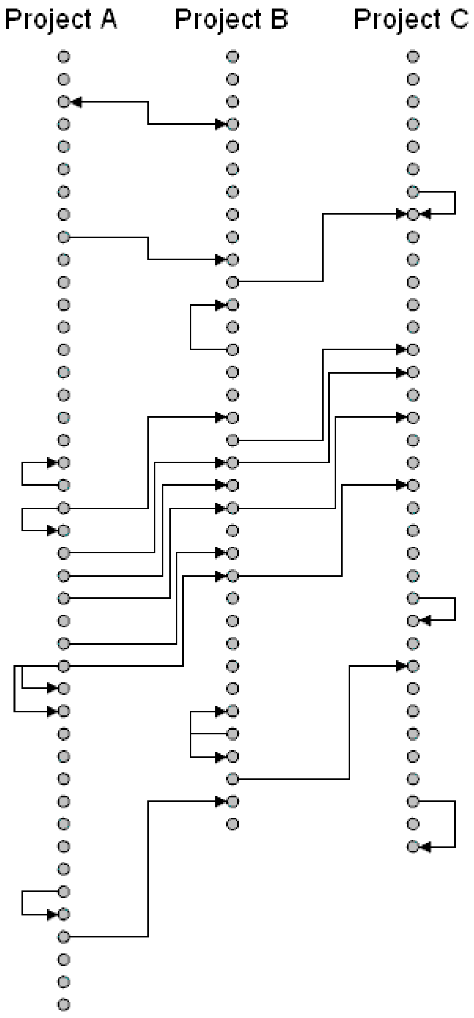


Figure 5.4: Risk Dependency Graphs

Some observations and collected feedbacks regarding to the risk dependency identification exercise are summarized as followed:

- No Type III and IV risk dependencies were determined between project level risks and program level risks. For Type III dependency, according to the existing practices of the organization, any program level risks that would affect project level risks had been fully assessed at the program level in the very beginning of the program; therefore, the initially assigned probability values of the concerned risks at the project level could have considered the effects of the risks from program level at the time of assessment. For Type IV dependency, as many program level risks had already assumed that the associated project level risks would occur, no additional risk dependencies could be determined.
- Although risks were identified at different stages of a project, Type I risk dependencies were usually determined between risks which were identified in the same project stage. This is expected as risks identified in the later project stages can hardly affect the risks identified in the earlier project stages. On the contrary, for those risks identified at the earlier project stages, if their risk severity levels are high, appropriate response actions should have been taken, and therefore, they should not have much impact on those risks occurred at the later project stages.
- For each project, we often identified a number of relationships between those identified risks; however only a small set of these relationships were determined as dependencies. By definition, a risk dependency will have an effect on the probability value of a risk if its predecessor risk does occur. According to the comments received, when a staff identified a risk, he/she

would have already considered every possible factor (including other risks that he/she had identified) before assigning the probability value of the risk. In fact, all the dependencies determined in the case study were formed by risks that were identified by different persons.

- Most of the risk dependencies occurred in the same risk categories. As there was a lack of effective risk dependency identification approach, project teams tended to only focus on the dependencies within the same risk categories, such as interface dependencies between systems and task dependencies in project schedules. If the identification approach could be improved, more dependencies might be identified across risk categories.
- There were more cross-project risk dependencies (i.e. Type II) than those within the same projects (i.e. Type I). This finding supports the observation made earlier that dependencies could rarely be determined between the risks that were identified by the same person within the same project. Risks across projects often need input from the other project teams. For the same reason, as all program level risks were identified by the same group (i.e. the Project Steering Committee), no risk dependencies could be determined between program level risks.
- Other than risk dependencies, as project teams had the opportunities to learn from the risks reported by other interdependent projects, they could then identify additional similar risks that were initially neglected due to different reasons.

5.3 Risk and Posterior Risk Evaluation

The risk assessment is an important step in risk management, as it provides the base for project teams to select the most appropriate response strategies to the risks and prioritize their response actions. As mentioned earlier, all risks were initially evaluated (i.e. risk evaluation) right after they were identified. After dependencies were determined between risks, those concerned risks (i.e. all direct successor risks) were re-evaluated (i.e. posterior risk evaluation) based on the values of posterior risks. Figure 5.5 shows the number of risks identified at different project stages and grouped by different severity levels for the three projects. It also presents the risk distribution after the posterior risk assessment.

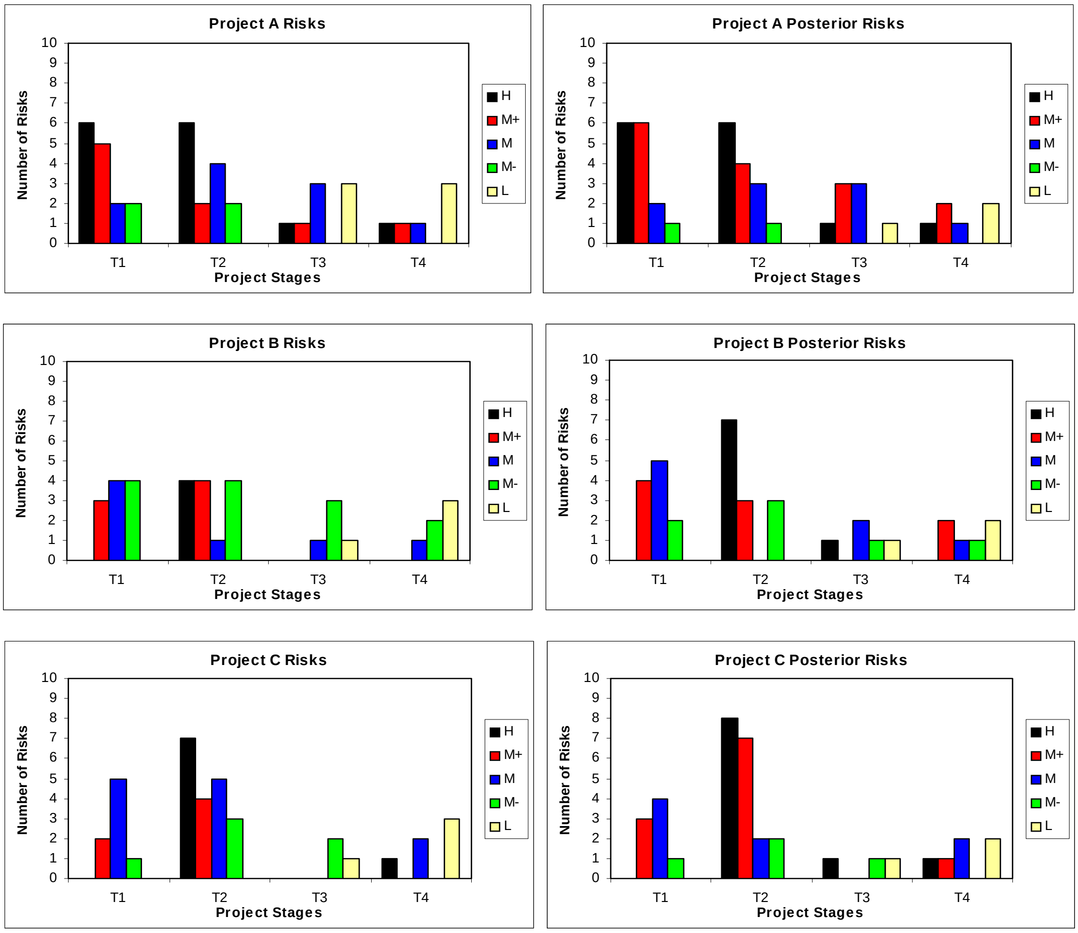


Figure 5.5: Distribution of Risks and Posterior Risks at Different Severity Levels

The risk value and posterior risk value of each identified risk in different project stages were averaged out and listed in Table 5.4. This table shows the average differences between risks and posterior risks after risk re-evaluation. Note that all dependency effects are non-favorable, and they increase the average risk values at each project stage by a minimum of +0.12 at T1 of Project C to a maximum of +1.00 at T3 of Project C.

Table 5.4: Comparison of Averaged Risk Values and Averaged Posterior Risk Values

Projects	Project Stages	Averaged Risk Values	Averaged Posterior Risk Values	Differences
Project A	**T1**	4.00	4.13	+0.13
	T2	3.86	4.07	+0.21
	T3	2.63	3.38	+0.75
	T4	2.50	3.00	+0.50
Project B	**T1**	2.91	3.18	+0.27
	T2	3.62	4.08	+0.46
	T3	2.00	2.80	+0.80
	T4	1.67	2.50	+0.83
Project C	**T1**	3.13	3.25	+0.12
	T2	3.79	4.11	+0.32
	T3	1.67	2.67	+1.00
	T4	2.33	2.83	+0.50

It can be observed that the risk originators had adjusted their risk response strategies after they learned that their identified risks were affected by other risks. There were about 14%, 20% and 19.4% of risks for projects A, B and C respectively that were adjusted in this re-evaluation exercise. A number of observations are summarized below regarding the risk severity levels:

- For the three projects, most of the critical risks (with risk severity “High” or “Medium High”) occurred in the first two project stages, T1 and T2. The situation was understandable as the activities of these stages mainly focused on project planning in which more critical risks could be identified. Besides, as mitigation actions had been taken for the critical risks, less critical risks remain in later stages. This is the reason that the average risk severity levels in T3 and T4 were generally lower than those in T1 and T2.

- The severity levels of posterior risks in all project stages of all projects were higher than the severity levels of initially evaluated risks; it implied that risk dependencies existed in all the project stages, and contributed to the difference in the severity levels between risks and posterior risks. In fact, as all the risk dependency values were positive, this increased the probability values of all the direct successors and increased the overall severity levels of posterior risks.
- Among all the project stages, T2 had the highest averaged risk severity levels, and had the highest numbers of risk dependencies identified for all three projects. Most of those dependencies were related to two risk classes: Product Engineering and Supporting Environments. In other words, most of the identified risk dependencies existed in the risks of Product Engineering and Supporting Environments. Given the nature of the projects, there were many interfaces between systems across projects, and many coordinative activities between projects were needed during development and testing. Hence, more related risks and associated risk dependencies were identified in the planning and system design stage.
- Among all 27 risk dependencies identified, there were 14 of them (51.9%) with a Risk Dependency Value of +1, 8 of them (29.6%) with a Risk Dependency Value of +2, and 5 of them (18.5%) with a Risk Dependency Value of +3. For those direct successors with the highest risk dependency value, their initial risk severity levels were all classified as “Low”, but were re-evaluated and re-classified to “Medium High”. It means that their

response priorities and strategies were revised significantly, and the response actions were changed from passive monitoring to seeking ways to reduce the risk probability or the impact or both.

Figure 5.6 shows the number of risks identified at different project stages and grouped by different risk scores for the entire program. It also presents the risk distribution after the posterior risk assessment. We noted that most of the critical risks (risk score = 4 or 5) occurred in the first two project stages, T1 and T2. The situation was understandable as the activities of these stages mainly focused on program and project planning in which risk identification was one of the major activities.

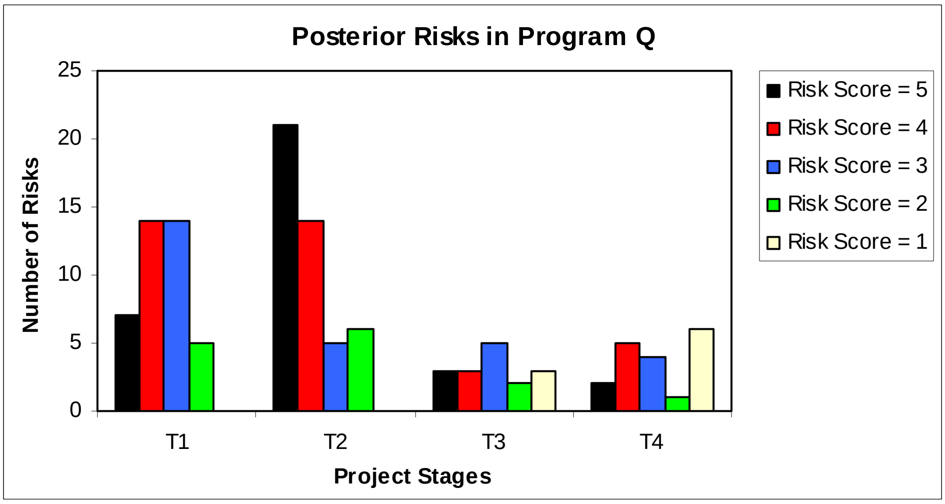


Figure 5.6: Distribution of Posterior Risks with Different Risk Scores

5.4 Risk and Risk Dependency Trends

The status of each risk was also monitored throughout the entire program. As risks might be added or removed over time (for example, if a risk's probability or its impact has become zero, there is no need to monitor it any more), the total number of active risks (i.e. the risk with *Probability* $\neq 0$ and *Impact* $\neq 0$) that were being monitored at each project stage would change. Table 5.5 summarizes the number of active risks at different project stages.

Table 5.5: Number of Risks Monitored at Different Project Stages

	T1	**T2**	**T3**	**T4**
Project A	15	29	35	39
Project B	11	24	24	30
Project C	8	27	27	28
Project Level (*PJR*)	34	80	86	97
Program Level (*PGR*)	6	6	6	6
Program Q (*PJR+PGR*)	**40**	**86**	**92**	**103**

5.4.1 Risk Trends

For each risk being monitored, its risk level might change at different project stages. It was mainly due to the risk response actions being taken for the risks. Tables 5.6 and 5.7 present the various posterior risk scores (Top 10, Top 20 and Overall) and averaged posterior risk scores (Top 10, Top 20 and Overall) calculated at different project stages.

Table 5.6: Posterior Risk Scores at Different Project Stages

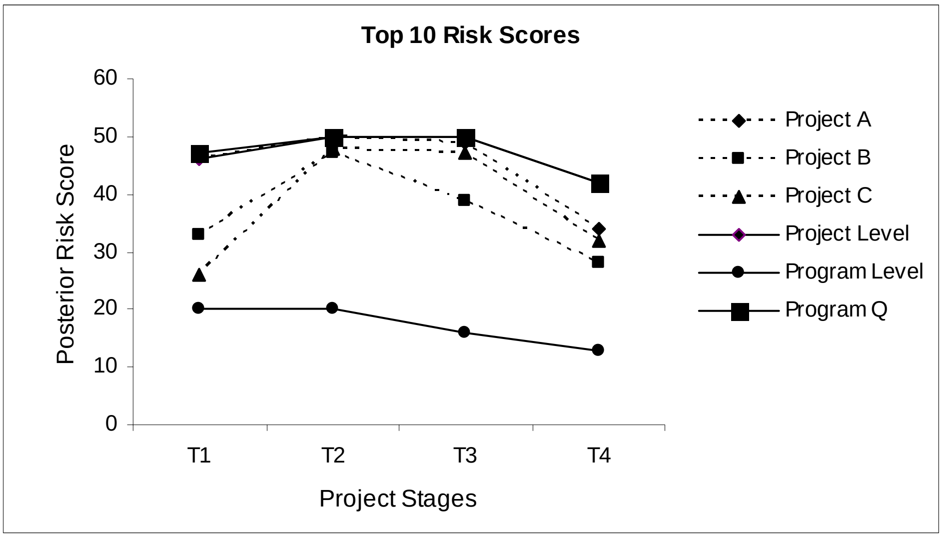
		T1	**T2**	**T3**	**T4**
Project A	RS_{10}^{+}	46	50	49	34
	RS_{20}^{+}	62	92	88	54
	RS^{+}	62	119	121	78
Project B	RS_{10}^{+}	33	47	39	28
	RS_{20}^{+}	35	80	63	46
	RS^{+}	35	88	67	56
Project C	RS_{10}^{+}	26	48	47	32
	RS_{20}^{+}	26	86	79	47
	RS^{+}	26	104	90	55
Project Level	$Q\_PJR\_RS_{10}^{+}$	46	50	50	42
	$Q\_PJR\_RS_{20}^{+}$	85	100	98	72
	$Q\_PJR\_RS^{+}$	123	311	278	189
Program Level	$Q\_PGR\_RS_{10}^{+}$	20	20	16	13
	$Q\_PGR\_RS_{20}^{+}$	20	20	16	13
	$Q\_PGR\_RS^{+}$	20	20	16	13

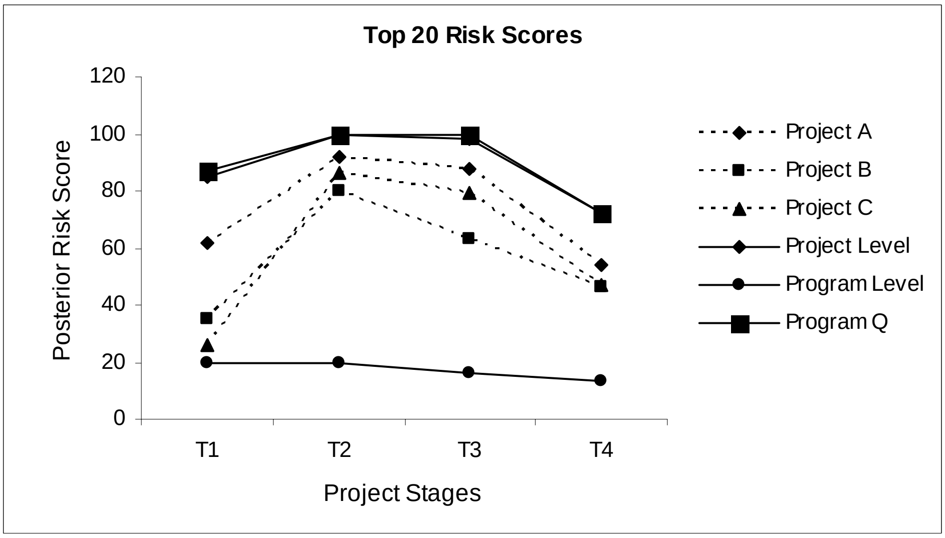
Program Q	$Q\_RS_{10}^{+}$	47	50	50	42
	$Q\_RS_{20}^{+}$	87	100	100	72
	Q_RS^{+}	143	331	294	202

Table 5.7: Averaged Posterior Risk Scores at Different Project Stages

		T1	**T2**	**T3**	**T4**
Project A	ARS_{10}^{+}	4.60	5.00	4.90	3.40
	ARS_{20}^{+}	4.13	4.60	4.40	2.70
	ARS^{+}	4.13	4.10	3.46	2.00
Project B	ARS_{10}^{+}	3.30	4.70	3.90	2.80
	ARS_{20}^{+}	3.18	4.00	3.15	2.30
	ARS^{+}	3.18	3.67	2.79	1.87
Project C	ARS_{10}^{+}	2.60	4.80	4.70	3.20
	ARS_{20}^{+}	3.25	4.30	3.95	2.35
	ARS^{+}	3.25	3.85	3.33	1.96
Project Level	$Q\_PJR\_ARS_{10}^{+}$	4.60	5.00	5.00	4.20
	$Q\_PJR\_ARS_{20}^{+}$	4.25	5.00	4.90	3.60
	$Q\_PJR\_ARS^{+}$	3.62	3.89	3.23	1.95
Program Level	$Q\_PGR\_ARS_{10}^{+}$	3.33	3.33	2.67	2.17
	$Q\_PGR\_ARS_{20}^{+}$	3.33	3.33	2.67	2.17
	$Q\_PGR\_ARS^{+}$	3.33	3.33	2.67	2.17
Program Q	$Q\_ARS_{10}^{+}$	4.70	5.00	5.00	4.20
	$Q\_ARS_{20}^{+}$	4.35	5.00	5.00	3.60
	Q_ARS^{+}	3.58	3.85	3.20	1.96

In Figure 5.7, the posterior risk scores for the Top 10 Risks, Top 20 Risks and Total Risks for projects A, B and C, project level, program level and the entire program Q, are plotted against the project stages.





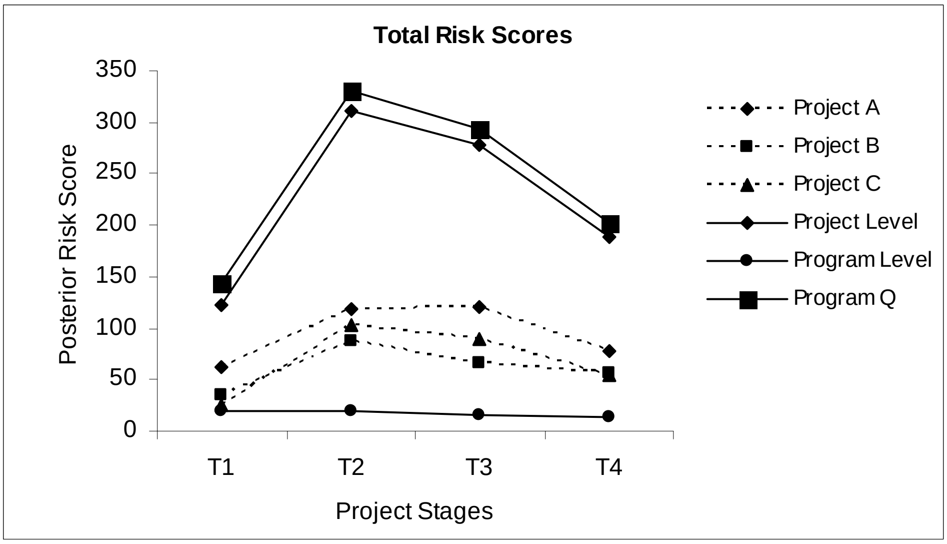


Figure 5.7: Posterior Risk Scores vs Project Stages

There are a number of observations regarding to the posterior risk scores:

- Posterior program risk scores reflected the risk level of the entire program, which included all the project level and program level risks. The posterior program risk scores were always higher than individual posterior project risk scores of Top 10 Risks, Top 20 Risks and Total Risks; it was understandable as more risks were counted in the posterior program risk scores. For example, the differences between the posterior program risk score and the individual posterior project risk scores in the Top 10 Risk Scores were very small at T2, but the differences became much larger in the Total Risk Scores as the number of risks jumped significantly from T1 to T2. In addition, as most of the program risks were project level risks (there were totally 120 identified program risks, in which 6 were program level risks and 114 were project level risks), all posterior project level risk scores were very closed to the posterior program risk scores.
- As the total numbers of risks in the initial project stage of the program and the three projects were all less than 20, the posterior risk scores for Top 10 Risks, Top 20 Risks and Total Risks for the program and projects were almost the same at T1. Besides, as the total number of program level risks was less than 10 at each project stage, the posterior program level risk scores for Top 10 Risks, Top 20 Risks and Total Risks were all the same (see the bottom curves of the three charts of Figure 5.7).
- The posterior risk score of the Top 20 Risks of each project was quite closed to that of Total Risks of each project, and the trend of the top 20 risks of each project was also quite similar to the trend of total risks of each project. These observations were particularly true for Project B and

Project C, as there were only slightly more than 20 risks to be monitored at each project stage. The observations agreed with the suggestion of Ferguson (2004) that monitoring of the top 20 project level risks could be sufficient in project management; however, it may not be suitable for managing program. We observed that the trends of the Top 10 risks and Top 20 Risks were quite different from the trend of Total Risks for the program. At T2 and T3, the posterior program level risk scores for Top 10 risks and Top 20 Risks were the same, but the score for Total Risks had dropped significantly; this situation reflected that there were more than 20 high severity risks in the program although some risks had been mitigated.

- Among all project stages of the three projects, the risk scores in the project initiation stages T1 were relatively low. At T1, the risks were only identified by project sponsors and system owners, who did not have much technical knowledge about the system development risks, and mainly concentrated on the risks related to project constraints. It could be observed that the scores increased rapidly at T2, the project planning and system design stage, where risks were identified by technical staff. T2 had the highest risk scores mainly because it had the highest number of risks identified among all project stages, and risk response actions had not been applied to those risks identified earlier in T1. This situation also partially reflected from the curve of Total Risks for program, as most of the program risks were project level risks.
- Other than the posterior risk scores of Project A, all scores had been continuously decreasing after project stage T2 due to the positive effect of risk response actions. The variation of Project A was mainly due to the

effects of a couple of new risk dependencies identified at project stages T2 and T3, in which many system related dependencies were found. The curve of Total Risks for program reflected that risk response actions had been effective in reducing the risk, thus, lowering the posterior risk scores.

- Although the averaged posterior risk scores can help to observe the program and project risk trends, they are not useful without further analysis, as the averaged scores can potentially obscure the wide variances among risk. For example, a low averaged score does not mean that there is no high severity risk.

Other than using posterior risk scores, the posterior risk indexes can also be used to monitor the trend of program risks. As the organization adopted the Ranking Method to evaluate risks, we obtained the various posterior risk indexes at different project stages (as shown in Table 5.8) by dividing the corresponding value in Table 5.7 by *5*, the maximum possible value of each risk.

Table 5.8: Posterior Risk Indexes at Different Project Stages

		T1	**T2**	**T3**	**T4**
Project A	RI_{10}^{+}	0.92	1.00	0.98	0.68
	RI_{20}^{+}	0.83	0.92	0.88	0.54
	RI^{+}	0.83	0.82	0.69	0.40
Project B	RI_{10}^{+}	0.66	0.94	0.78	0.56
	RI_{20}^{+}	0.64	0.8	0.63	0.46
	RI^{+}	0.64	0.73	0.56	0.37
Project C	RI_{10}^{+}	0.52	0.96	0.94	0.64
	RI_{20}^{+}	0.65	0.86	0.79	0.47
	RI^{+}	0.65	0.77	0.67	0.39
Project Level	$Q\_PJR\_RI_{10}^{+}$	0.92	1.00	1.00	0.84
	$Q\_PJR\_RI_{20}^{+}$	0.85	1.00	0.98	0.72
	$Q\_PJR\_RI^{+}$	0.72	0.78	0.65	0.39
Program Level	$Q\_PGR\_RI_{10}^{+}$	0.67	0.67	0.53	0.43
	$Q\_PGR\_RI_{20}^{+}$	0.67	0.67	0.53	0.43
	$Q\_PGR\_RI^{+}$	0.67	0.67	0.53	0.43

Program Q	$Q\_RI_{10}^{+}$	0.94	1.00	1.00	0.84
	$Q\_RI_{20}^{+}$	0.87	1.00	1.00	0.72
	Q_RI^{+}	0.71	0.77	0.64	0.39

Figure 5.8 presents the posterior risk indexes of all posterior risks at each project stage. From the figure, it can be noticed that most of the indexes (except the indexes for Project A risks and program level risks) were going up from T1 to T2; it reflected that many higher severity risks were identified in T2. Afterwards, as many risk response actions had been taken, the probabilities and/or impacts of posterior risks were reduced; therefore, all indexes were going down, and they reached the lowest values at the final testing stage T4. At T4, the posterior risk indexes were all below 0.5.

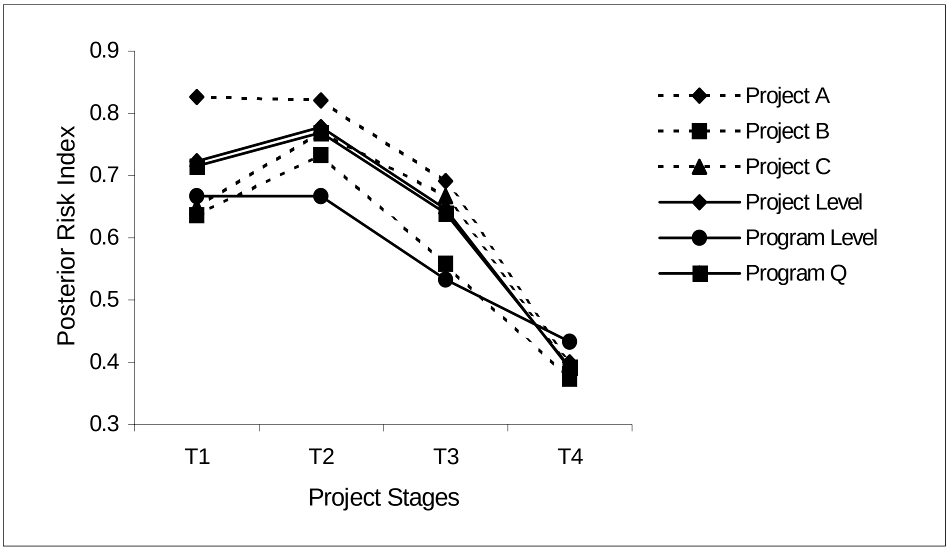


Figure 5.8: Posterior Risk Indexes

We also noticed that the posterior program level risk index had the highest value among other indexes at T4. As many program level risks were associated with external factors, these risks could not be reduced significantly even though their response actions had been taken. A contingency plan was developed to prepare for the occurrences of those risks.

In addition, with the aid of the posterior program risk index, we noticed that Project A was the only project with an index that was higher than the index of entire program throughout all project stages. This project should have been monitored closely.

5.4.2 Risk Dependency Trends

The risk dependency index can help to measure the degree of risk dependency of a project. A higher index value means that more dependencies are identified among risks and indicates that assessing dependencies and selecting appropriate response strategies are needed. All the Total Risk Dependency Counts and the associated Risk Dependency Indexes at different project stages are summarized in Table 5.9, and are plotted in Figure 5.9. As our study involved three projects and there were risk dependencies across projects, the total number of direct successors (*NDS*) and the total number of direct predecessors (*NDP*) would not be the same for each project. In order to obtain the number of risks being affected by dependencies in computing *RDI*s, the *NDP*s were counted and used to obtain the *TRDC*s.

Table 5.9: Total Risk Dependency Counts and Risk Dependency Indexes

		T1	**T2**	**T3**	**T4**
Project A	*TRDC(Q, A)*	1	3	5	5
	RDI(Q, A)	0.005	0.004	0.004	0.003
Project B	*TRDC(Q, B)*	2	8	10	8
	RDI(Q, B)	0.018	0.014	0.018	0.009
Project C	*TRDC(Q, C)*	1	6	7	5
	RDI(Q, C)	0.018	0.009	0.010	0.007
Project Level	*TRDC*	4	17	22	18
	RDI	0.004	0.003	0.003	0.002
Program Level	*TRDC*	0	0	0	0
	RDI	0	0	0	0
Program Q	*TRDC*	4	17	22	18
	RDI	0.003	0.002	0.003	0.002

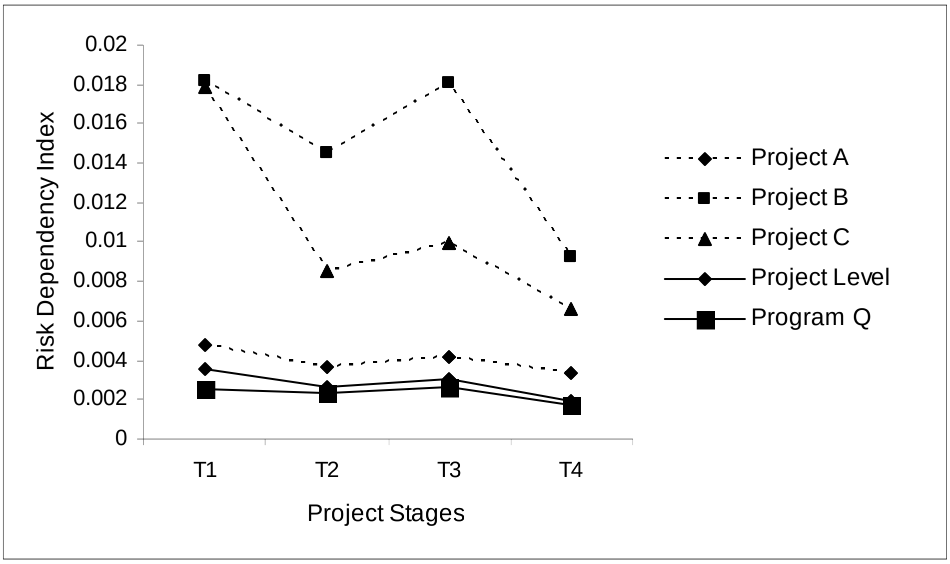


Figure 5.9: Risk Dependency Indexes

Some observations are noticed regarding the trends of risk dependencies:

- By definition, the highest possible value of risk dependency index is 1. From Figure 5.9, the overall values of all indexes were relatively low, and it meant that many risks identified were not associated with any risk dependencies. However, as discussed in Section 5.2, more dependencies may have been identified if the identification approach can be improved.
- Although the risk dependency indexes of all three projects were moving up from project stage T2 to T3, the project risk indexes (Figure 5.8) were still going down at T3. It was because a large number of risks identified in the earlier stages were removed or their severity levels were reduced due to the effect of the risk response actions.
- The risks in Project B had been affected the most by the risk dependency effect, while those in Project A had been least affected. According to the constraints of the projects, the system output of Project A was the input to the system of Project B, and the system output of Project B was the input

to the system of Project C. It was quite reasonable that the system of Project B had been affected the most with the highest number of interfaces between the systems of both Project A and Project C. On the other hand, the system of Project A was affected the least due to the lowest number of interfaces with other systems.

- All three projects had the lowest index values at T4, the system testing stage of the projects. It was because most of the planned risk response actions had been taken and the risk severity levels were reduced. Therefore, the dependency indexes were relatively low.

5.5 Discussion and Limitations

From the study of the program with three IT projects, risk dependencies did exist and could be identified. Applying the risk dependency concept and the enhanced risk management practices, some tangible benefits were recognized.

- With the additional information of risk dependency relationship between risks, each dependent risk could have a chance to be re-evaluated, which could directly affect its severity level, response priority and finally the associated response actions.
- Instead of evaluating project risks individually, risk dependencies are explicitly considered while applying the posterior risks to determine the response strategies and calculate the program and project level risk scores. The risk scores can better reflect the overall program risk at a given time because:

    - at the project level, we consider not only risk dependencies within each project but also those dependencies across projects;
    - at the program level, a consolidated view of all risks in the program can be obtained.
- ➢ Current project risk management practices ignore the existence of risk dependencies and miss the response actions to deal with them, especially for those risks with a high posterior risk severity level due to the high degree of risk dependency effect. With the consideration of risk dependencies, more risk response strategies can be considered and developed.
- ➢ Traditionally, the project teams only focused on those risks that they identified themselves and those that they were responsible; thus, any risk response actions could only be planned and monitored by themselves. To deal with risk dependency issues, the communications between project teams under the same program were improved, and projects gained the following benefits:
    - some risk response actions could be shared and applied to more than one risk;
    - some related risks could be managed together;
    - more risks could be identified while learning from the risks of other inter-dependent projects.

However, during the risk dependency identification and monitoring exercises, a number of observations and limitations were noticed, which could provide some insight into studying of the risk dependency problems and improving the project and program risk management practices:

- **Managing Opportunities**

The case study mainly focused on risks and risk dependencies and ignored opportunities. Although the same set of practices and measures can be applied to opportunities, it is still common in IT projects that opportunities are not adequately managed.

- **Risk Dependencies Identification**

From the case study, we noted that no dependencies could be established between program level risks and project level risks. Further research is needed to investigate whether this is true for general cases. Moreover, we noted that the identified risk dependencies had the following common characteristics:

- they were usually identified at the same project stages;
- the associated risks came from the same risk categories;
- they were only established between the risks identified by different persons.

This implies that the identification process may not be effective, and we need to develop more systematic methods of identifying risk dependencies. As the identified dependencies in the study came from the risks identified by different persons, and there may not always be a QA role for every project, a designated person is still needed to coordinate all the activities. In addition, it would be worthwhile to build an organizational risk repository archive to capture past experiences for future projects.

- **Risk Dependency Responses**

We have proposed two sets of risk response strategies to deal with risk dependencies: the first set is based on enhancing the current risk management practices by applying posterior risks, and the second is a new set of strategies focusing on risk dependencies. In the case study, we noted that both sets of strategies had also been applied, although the first set was used the most. In fact, the risk dependency index measures the degree of risk dependency at different project stages. When the index value is high, it means that many dependencies exist among risks, and extra effort should be devoted to analyze the dependencies and select appropriate response strategies.

As the case study involved three critical projects, and prudent risk responses were taken, no identified risk really occurred. We could not confirm whether risks would occur if we did not consider the dependency issues; in other words, we do not know whether we had overestimated the risks. Nevertheless, the risk responses taken were still justifiable as the projects were critical to the organization and adequate recourses were provided.

- **Risk Monitoring**

Both risk score and risk index are useful to analyze the trend of project and program risk levels and help to evaluate the effectiveness and performance of risk response actions. Not knowing the maximum possible value of a risk, we can only apply the risk score. If the maximum possible value of a risk can be determined, like using the ranking method with assigned risk values, the risk index can be calculated to facilitate the comparison of risk trends of different projects.

According to the risk definition, if a risk does occur, it is not a risk anymore, but becomes an issue. As it would be taken out of any risk related calculations, other measures are desired to account for the effect of an occurred risk on the program or project objectives.

Chapter 6 Conclusion and Future Work

We have modeled the risk dependency and proposed a new risk management method that can systematically manage risk dependencies with the concept of *Posterior Risk*. We have also identified four types of risk dependencies in a program.

In this thesis, we first formally defined risk dependency and proposed methods to re-estimate risk by taking account of risk dependency effects. The risk dependency model is then integrated into a set of enhanced management practices which largely follow the basic steps of SEI's risk paradigm. As the risk dependency effects can either be favorable or non-favorable, we further proposed another set of new practices to evaluate, react, monitor and control the risk dependencies by assessing the favorability and the degree of risk dependency effect. With these new practices, additional risk response strategies that focus on risk dependencies are proposed.

In addition, the risk metrics are usually developed for measuring independent project risks but cannot be directly applied in program environments. In this thesis, based upon the current metrics, we have enhanced those metrics to measure risks from the program perspective with due considerations of the risk dependencies.

We have conducted case studies by applying the enhanced practices to a program with three real life IT projects, and confirmed that dependencies between risks do exist, especially if the risks were identified by different groups of stakeholders. The enhanced and new risk management practices for evaluating, prioritizing, and responding to risk and risk dependencies, as well as the designated metrics, also showed valuable and supportive results. Although program management provides a higher level structure that helps to improve the communications between projects at management level, we observed that, as project teams needed to deal with risk dependency issues, communications between projects were also improved at working levels. In conclusion, the new methodology is practical in managing project risks.

Due to the nature of case studies, we did not perform any benchmarking analysis and compare the project results with and without the consideration of risk dependencies. In addition, the proposed methodology still has a number of areas that need further verification and improvement:

- The case studies only focused on studying risks. More work is needed to confirm whether the same concept can be applied in managing opportunities.
- From the case studies, we noted that no dependencies could be established between program level risks and project level risks. Further research is needed to investigate whether this is true for general cases. Moreover, there is a lack of efficient methods to identify risk dependencies.
- We have proposed two different sets of risk response strategies: one focuses on posterior risks and another one focuses on risk dependencies. As both strategies consider the existence of risk dependencies, further work is needed to study how these two sets of strategies can be integrated together.
- In defining the risk metrics, we have assumed that program and project objectives are all independent. In fact, an objective may depend on another, and each objective can also be divided into sub-objectives. We need to enhance the proposed metrics for measuring program and project risks by their objectives.

Another fruitful area of research is to apply the risk dependency concept to program risk management. The current practices of program risk management ignore the existence of risk dependencies. The practices proposed by us could be enhanced to apply at the program level. Some of the directions for further study on program management may include:

- how to identify risk dependencies within a program;
- how to evaluate the identified dependencies and determine their impact on the concerned program risks;
- how to re-evaluate the concerned program risks and estimate their posterior risks;
- how to develop strategies to deal with risk dependencies between projects.

In developing the risk dependency model, the possibilities of progressions other than arithmetic are not considered, but they may offer better solutions in the long run. In the future research, the risk management framework may also be extended to include the use of Quotients as additional analysis.

Bibliography

APM (2004), *The APM Project Risk Analysis & Management Guide (PRAM)*, 2nd ed., Association for Project Management, APM Publishing

ASC (2003), *Risk Management Process & Implementation, American Systems Corporation*, Chantilly, VA

Asnar, Y. and Giorgini, P. (2006), “Modelling Risk and Identifying Countermeasure in Organizations”, *Proceeding of 1st International Workshop on Critical Information Infrastructures Security*, Samos Island, Greece

Asnar, Y. and Giorgini, P. (2007), “Risk Analysis as part of the Requirements Engineering Process”, *Technical Report DIT-07-014*, Department of Information and Communication Technology, University of Trento

Aven T. (1992), *Reliability and Risk Analysis*, 1st ed., London: Elsevier Applied Science

Boehm, B. (1989), *Software Risk Management*, Tutorial, IEEE Computer Society Press

Boehm, B. (1991), “Software Risk Management: Principles and Practices”, *IEEE Software*, vol. 8, issue 1, pp.32-41, Jan

Bouti, A. and Kadi, A. (1994), “A state-of-the-art review of FMEA/FMECA”, *International Journal of Reliability, Quality and Safety Engineering*, vol. 1, pp.515-543

Brown, J. (2008), *The Handbook of Program Management: How to Facilitate Project Success with Optimal Program Management*, McGraw-Hill, NY, pp.177-178

Carr, M., Konda, S., Monarch, I., Ulrich, F. and Walker, C., (1993), *Taxonomy-based risk identification*, Pittsburgh, PA: Software Engineering Institute

Charette, R. (1989), *Software Engineering Risk Analysis and Management*, McGraw-Hill, New York

Chittister, C. and Haimes, Y. (1993), “Risk associated with software development: a holistic framework for assessment and management”, *IEEE Transactions on Systems, Man and Cybernetics*, 23(3), pp.710-723

Conrow, E. and Shishido, P. (1997), “Implementing risk management on software intensive projects”, *IEEE Software*, 14(3), pp.83-89

Cormen, T., Leiserson, C., Rivest, R. and Stein, C. (2001), *Introduction to Algorithms*, 2nd ed., MIT Press and McGraw-Hill, pp.531–539

COSO (2004), *Enterprise Risk Management: Integrated Framework*, Committee of Sponsoring Organisations of the Treadway Commission (COSO)

Dorofee, A., Walker, J., Alberts, C., Higuera, R., Murphy, R., and Williams, R. (1996), *Continuous Risk Management Guidebook*, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA

Fenton, N. and Neil, M. (2004), *Combining evidence in Risk Analysis using Bayesian Networks*, Agena White Paper, W0704/01, v01.01

Ferguson, R. (2004), "A Project Risk Metric", *CrossTalk, The Journal of Defense Software Engineering*, vol.17, no.4, pp.12-15, Apr

Garvey, P. (2000), *Probability Methods for Cost Uncertainty Analysis - A Systems Engineering Perspective*, Marcel Dekker, Inc., New York

Giorgini, P., Mylopoulos, J., Nicchiarelli, E. and Sebastiani, R. (2003), "Formal Reasoning Techniques for Goal Models", *Journal of Data Semantics*, Springer, vol.1, pp.1-20

Grant, K. and Pennypacker, J. (2006), "Project Management Maturity: An Assessment of Project Management Capabilities Among and Between Selected Industries", *IEEE Transactions on Engineering Management*, vol.53, no.1, pp.59-68

Higuera, R. and Haimes, Y. (1996), *Software Risk Management*, Pittsburgh, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA

Hillson, D. (2001), "Effective Strategies for Exploiting Opportunities", *Proceedings of the Project Management Institute Annual Seminars & Symposium*, Nashville, Tenn., USA, Nov 5-7

Hillson, D. (2008), "Towards Program Risk Management", *PMI Global Congress Proceedings*, Denver, Colo.

Houmb, S., Georg, G., France, R. and Matheson, D. (2004), "Using aspects to manage security risks in risk-driven development", *3rd International Workshop on Critical Systems Development with UML*, Oct, number TUM-I0415, pp.71-84

Hyatt, L. and Rosenberg, L. (1996), *A Software Quality Model and Metrics for Identifying Project Risks and Assessing Software Quality*, 8th Annual

Software Technology Conference

IEC60300-3-9 (1995), *Dependability management - Part 3: Application guide - Section 9: Risk analysis of technological systems - Event Tree Analysis (ETA)*, 1st ed., International Electrotechnical Commission

IEC61025 (2006), *Fault Tree Analysis (FTA)*, 2nd ed., International Electrotechnical Commission

IEC61165 (2006), *Application of Markov techniques*, 2nd ed., International Electrotechnical Commission

ISO20000-2 (2005), *Information technology - Service management - Part 2: Code of practice*, International Organization for Standardization

ITGI (2007), *IT Assurance Guide Using COBIT*, The IT Governance Institute

Johnson, W. (1973), *MORT - The Management Oversight and Risk Tree*, SAN 821-2, US Atomic Energy Commission

Kähkönen, K. (2001), “Integration of Risk and Opportunity Thinking in Projects”, *Proceedings of the 4th European Project Management Conference (PMI Europe 2001)*, London, UK, Jun 6-7

Kerzner, H. (2004), *Advanced Project Management: Best Practices on Implementation*, New York, NY: John Wiley & Sons

Kwan, T. and Leung, H. (2005), “An Enhanced Risk Taxonomy for Information Technology Projects”, *Proceeding of IASTED International Conference on Software Engineering and Applications*, Phoenix, AZ, USA, Nov 14-16, pp.285-291

Kwan, T. and Leung, H. (2007), “Improving Risk Management Practices for IT Projects”, *Proceeding of the Third IASTED International Conference on Advances in Computer Science and Technology*, Phuket, Thailand, Apr 2-4, pp.443-448

Letavec, C. (2006), *The Program Management Office: Establishing, Managing and Growing the Value of a PMO*, FL:J. Ross Publishing, pp.2-3

Lientz, B. and Rea, K. (2001), *Breakthrough Technology Project Management*, San Diego, CA: Academic Press

Martinelli, R. and Waddell, J. (2004), *Program and Project Management: Understanding the Differences*, PM Forum, Oct

Martinelli, R. and Waddell, J. (2007), “Program Management: it’s about the business”, *PM World Today*, vol. IX, issue I, Jan

Milosevic, D., Martinelli, R. and Waddell, J. (2007), *Program Management for Improved Business Results*, Hoboken, NJ: John Wiley & Sons

Navarro, E., Letelier, P., Reolid, D. and Ramos, I. (2007), “Configurable Satisfiability Propagation for Goal Models Using Dynamic Compilation Techniques”, *Advances in information systems development: new methods and practice for the networked society*, Springer, pp.167-179

Nielsen, D. (1971), *The Cause/Consequence Diagram Method as a Basis for Quantitative Accident Analysis*, Danish Atomic Energy Commission, RISO-M-1374

Lientz, B. and Rea, K. (2001), *Breakthrough Technology Project Management,* San Diego, CA: Academic Press

Lock, D. (2007), *The Essentials of Project Management*, 3rd ed., Hampshire:Gower Publishing

Longstaff, T., Chittister, C., Pethia, R. and Haimes, Y. (2000), “Are we forgetting the risks of information technology”, *Computer*, 33(12), pp.43-51

Murthi, S. (2002), “Preventive risk management for software projects, IT Professional”, 4(5), pp.9-15

Noor, I. (2001)*, Risk and issue management - principles and practice*, PMOLink

Pang, P. (2004), *Essentials of Manufacturing Engineering Management*, 1st ed., iUniverse, Inc., p.301

Parker, R. (2008), “ISACA Releases Results of Top Business/Technology Issues Survey”, *COBIT Focus*, vol.4, Oct

Pearl, J. (1988), *Probabilistic Reasoning in Intelligent Systems: networks of plausible inference*, San Mateo, CA: Morgan Kaufmann

PMCC (2002), *A Guidebook of Project & Program Management for Enterprise Innovation*, rev 1, Project Management Professionals Certification Center, Japan, Aug

PMI (2007), *Program Management Professional Examination Specification*, Project Management Institute, Newtown, PA

PMI (2008a), *A Guide to the Project Management Body of Knowledge (PMBOK Guide)*, 4th ed., Project Management Institute, Newtown, PA

PMI (2008b), *The Standard for Program Management*, 2nd ed., Project Management

Institute, Newtown, PA

Pressman, R. (2001), *Software Engineering: A Practitioner's Approach*, 5th ed., McGraw-Hill

Reiss, G. (1996), *Programme Management Demystified: Managing multiple projects successfully*, E&FN Spon, UK

Ropponen, J. and Lyytinen, K. (2000), "Components of software development risk: how to address them? A project manager survey", *IEEE Transactions on Software Engineering*, 26(2), pp.98-112

Sage, A. (1995), *Systems management for information technology and software engineering*, New York, NY: John Wiley & Sons

SEI (2006), *CMMI for Development*, ver.1.2, Carnegie Mellon Software Engineering Institute, Pittsburgh, PA

Sherer, S. (1995), "The three dimensions of software risk: technical, organizational, and environmental", *Proceedings of 28th Hawaii International Conference on System Sciences*, pp.369-378

Sherer, S. (2004), "Managing risk beyond the control of IS managers: the role of business management", *Proceedings of 37th Hawaii International Conference on System and Sciences*, Maui, Hawaii, Jan 5-8, track 8, vol.8

Standish (2009), *Chaos Summary 2009 – The 10 Laws of Chaos*, The Standish Group International, Inc., Boston, MA

Sutton, I. (1992), *Process Reliability and Risk Management*, 1st ed., Van Nostrand Reinhold

Van Scoy, R. (1992), *Software Development Risk: Opportunity, Not Problem*, CMU/SEI-92-TR-30, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA

White, B. (2006), "Enterprise Opportunity and Risk", *INCOSE 2006 Symposium Proceedings*, Orlando, FL, Jul 9-13

Williams, R., Walker J. and Dorofee A. (1997), "Putting Risk Management into Practice", *IEEE Software*, vol.14, no.3, May/June, pp.75-82

Zacharias, O., Panopoulos, D. and Askounis, D. Th. (2008), "Large Scale Program Risk Analysis Using a Risk Breakdown Structure", *European Journal of Economics, Finance and Administrative Sciences*, issue 12, pp.170-181