DECISION SCIENCES
A JOURNAL OF THE DECISION SCIENCE INSTITUTE

Decision Sciences
Volume 43 Number 5
October 2012

Reducing Requirement Incorrectness and Coping with Its Negative Impact in Information System Development Projects

Jack Shih-Chieh Hsu,† Tung-Chin Lin, and Kuang-Ting Cheng
Information Management Department, National Sun Yat-sen University, Kaohsiung, Taiwan, ROC, e-mail: jackshsu@mis.nsysu.edu.tw, tclin@mis.nsysu.edu.tw, nsysuichiro@gmail.com

Lars P. Linden
5704 Dahlia Avenue, New Port Richey, FL 34652, e-mail: lars@mudia.com

ABSTRACT

The negative impact of incorrect requirements on information system development (ISD) project performance has long been acknowledged. This study addresses the problem of incorrect requirements by proposing a model that combines the error reduction and coping concepts proposed by Field, Ritzman, Safizadeh, and Downing (2006) with the view that ISD is a knowledge-intensive process. The model hypothesizes that when developers and users possess an understanding of each other's primary domain of knowledge, the prevention of incorrect requirements and the mitigation of the negative consequences of incorrect requirements tend to improve project performance. Data collected from 250 ISD professionals on the basis of their experiences of recently completed ISD projects confirmed all of our hypotheses. The results demonstrate that the eliciting of incorrect requirements can be reduced when users and developers possess cross-domain understanding and when requirement analysis methodologies and techniques are available. Furthermore, the negative impact of incorrect requirements on project performance can be mitigated when developers have sufficient ISD knowledge and behavioral knowledge.
[Submitted: September 16, 2010. Revised: April 15, 2011; August 27, 2011; December 28, 2011. Accepted: January 10, 2012.]

Subject Areas: Incorrect Requirements, Project Performance, Reduction and Coping, and Software Project.

INTRODUCTION

Requirements engineering occupies much of the time spent during the early stages of an information system development (ISD) project. Yet the resulting requirements are often the greatest cause of project failure (Hofmann & Lehner, 2001). Requirements are problematic for several reasons: developers elicit the wrong requirements, developers fail to validate requirements correctly, environmental change increases uncertainty, or developers mis-manage a system's requirements (Boehm, 1991; Schmidt, Lyytinen, Keil, & Cule, 2001; Wallace & Keil, 2004a).

†Corresponding author.

In this article, we focus on the requirements problem of *incorrect requirements* because it hinders the planning and managing of a project (Gemino, Reich, & Sauer, 2007) and increases the residual risks that occur during the later stages of an ISD project (Nidumolu, 1996). Incorrect requirements are a risk factor because they threaten the success of an ISD project and generate negative impacts on the final performance (Wallace & Keil, 2004a). When requirements are incorrect, the beneficial outcomes tend to decrease (Cooper & Swanson, 1979; Davis, 1982; Han & Huang, 2007). A better understanding of incorrect requirements is important because it may enable managers to decrease their ill impacts.

This problem has not been ignored. Mechanisms proposed by information systems researchers to address the problem of incorrect requirements fall into two categories. The first category includes techniques, managerial intervention practices, and methodologies that aim to increase the quality of requirements elicited and seek to prevent incorrect requirements from occurring (e.g., Byrd, Cossick, & Zmud, 1992; Nidumolu, 1996; Faraj & Sproull, 2000; Hickey & Davis, 2004). The second category includes mechanisms that emphasize the importance of mitigating the negative impact that poor requirements have on final performance (e.g., Fairley, 1994; Barki, Rivard, & Talbot, 2001; Hsu, Chan, Liu, & Chen, 2008). In this article, we attempt to integrate the general concepts implied by these two categories. We explore whether the impacts of requirements incorrectness may be diminished by improving the number of correct requirements and by mitigating the negative impact of existing incorrect requirements.

While considering this dual approach to the problem of incorrect requirements, we go one step further and adopt the perspective that ISD is knowledge-intensive. We consider that developers, as they work toward supporting the users' business domain, and users, as they describe their needs to developers, require knowledge from many distinct subjects. Exploring the importance of knowledge in ISD is not new. However, little is known about how knowledge might impact the problem of incorrect requirements. Might the enhancement of a particular type of knowledge, or some combination of knowledge from separate domains, help to reduce the number of incorrect requirements or help to mitigate the negative impacts of any unavoidable incorrect requirements?

We draw on theoretical and empirical reports that argue that requirements incorrectness is a major problem in ISD projects, and that ISD is knowledge intensive; this leads us to a two-fold theoretical approach to incorrect requirements. First, the reduction and coping strategies proposed by Field et al. (2006) are adopted to investigate how incorrect requirements can be reduced and how their negative impacts on project performance can be mitigated. Second, the knowledge-intensive perspective is influenced by a variety of ISD studies relating various types of knowledge to development. Grounded upon these theories, the proposed model incorporates the approaches of reduction and mitigation by focusing on the knowledge that influences reduction and coping. This article contributes to the related literature by conceptualizing the dual approach of reduction and coping in terms of the knowledge possessed by developers and users.

The remainder of this article is organized in the following way. In the next section, we review the requirement eliciting literature and negative impact mitigation literature and develop research hypotheses. In the third section, we provide details of the survey process. In the final section, we report upon our hypotheses testing, discuss implications, present conclusions, and outline the study's limitations.

LITERATURE REVIEW AND HYPOTHESES DEVELOPMENT

In an effort to provide insight into the high failure rate of software projects, several factors have been identified which impede the efficiency of people working on ISD projects. The presence of these problem-causing factors jeopardizes the successful performance of a project, and, by adopting a managerial perspective of risk, the factors can be viewed as constituting risk (Wallace, Keil, & Rai, 2004b). One categorization of the dimensions of software project risk (Wallace & Keil, 2004a) lists broadly scoped categorizations of risk factors: organizational environment, user, requirements, project complexity, planning and control, and team. Studies that have identified risk factors (e.g., McFarlan, 1981; Boehm, 1991; Barki et al., 2001) usefully incorporate various groupings of these risk factors into models. However, the large scale and complexity of these studies lead us to focus on requirements as a single risk factor, and to narrow our focus to the particular problem of incorrect requirements, so that we might aim for a research model with parsimonious and controllable parameters.

Incorrect Requirements and ISD Project Performance

During an ISD project, developers typically elicit requirements, a process that produces an artifact that helps to define the information systems being developed. Not surprisingly, the requirements elicitation process and the requirements artifacts themselves have been found to be problematic. If requirements are not complete, clear, adequate, and correct, difficulties arise during the ISD process (Roman, 1985; Curtis, Krasner, & Iscoe, 1988). These difficulties then often lead to information system failures (Boehm, 1981; Lyytinen & Hirschheim, 1987; Lederer & Prasad, 1992; Jones, 1994). We label requirements that are not complete, not clear, or not adequate as *incorrect requirements*. Efforts aimed at reducing the number of incorrect requirements are found to be an important part of an ISD project (Hofmann & Lehner, 2001; Sage & Rouse, 2009).

If the requirements captured do not reflect the information system actually needed by the user, the design of the system may be found lacking, creating a situation whereby users resist using the system. Incorrect requirements and any resulting incorrect system design also increase cost and delay scheduled deadlines because the requirements may need to be redefined and the system may need to be redesigned. By tracing the input of incorrect requirements through the development process, the phenomenon of incorrect requirements is found to impair the final aggregate assessment of a project: project performance.

System design work is seen as an important leverage point at which any errors in the requirements are manifest into design problems with the system being developed. It is during the production of the system design that the systems analysts use their knowledge of how to design information systems to transform the perceived knowledge of the users into an appropriate system design. By focusing on how system design work is a critical juncture at which requirements incorrectness influences the design of the information system, we can begin to trace the impact of incorrect requirements on *project performance*—the ability to meet project goals within a predefined budget and schedule (Schwalbe, 2002). Given that empirical studies show that project difficulties resulting from various requirement issues are closely associated with project performance (Nidumolu, 1996; Wallace & Keil, 2004a; Gemino et al., 2007; Han & Huang, 2007), we reason that the narrower phenomenon of incorrect requirements also tends to diminish project performance. Therefore, we hypothesize:

> *H1*: *The level of requirement incorrectness is negatively associated with project performance.*

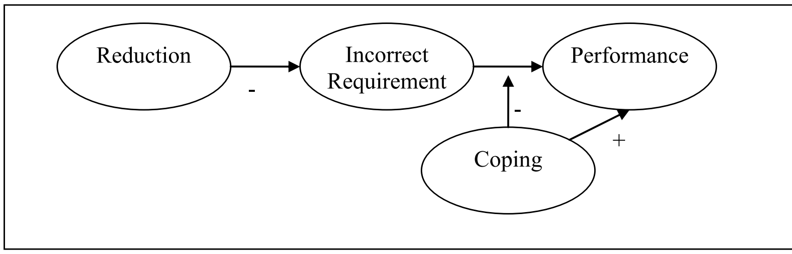
Theoretical Approach to Requirements Incorrectness

Incorrect requirements captured during the early stages of development are a major contributor to schedule delays and result in a low quality system. When the requirements are inadequate, the members of the project team encounter increased difficulties with the planning and control of the project (Nidumolu, 1996; Wallace & Keil, 2004a). But while they may be difficult to control, incorrect requirements are not uncontrollable (Keil, Cule, Lyytinen, & Schmidt, 1998). Methods said to be effective in countering poor requirements are numerous: horizontal or vertical coordination (Nidumolu, 1996; Barki et al., 2001), expertise coordination among stakeholders (Faraj & Sproull, 2000), user participation (Barki et al., 2001; Hsu et al., 2008), formal integration (Barki et al., 2001), effective change management (Fairley, 1994), and selecting the correct eliciting mechanism (Byrd et al., 1992). We seek to understand further the problem of incorrect requirements. We exercise a two-fold theoretical approach to the problem. First, as the aim is to mitigate negative impacts, we adopt a theoretical framework which categorizes approaches to the problem into reduction and coping. Second, because ISD is knowledge intensive, we aim to know, so we adopt theories that suggest the beneficial types of knowledge that can be applied to these approaches of reduction and coping.

Reduction and Coping

A service operations management study conducted by Field et al. (2006) categorized the approaches aimed at reducing the negative impacts raised from operational uncertainty into two categories, reduction and coping. The Field et al. (2006) study's conclusion is that, to improve performance, managers should attempt to reduce unwanted situations and attempt to cope with their negative effects. The objective of *reduction* is to decrease the amount of uncertainty and its negative effects by addressing the uncertainty itself. The objective of *coping* is to mitigate the negative effects of the uncertainty. Coping does not address the underlying uncertainty itself. If incorrect requirements are viewed as unwanted situations that may negatively impact performance, then the approaches of reduction and coping (Field et al., 2006) can provide a foundation for a theory applied within the context of ISD. Following the overall pattern of these two categories, we argue that members of ISD projects who reduce the level of requirement incorrectness and who

Figure 1: Conceptual model.



enhance their capacity to react to its negative impacts will, in turn, achieve higher performance. An illustration of our conceptual model is shown in Figure 1.

Adopting the Knowledge Perspective

The subject of incorrect requirements can also be viewed in terms of the different types of knowledge present in the project's participants. For example, Barki et al. (2001) point out that if developers do not possess sufficient ISD and behavioral knowledge, software projects are exposed to higher levels of threat in execution. Gemino et al. (2007) specify that a lack of knowledge resources increases the difficulty for organizations to provide support and ultimately reduces the effectiveness of project management practices. Fink and Neumann (2009) indicate that capabilities are enhanced when technical, behavioral, and business knowledge are better positioned. Several researchers (Ropponen & Lyytinen, 2000; Wallace & Keil, 2004a) argue that the impact of incorrect requirements on project outcomes can be mitigated when developers are capable of executing projects in an effective manner. Faraj and Sproull (2000) argue that because knowledge-exchange and integration activities are conducted to counter challenges faced, knowledge itself can be viewed as one of the most important resources in such a project. We hope to extend this trajectory of research as we work to understand more fully the types of knowledge possessed by ISD project participants and the ways these knowledge types tend to counter the impact of incorrect requirements.

Reduction via Increased Developers' Business Knowledge

One way to view incorrect requirements is to look at the requirements and consider whether any information needed to design the system is missing from the requirements. Information incompleteness is defined as the difference between the amount of information required and the amount of information already possessed (Galbraith, 1977). There are at least two reasons information necessary to design a quality system may not be expressed in the requirements: the system analysts may not possess the analytical knowledge necessary to process such information, or the information exchange among users and developers may be ambiguous or equivocal (e.g., Weick, 1979; Daft & Lengel, 1986).

Organizational information processing theory indicates that an organization may cope with information incompleteness by increasing its information processing capability or by reducing the need for processing information (Galbraith, 1973). In terms of the problem of incorrect requirements, this theory implies that members of an ISD project can reduce incorrect requirements through a variety of knowledge types and not just by enhancing analytical knowledge. What other types of knowledge may benefit the project? One type of knowledge that may also help developers is business knowledge.

Knowledge of the appropriate business domain, such as the company's key success factors and business functions, may be just as important as technical knowledge. To resolve ambiguities in requirement determination, it has become critical for developers to possess business domain knowledge in order to fully capture users' needs during the eliciting process (Tiwana, 2009). Literature has emphasized the importance of developers acquiring sufficient business domain knowledge (Kaiser & Hawk, 2004; Ghosh & Scott, 2009). The effective interaction among users and developers when identifying actual requirements impacts the performance of requirement determination (Mathiassen, Saarinen, Tuunanen, & Rossi, 2007).

Developers' business knowledge refers to the developers' understanding of the business domain—how to speak the language of business, and how to interact with their business partners (Bassellier & Benbasat, 2004). This business-related knowledge allows developers to understand how to communicate with users in business terms (Preston & Karahanna, 2009). Even though developers may possess excellent technical knowledge for developing a high performance system, if developers' lack business domain knowledge, it reduces the chance of fully capturing users' needs, and in turn, reduces the chance that the next system design will address these needs. In view of this, we hypothesize:

H2: *The level of developers' business knowledge is negatively associated with the level of requirement incorrectness.*

Reduction via Increased Users' ISD Knowledge

Likewise, it is important for users to have knowledge of the developer's domain. The role that users have during an ISD project is described in the literature. Barki et al. (2001) recommend that user participation in the software project may counter requirement-related problems, making possible the production of a high quality system. Hofmann and Lehner (2001) argue that users should be included in the software development process as early as possible so as to increase the amount of business domain knowledge available to the project (Hofmann & Lehner, 2001). Tiwana, Keil, and Fichman (2006) emphasize the importance of the customer's role, viewing the requirement elicitation process as a knowledge-transforming process. The effect of user participation can be extended when common understanding between users and developers exists, which makes it possible to elicit correct requirements, and in turn, renders the making of correct decisions more likely (Tesch, Sobol, Klein, & Jiang, 2009; Tiwana, 2009).

Together, these user-focused studies give credence to the idea that developers alone do not determine the degree to which the needs of users are accurately reflected in the system design. When users express their needs more accurately, developers are better able to internalize and understand the information. The requirements identification involves the user in a communication process. Effective communication is needed to support the collaborative behaviors necessary for requirements elicitation (Mittermeir, Hsia, & Yeh, 1987; Bostrom, 1989; Newman & Robey, 1992).

Possessing ISD knowledge, in addition to business knowledge, helps the requirement elicitation process. Users' ISD knowledge refers to the understanding needed to develop information technology (IT) applications using available technology such as knowledge of programming languages, experience with system development, and understanding of system operation. The abilities of users to operate IT applications are also considered.

One cause of low performance in requirements determination is that users may not fully understand computer capabilities and limitations (Kotonya & Sommerville, 1998). Unrealistic expectations also increase the difficulty of eliciting correct requirements by ultimately reducing the acceptance of the developed system. The possession of IT knowledge allows users to express their needs in a way that developers can understand, which is essential for eliciting correct requirements. Users who possess ISD knowledge are better able to engage in the key system development processes and to respect developers' contributions and opinions (Tiwana & McLean, 2005). Based upon these arguments, we hypothesize:

> *H3*: *The level of users' ISD knowledge is negatively associated with the level of requirement incorrectness.*

Reduction via Increase in Developers' Requirement Analyzing Knowledge

The knowledge necessary to effectively analyze requirements, which we will refer to as requirement analyzing knowledge, includes understanding how to elicit, gather, and organize information from users to produce a high quality system (Ravichandran & Rai, 2000). This type of knowledge is required for successful requirement elicitation (Hickey & Davis, 2004). System analysts (i.e., developers) concentrate on the needs of customers in order to deliver complete and accurate requirements. Developers need to be skilled in the use of scenarios, prototypes, design rationale, or joint application development to elicit correct requirements (Holbrook, 1990; Hsia, Samuel, Gao, Kung, Toyoshima, & Chen, 1994; Wood & Silver, 1995; Sutcliffe & Ryan, 1998; Robertson & Robertson, 1999). Empirical studies indicate that there is a high chance of eliciting incorrect requirements when requirement analysis or elicitation knowledge is unavailable (Nidumolu, 1996). These studies lead us to conclude that developers' requirement analyzing knowledge has an impact on incorrect requirements. Therefore, we hypothesize:

> *H4*: *Requirement analyzing knowledge is negatively associated with the level of requirement incorrectness.*

Coping via Increase in Developers' ISD Knowledge

Having examined the knowledge needed for reduction, we turn to the knowledge needed for coping. Coping concerns the mitigation of the negative effects that arise from requirement incorrectness, without addressing the problem itself (Field et al., 2006). We argue here that there are two types of developer knowledge that, when present, enhance project performance. In addition, we argue that if developers possess such knowledge the application of this knowledge will help mitigate the negative impacts of incorrect requirements, and consequently, will reduce the possibility of negative impacts upon project performance.

Developers' ISD knowledge includes the understanding of a range of development methodologies and tools. Developers use system development methodology to define the structure of the software development life-cycle, and they use knowledge of supporting tools such as data flow diagrams (DFD), data flowcharts, and the entity-relationship (E-R) models to organize the relationship between data cues and system functions for developers. They also use their knowledge of project management tools to plan and control. These methodologies and tools enable the developer to organize the development work and to eliminate possible distractions originating from ineffective management (Sherif & Menon, 2004).

ISD tools are useful because they enable the development team to cope with the negative impacts that arise from failure to identify the correct requirement. The impact of incorrect requirements on project outcome is increased when developers lack the required knowledge of ISD tools (Wallace & Keil, 2004a). It follows that developers' ISD knowledge has both a direct and a moderating effect on the relationships among incorrect requirements and project performance. The direct effect represents the positive benefit of knowledge on performance regardless of the level of requirement incorrectness. The moderating effect implies that negative impacts resulting from incorrect requirements can be suppressed when developers possess the appropriate knowledge. Therefore, we hypothesize both a direct and indirect impact:

H5a: *Project performance is positively associated with developers' ISD knowledge.*

H5b: *The magnitude of the impact of requirement incorrectness on project performance decreases as the level of developers' ISD knowledge increases.*

Coping via Increase in Developers' Behavioral Knowledge

There is an additional type of knowledge considered to be relevant to the problem of coping with incorrect requirements. This is *developers' behavioral knowledge*, which consists of one's understanding of how to work in a collective environment, the ability to support others, a proactive work attitude, the capacity to work closely with different stakeholders, and the ability to maintain working relationships with partners (Ross, Beath, & Goodhue, 1998). In contrast to the "hard" knowledge of ISD (i.e., methodologies, tools, and computer systems), behavioral knowledge focuses on the "soft" knowledge applicable during ISD work. One example of soft ISD knowledge is an understanding of teams. High quality teamwork requires members to know how to coordinate with each other and provide mutual support when needed (Hoegl & Gemuenden, 2001). Developers have behavioral knowledge, and this type of knowledge is important in addition to the technical knowledge they possess (Lee, Trauth, & Farwell, 1995; Mata, Fuerst, & Barney, 1995).

We argue that even though project performance is impaired to some degree by incorrect requirements, the negative effect of incorrect requirements upon project performance is lessened when developers possess high levels of behavioral knowledge. In addition to increasing project performance directly, behavioral knowledge increases developers' capacity to cope with the negative impacts resulting from incorrect requirements.

During the early stages of software development, developers and users play major roles in determining the requirements. If any changes are required in the later stages, the need for cooperation remains. For example, cross-functional cooperation is needed to verify the correctness of developed works, to discover inappropriate designs, and to prompt redesign in the later stages of system development in cases where the initial design fails to reflect users' needs (Hsu et al., 2008). The process of changing the system design requires different stakeholders to work collectively with each other. Negative consequences, such as conflicts, may be unavoidable when inappropriate designs are found. Nevertheless, developers who possess a certain level of behavioral knowledge may tend to ease the negative consequences in order to work collaboratively with other stakeholders and allow the project to proceed. Furthermore, in the later stages of development, various types of specialist expertise are needed. An effective outcome can be generated only with the mutual support of specialists such as project managers, programmers, database administrators, or even network administrators (Tiwana & McLean, 2005). Developers that possess behavioral knowledge improve project performance because they are able to cope more effectively with any negative consequences that may result from incorrect requirements. Both a direct and an indirect impact appear to be important. So, we hypothesize:

H6a: *Project performance is positively associated with developers' behavioral knowledge.*

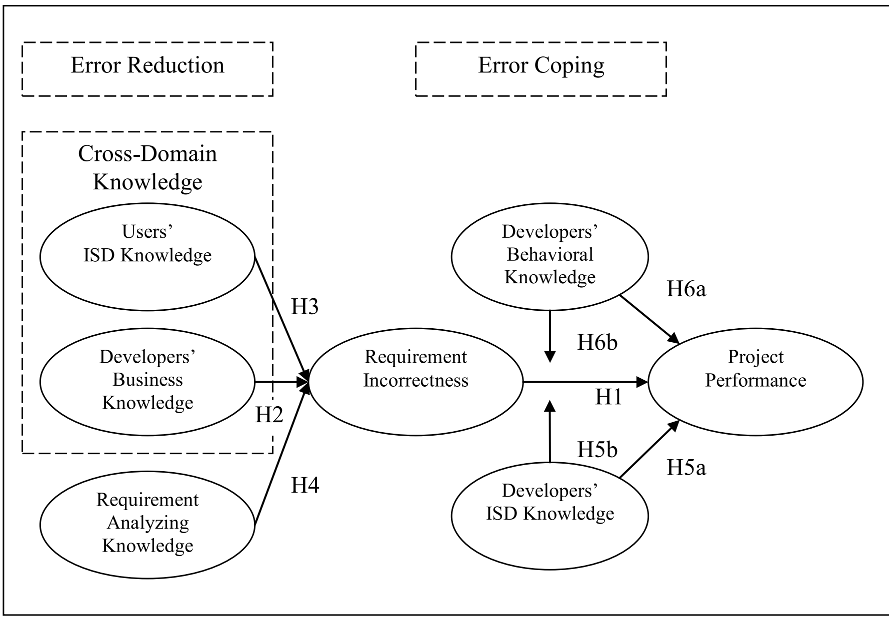
H6b: *The magnitude of the impact of requirement incorrectness on project performance decreases as the level of developers' behavioral knowledge increases.*

A variance model (Figure 2) illustrates the hypothesized relationships among the different types of knowledge possessed by developers and users as they influence the developers' and users' ability to reduce incorrect requirements and to cope with the implications of incorrect requirements upon project performance.

METHODOLOGY

The research model is shown in Figure 2. The data used to examine the proposed model were collected from practitioners using a two-staged approach which included a pilot test phase followed by a questionnaire survey phase. All survey items were adapted from past studies and translated into Chinese by the first author. The translated questionnaire was reviewed by two faculty members, three PhD students, and five practitioners. Minor modifications were made based on the feedback provided. In addition, a Chinese-to-English back-translation was made by the remaining two authors to evaluate the quality of translation. A total of 32

Figure 2: Research model.



part-time MBA students with ISD experience were invited to complete the survey to ensure the quality of our instrument.

We adopted a two-step approach to collect the required data. First, we sent a letter to all 359 members of the Information Management Association (IMA) in Taiwan. IMA is an organization that aims at improve IT usage and enhance communication among IS professionals. Almost every member of this organization is an IS department manager. Many members of IMA are IT department heads of the top 500 companies in Taiwan. Members who were willing to participate in our study were then contacted by telephone. Over the phone, we introduced the major purpose of this study and detailed data collection procedures. The number of project teams in each member's organization was then recorded. Of the 359 IMA members, 158 were willing to participate in this study and identified a number of recently completed ISD projects (ranging from 1 to 10) in their organizations. As such, we identified a sampling frame of 750 candidate ISD projects across the 158 firms. Please note that our unit of analysis is an individual ISD project.

In the second stage, we delivered the survey package to 750 project managers, team leaders, or senior members using contact information collected from the previous stage. On the consent form, two important criteria were set to ensure the quality of response. First, respondents were asked to provide the most appropriate answer for each question based on the most recently completed project in which they were involved. Second, because some survey items attempted to capture user representatives' knowledge of ISD, respondents were required to have experience interacting with user representatives during the system development

Table 1: Sample demographics (N = 250).

Measure	Categories	#	%	Measure	Categories	#	%
Tenure	<4 years	51	20.4	Duration of project	Less than 6 months	98	39.2
	4–10 years	12	48.8		7–12 months	79	31.6
	11–20 years	2	27.2		13–24 months	47	18.8
	>20 years	68	2.8		25–36 months	12	4.8
	Missing	7	0.8		More than 36 months	13	5.2
		2			Missing	1	0.4
Team size	<5	95	38	Educational background	Less than college	12	4.8
	6–10	89	35.6		Bachelor	151	60.4
	11–20	52	20.8		Masters	83	33.2
	21–30	7	2.8		Doctoral	1	0.4
	>30	7	2.8		Missing	3	1.2
Position	Programmer	105	42.0	Industry type	Manufacturing	104	41.6
	SA	46	18.4		Service	46	18.4
	Project leader	42	16.8		Education	8	3.2
	CIO	23	9.2		Finance	20	8.0
	Other specialists	32	12.8		Others	39	15.6
	Missing	2	0.8		Missing	33	13.2
Number of projects in each company	1 Project	71	55.0	Gender	Male	180	72.0
	2 Projects	29	22.5		Female	68	27.2
	3 Projects	10	7.8		Missing	2	0.8
	4 Projects	10	7.8				
	5 Projects	6	4.7	Age	21–30	69	27.6
	6 Projects	2	1.6		31–40	151	60.4
	9 Projects	1	0.8		41–50	26	10.4
					Over 50	4	1.6

process in order to qualify for participation in the survey. The data collection ran from November 1, 2009 to February 28, 2010. We received 279 responses to the survey package, 29 of which were excluded from the following analysis due to missing values. This yielded a valid response rate of 33.3%. Table 1 summarizes the demographic characteristics of the final sample. Because several projects may come from the same company, Table 1 also includes the distribution of respondents by company.

Construct and Measurement

All research variables were measured using multi-item scales from prior research. Some minor revisions were made before the survey was officially delivered. All questions were in Likert scale format anchoring from 1 (strongly disagree) to 7 (strongly agree).

Requirement incorrectness (RI) refers to the extent to which requirements reflect the actual users' needs. We used three items from Wallace and Keil (2004a) to measure the extent to which correct requirements were understood, identified, and captured by developers during the systems analysis stage. Those three items were then reversed to represent the level of RI. All the following analyses are based on the reversed result. With the reversed scales, a higher score indicates a greater degree of RI.

Developers' ISD knowledge (ISDK) refers to developers' understanding of the methodology, support tools, project management tools, and implementation tools used in the target project. A total of four items were adopted from Barki et al. (2001) to measure the extent to which developers were familiar with methodology, support tools, project management tools, and implementation tools used in the target project.

Developers' behavioral knowledge (BHK) refers to the abilities of developers to work in a collective environment, such as being able to support each other, having proactive work attitudes, being able to work closely with different stakeholders, and having the capacity to maintain working relationships with partners (Ross et al., 1998). A total of four items adapted from Fink and Neumann (2009) were used to capture the extent to which developers were proactive, were able to work in a collective environment, and were supportive of each other.

Requirement analyzing knowledge aims at measuring the extent to which developers are able to concentrate on customer needs in order to carry out complete and accurate requirements (Ravichandran & Rai, 2000). A total of three items adapted from Nidumolu (1995) were used to capture the developers' requirement analyzability.

Developers' Business Knowledge relates to developers' knowledge of the business domain. *Users' ISD knowledge* is the user representatives' overall knowledge/expertise in IS development methods and processes. A total of four items for developers' business knowledge and six items for users' ISD knowledge, obtained from Barki et al. (2001) and Tesch et al. (2009), were used to capture developers' familiarity with knowledge in the business domain and user representatives' understanding of ISD.

Project performance refers to the success of the development process itself (Wallace & Keil, 2004a). It was measured using five items adopted from existing scales (Henderson & Lee, 1992; Jones & Harrison, 1996; Guinan, Cooprider, & Faraj, 1998) that tapped into subjects' perceptions of project performance in terms of schedule, budget, and work quality. The details of each item are shown in Table 2.

The complex nature of our research model suggests a need to use structural equation modeling (SEM) with a latent variable approach to verify the measurement and test the proposed hypotheses. Partial least squares (PLS), a component-based SEM technique, is preferable to covariance-based SEM for several reasons. First, PLS is not contingent upon data having a multivariate normal distribution. Most variables in this study are significant at 0.01 (p-value) levels of the Kolmogorov–Smirnov and Shapiro–Wilk normality test results; this implies that our data may not fit the normality requirements of covariance-based SEM. Second, PLS is also an effective technique for exploring causal relationships in high-complexity situations where strong theoretical knowledge about the relationships are not well developed (Wold, 1979). In this study, the proposed model contains numerous factors as well as two moderating effects. A high complexity model reduces the chance of meeting the suggested cut-off values when using SEM (Cheung & Rensvold, 2002). Further, our model is not based on a broadly examined theory. Third, as shown in Table 1, two or more projects may be drawn from the same company. It is inappropriate to assume that those projects are perfectly

Table 2: The results of factor analysis.

Constructs	Items	Factors Loadings	ITC*
Developers' Business Knowledge	***In the most recently completed ISD project that you were involved.***		
CR = 0.895, Alpha = 0.845, AVE = 0.682	1 The developers are knowledgeable about the key success factors that must go right if the company is to succeed.	0.84	0.79
	2 The developers understand the company's policies and plans.	0.79	0.81
	3 The developers are able to interpret business problems and develop appropriate technical solutions.	0.88	0.77
	4 The developers are knowledgeable about business functions.	0.80	0.83
Users' ISD Knowledge	***For those users involved in that project***...		
CR = 0.914, Alpha = 0.888, AVE = 0.641	1 Users are familiar with IT.	0.71	0.88
	2 Users have a lot of experience in IS development.	0.83	0.86
	3 Users are familiar with this application.	0.86	0.87
	4 Users are familiar with the process of IS development.	0.85	0.85
	5 Users are familiar with their role in this project.	0.84	0.87
	6 Users are aware of the importance of their role in this project.	0.76	0.88
Requirement Analyzability	***In the most recently completed ISD project in which you were involved.***		
CR = 0.946, Alpha = 0.923, AVE = 0.814	1 There was a clearly known way to convert user needs to requirements specifications	0.87	0.92
	2 Available knowledge was of great help in converting user needs to requirements specifications	0.92	0.89
	3 Established procedures and practices could be relied upon to generate requirements specifications	0.93	0.88
	4 An understandable sequence of steps could be followed for converting user needs to requirements specifications	0.89	0.90
Requirement Incorrectness	***In the most recently completed ISD project that you were involved.***		
CR = 0.915, Alpha = 0.861, AVE = 0.783	1 User requirements were adequately identified by developers (R)	0.91	0.77
	2 Developers understood user requirements clearly (R)	0.91	0.75
	3 Developers elicited rare incorrect user requirements (R)	0.83	0.86
Developers' Behavioral Knowledge *CR = 0.872, Alpha = 0.806, AVE = 0.632*	1 The developers are self-directed and proactive.	0.79	0.77
	3 The developers have the ability to plan and execute work in a collective environment.	0.89	0.72
	4 The developers work well in cross-functional teams addressing business problems.	0.73	0.75

Continued

Table 2: Continued

Constructs	Items	Factors Loadings	ITC*
	5 The developers are cross-trained to support other IT services outside their primary knowledge domain.	0.77	0.76
Developers' ISD Knowledge	***In the most recently completed ISD project that you were involved.***		
CR = 0.878, Alpha = 0.816, AVE = 0.643	1 Developers understand the development methodology used in this project.	0.81	0.78
	2 Developers understand the development support tools used in this project (e.g., DFD, flowcharts, ER model, CASE tools).	0.87	0.72
	3 Developers understand the project management tools used in this project (e.g., PERT charts, Gantt diagrams, walkthroughs, and project management software).	0.79	0.75
	4 Developers understand the implementation tools used in this project (e.g., programming languages, data base query languages, screen generators).	0.74	0.81
Project Performance	***In the most recently completed ISD project that you were involved.***		
CR = 0.897, Alpha = 0.857, AVE = 0.636	1 This ISD project meets project goals.	0.82	0.82
	2 In this ISD project, expected amount of work completed.	0.81	0.81
	3 In this ISD project, high quality of work completed	0.85	0.81
	4 In this ISD project, there is adherence to schedule.	0.81	0.80
	5 In this ISD project, there is adherence to budget.	0.69	0.84

Note: (R) represents a reversed item. With the reversed scale, a higher score indicates greater degree of requirement incorrectness.
*ITC: Item-total correlation.

independent. Therefore, we adopted a cluster-robust analysis approach, which takes within-group correlation into account, to avoid possible interference. PLS is a two stage path-estimating tool which contains component score generation in the first stage and coefficient estimation, based on least squares, in the second stage. We are able to perform cluster-robust analysis in different software (e.g., STATA) based on the component scores generated from PLS.

In this study, PLS-Graph was used to evaluate the measurement and structural models. To satisfy reviewer requests, we also replicated the results using confirmatory factor analysis (CFA) and SEM techniques in EQS (structural equation modeling software; not reported here), and found these results to be very similar to those from PLS-Graph. We used a two-step procedure including measurement validation and path analysis for data analysis. The validation of measurement includes item reliability, convergent validity, and discriminant validity tests.

To ensure high item-reliability, factor loadings should be greater than 0.7 and item-total correlations should not be lower than 0.3. To ensure reliability, factors with loadings lower than 0.5 should be dropped. As shown in Table 2, all but one of the indicators in this study have loadings greater than 0.7. This exception was kept because the loading value is much higher than the cut-off threshold, 0.5. The item-total correlations are all greater than 0.5.

Convergent validity can be ensured by using multiple indicators to measure one construct. To check for convergent validity, we examine the composite reliability and average variance extracted (AVE) of the constructs (Fornell & Larcker, 1981; Kerlinger, 1986). If the square root of the AVE is less than 0.707, the variance captured by the construct is less than the measurement effort, and the validity of a single indicator and construct is questionable (Fornell & Larcker, 1981). Moreover, to have the required convergent validity, composite reliability should be greater than 0.7. In our study, the respective minimum composite reliability is 0.87 for instrumentality and all AVE values exceed the threshold of 0.707, thus, convergent validity is ensured.

For the required discriminant validity, the correlation between pairs of constructs should be less than 0.90 and the square root of AVE should be greater than the inter-construct correlation coefficients (Fornell & Larcker, 1981; Bagozzi, Yi, & Phillips, 1991; Chin, 1998). As indicated in Table 3, the correlation coefficients range from low to moderate (0.30 to 0.68) and the AVEs are greater than the inter-construct correlations. The results exhibit strong discriminant validity.

Common Method Variance

Because we collected both independent and dependent variables simultaneously from the same respondent, common method variance (CMV) might be a concern in this study. Harman's single factor test was implemented to ensure that there was no significant method effect on the predefined causal relationship. This approach assumes that more than one factor should be generated through a factor analysis process. Both exploratory factor analysis and CFA were conducted. The exploratory factor analysis shows that a total of six factors can be derived (the Eigenvalue for the 7^{th} factor is 0.98), with the first factor explaining 35.7% of the variance.

We also tested the impact of method variance by creating one method variable (using all of the principal constructs' indicators) and linking it to both independent and dependent variables (Podsakoff, MacKenzie, Jeong-Yeon, & Podsakoff, 2003; Pavlou & Gefen, 2005). The impact of this method variable is insignificant, which suggests that the common method bias problem should not be problematic in this study. In addition, following the approach proposed by Podsakoff et al. (2003) and Williams, Edwards, and Vendengerg (2003), we included a common method factor in the PLS model. This factor includes all the principal constructs' indicators. We then calculated the variance substantively explained by the principal construct, and the variance explained by the method, for each indicator. The average substantive variance explained by principal indicators is 0.69 and the average variance

Table 3: Descriptive statistics and correlation matrix.

Variables	Mean	Std. Dev.	M3	M4	Correlation Matrix						
					RI	DBK	UIK	RA	BHK	ISDK	PP
Requirement Incorrectness	2.880	0.926	0.288	−0.552	**0.885**						
Developers' Business Knowledge	5.308	0.854	−0.421	−0.147	−0.445	**0.826**					
Users' ISD Knowledge	4.510	1.030	−0.117	−0.222	−0.403	0.336	**0.801**				
Requirement Analyzability	5.235	0.949	−0.515	−0.497	−0.585	0.317	0.356	**0.902**			
Developers' Behavioral Knowledge	5.411	0.761	−0.113	−0.250	−0.504	0.681	0.375	0.462	**0.795**		
Developers' ISD Knowledge	5.216	0.949	−0.396	−0.178	−0.472	0.422	0.395	0.462	0.525	**0.802**	
Project Performance	5.253	0.912	−0.483	0.118	−0.463	0.359	0.302	0.344	0.423	0.400	**0.797**

Note: The boldfaced data on diagonal of the correlation matrix represent the squared root of AVE.

explained by method is 0.007. The ratio of substantive variance to method variance is approximately 99:1, which exceeds the minimum requirement. In addition, most method factor loadings are not significant. Based on the above evidence, we believe that CMV is unlikely to be a serious concern in this study (Liang, Saraf, Hu, & Xue, 2007).

HYPOTHESES TESTING

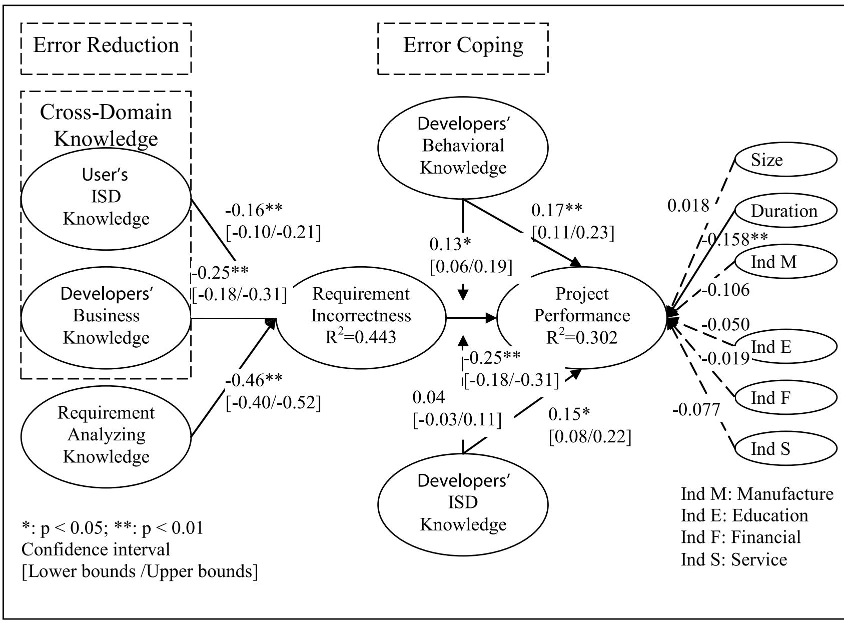
Because the model contains two moderators, we created two interaction terms before conducting path analysis. To create these, we cross-multiplied the items from the independent variable with the moderators. For robustness, we entered the component scores obtained from PLS into STATA and performed two regressions. For one regression we set RI as the dependent variable, and for the other we set project performance as the independent variable; Standard errors were replaced by clustered-robust standard errors for both regressions. The results obtained from STATA are exactly the same as those obtained from PLS. This indicates that our results are not biased due to the inclusion of multiple projects from certain firms.

Figure 3a shows the analysis results of the path model. Given that performance may be affected by the characteristics of project and industrial factors, several control variables including team size, duration of the project, and industry, have been entered simultaneously with model variables. A significant negative coefficient between RI and project performance confirms our hypothesis that project performance is eroded when actual requirements cannot be elicited. Therefore, H1 is supported. For the reduction approaches hypothesized, the path coefficients from users' ISD knowledge, developers' business knowledge, and requirement analysis knowledge to RI indicate that H2, H3, and H4 are supported. Correct requirements can be better elicited when users possess certain ISD knowledge, when developers know the domain's business operation, and when developers have an understanding of requirement eliciting tools.

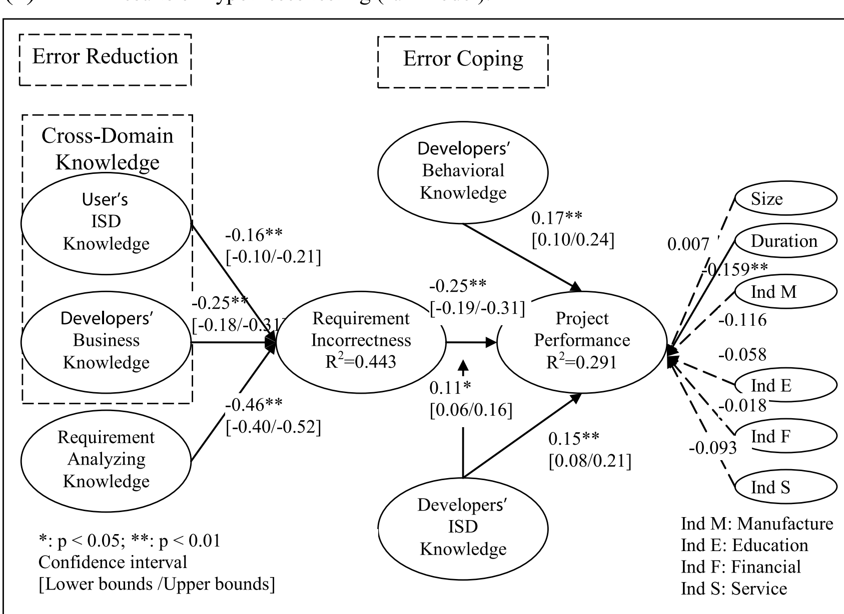
The results show that both ISDK and behavioral knowledge have a positive association with project performance. This indicates that H5a and H6a are supported. However, for the moderating effects, by taking two interaction terms into consideration simultaneously, only behavioral knowledge shows a significant moderating effect on the relationship between RI and project performance. This indicates that H6b is supported but H5b is not.

The only hypothesis not supported is the moderating effect of ISDK on the relationship between RI and project performance (H5b). We conducted an additional analysis, as shown in Figure 3b, to examine whether the moderating effect of ISDK (ISDK $*$ RI) can be observed when the interaction between RI and behavioral knowledge (BHK $*$ RI) is absent. Interestingly, the moderating effect of ISD knowledge is found to be significant when the moderating effect of behavioral knowledge is excluded from the model. In addition, the R-square value drops significantly from 0.290 to 0.279. This finding indicates that variances in performance explained by ISDK $*$ RI are also explained by BHK $*$ RI, but not vice versa. We conclude that members with sufficient hard-knowledge skills will perform better due to the direct impact of ISD knowledge on project performance, and also due to the indirect impact that mitigating the negative impact resulting

Figure 3: (a) Results of hypotheses testing (full model). (b) Results of hypotheses testing (without behavioral K* requirement incorrectness).

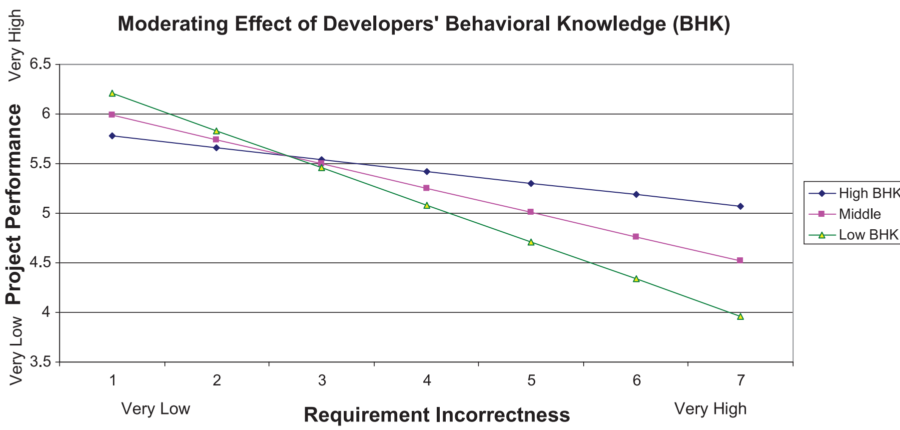


(a) Results of hypotheses testing (full model).



(b) Results of hypotheses testing (without behavioral K* requirement incorrectness).

Figure 4: Moderating effect diagram—Developers' behavioral knowledge.



from incorrect requirements elicited has on project performance. However, it is more critical to have soft-knowledge skills (such as behavioral knowledge) because in comparison to hard-knowledge skills, soft-knowledge skills are more effective at mitigating the negative impacts that result from incorrect requirements.

To further understand how the moderators affect the targeted relationship, we created a moderating-effect diagram (Figure 4). We first obtained coefficients of the independent variable, developer behavioral knowledge, and their interaction term. As suggested by Cohen and Cohen (1983) and Aiken and West (1993), we then used a positive standard deviation (High BHK), an averaged mean (Middle BHK), and one negative standard deviation (Low BHK) to replace the level of behavioral knowledge to illustrate the relationship between RI and project performance at different knowledge levels.

Some significant observations can be made by examining Figure 4. First, when requirements are incorrectly elicited, project performance is low irrespective of the level of behavioral knowledge. Second, project performance diminishes as the level of incorrect requirements increases. However, the slope of project performance to RI is flatter for projects developed by members with high behavioral knowledge, which indicates a low deterioration rate for project performance. In contrast, for those projects where members lack sufficient behavioral knowledge, project performance deteriorates significantly as the level of incorrectness increases. This results in a significant gap between the project performance of those with low behavioral knowledge compared to those with high behavioral knowledge when requirements deviate significantly from the users' actual needs.

DISCUSSION AND CONCLUSION

This study, based on the information processing perspective, aims to understand the effectiveness of knowledge in countering the RI issue in software projects based on the reduction and coping approaches proposed by Field et al. (2006). We hypothesized that incorrect requirements can be reduced by increasing developers' requirement eliciting techniques and promoting cross-domain knowledge between users and developers. We also attempted to demonstrate the effect of the mitigation concept through examination of the moderating effect of ISDK and behavioral knowledge on the relationship between RI and project performance. Data collected from 250 IS practitioners confirmed most of our hypotheses; RI can be reduced when there is a mature requirement eliciting technique, and when users and developers are familiar with the expertise domain of one another. In addition, better performance can be obtained when developers possess mature ISD and behavioral knowledge. Finally, BHK can effectively suppress the negative impact of RI on project performance.

Practical Implications

This study is useful to practitioners in several ways. First, although both reduction capabilities and coping capabilities contribute to the reduction of incorrect requirements and their negative impacts, project managers should pay attention to different knowledge at different stages of system development. The most important implication we obtained from our results is that managers should ensure that developers and users have sufficient cross-domain understanding in order to lower the level of incorrect requirements. Because total elimination of all incorrect requirements is almost impossible, the capability to control negative impacts resulting from incorrect requirements that cannot be eliminated becomes critical. The direct relationship between RI and project performance implies that project performance can be improved through high quality requirement elicitation during the early project stage. This is true because changes to requirements are less likely during the later stages of system development if requirements are correctly determined in an early stage.

It is widely accepted that domain experts, such as end users or customers, should be included in the initial requirement eliciting process to reduce the chance of obtaining incorrect requirements. Our results further suggest that domain experts can better contribute to an ISD project when they are able to explicate domain knowledge in a way that system developers are able to fully understand. Cross-domain understanding can be viewed as the overlap of expertise domains and plays a critical role in the knowledge-transfer process. High levels of cross-domain understanding result in an effective communication process, which is essential for requirement elicitation. Therefore, project managers should pay attention to the member selection process and recruit those developers with strong business domain knowledge. Furthermore, sufficient training should be provided when developers are not familiar with the application domain.

On the other hand, user participation literature asserts that a certain level of IT knowledge or ISD experience is required for users to engage in the development process effectively and generate expected benefits. Our results also highlight the importance of choosing correct user representatives. Users with IT knowledge or system development experience should be invited to join the requirement determination process. With strong IT knowledge, users can fully digest information from developers and then provide adequate feedback. Such IT-literate users can also express their opinions in ways that developers can easily understand. As a result, the chance of eliciting incorrect requirements is reduced.

Second, our study highlights the importance of managers' selecting team players during the formative stage of a project and/or providing adequate training to develop a high performance team. Various factors block the team from achieving the predefined goal effectively. The extent to which the project team has sufficient ISD knowledge and behavioral knowledge to deal with problems determines the final project performance. ISD knowledge and behavioral knowledge allow members to take measures to minimize project inefficiency and overcome barriers so as to attain the predefined project goal. In particular, ISD and behavioral knowledge should be enhanced irrespective of whether requirements have been elicited correctly or not. This is due to their direct effect on performance as well as their moderating effect on the relationship between RI and performance.

Furthermore, behavioral knowledge is needed to mitigate the negative impacts resulting from requirement risks. Although ISDK is able to mitigate the negative impact of incorrect requirements, behavioral knowledge plays a more important role here. While ISD knowledge depicts the ability of IS personnel to use tools (hard-knowledge skills), behavioral knowledge represents their ability to work with different stakeholders collectively (soft-knowledge skills). If systems analysts are unable to identify correct requirements when users do not know what they require or are unable to express their needs in the way that systems analysts can understand, the assurance of final performance then relies on the availability of remedying capability. Our results suggest that remedying capability can be enhanced by having proactive team members who are able to work with different stakeholders or people from different disciplines collectively and are cross-trained to support each other. Those capabilities are not easy to obtain through a formal training process; rather, individuals' social capital and personalities play critical roles. Therefore, in addition to mature hard-knowledge skills (e.g., ISD knowledge), project managers should place an even higher priority on the soft-knowledge skills (e.g., behavioral knowledge) of potential team members during the member selection process.

Academic Implications

This study contributes to the academic literature in a number of ways. With respect to the management literature, based on the reduction and coping concept proposed by Field et al. (2006), we have successfully demonstrated that (i) a project team can reduce incorrect requirements elicited when users and developers have sufficient cross-domain understanding and when mature requirement analyzing techniques are available, and (ii) BHK can mitigate the negative impacts of RI on final project performance. The negative coefficients between three types of knowledge and RI indicate that increasing such knowledge can effectively reduce the eliciting of incorrect requirements. On the other hand, in addition to its direct effect on project performance, the moderating role of coping strategy shows that incorrect requirements generate a less negative impact when developers possess higher levels of behavioral knowledge.

Second, with respect to the project management literature, this study has confirmed that it is important for users and developers to have cross-domain understanding. The impact of cross-domain understanding on project performance has been proposed in previous studies (e.g., Tesch et al., 2009). We have advanced this research stream by employing RI as a mediator between cross-domain understanding and project performance. This indicates that cross-domain understanding can reduce the level of RI, and in turn, can improve project performance. We have also demonstrated that the negative impacts of RI can be reduced when the development team has strong behavioral knowledge. Taking these two findings as one, we highlight an important concept, namely, that it is necessary to enhance capability in both reduction and coping in order to counter errors such as RI during software projects. Because it is impossible to completely eliminate all risk factors, managers are advised to minimize the negative impacts resulting from those risk factors.

Limitations and Future Research Suggestions

This study is not without limitations. First, cross-sectional data collected from practitioners were used to test the proposed concepts. Although many studies adopt cross-sectional data based on individual memory in order to understand the causal relationship between variables, future research may conduct longitudinal observation with multiple-wave data collection, or qualitative study approaches, to verify the concept proposed by this study. Second, we examined this issue from a knowledge perspective only because software development is a knowledge-intensive process. However, because dozens of approaches may be used to reduce RI or counter its negative impact, future research may extend the current study by employing different perspectives and including other approaches. Third, we collected the data from IMA members in Taiwan only. Although we believe that incorrect requirements in ISD projects is a universal phenomenon, western culture-based data may be needed to examine the generalizability of the model proposed in this study. Fourth, because two or more projects were drawn from the same company in some cases, a firm-level control variable, missing in this study, may be needed to control for the impact resulting from different project management capabilities and practices within each firm. Although we used cluster-robust standard errors to exclude the impact of a firm level variable in the analysis stage, future researchers are encouraged to exclude possible interferences in the research design stage. Lastly, the lack of firm-level control variables also limits what can be concluded about RI with respect to an organization's external environment. Although failure to identify correct requirements in the initial stage of a project leads to requirement change, the alteration of the external environment may also be instrumental in changes to the initial determined requirements. Therefore, future studies should take external environmental changes into consideration. In addition, requirements are one of the six risk factors identified by Wallace and Keil (2004a). Future studies are encouraged to explore possible reduction and coping strategies for other important risk factors.

REFERENCES

Aiken, L. S., & West, S. G. (1993). *Multiple regression: Testing and interpreting interactions*. Newbury Park, CA: Sage.

Bagozzi, R. P., Yi, Y., & Phillips, L. W. (1991). Assessing construct validity in organizational research. *Administrative Science Quarterly*, *36*(3), 421–458.

Barki, H., Rivard, S., & Talbot, J. (2001). An integrative contingency model of software project risk management. *Journal of Management Information Systems*, *17*(4), 37–69.

Bassellier, G., & Benbasat, I. (2004). Business competence of information technology professionals: Conceptual development and influence on IT-Business partnerships. *MIS Quarterly*, *28*(4), 673–694.

Boehm, B. (1981). *Software engineering economics*. New York, NY: Prentice Hall.

Boehm, B. W. (1991). Software risk management: Principles and practices. *Software, IEEE*, *8*(1), 32–41.

Bostrom, R. P. (1989). Successful application of communication techniques to improve the systems development process. *Information & Management*, *16*(5), 279–295.

Byrd, T. A., Cossick, K. L., & Zmud, R. W. (1992). A synthesis of research on requirements analysis and knowledge acquisition techniques. *MIS Quarterly*, *16*(1), 117–138.

Campbell, D. T., & Fiske, D. W. (1959). Convergent and discriminant validation by the multitrait-multimethod matrix. *Psychological Bulletin*, *56*(2), 81–105.

Cheung, G. W., & Rensvold, R. B. (2002). Evaluating goodness-of-fit indexes for testing measurement invariance. *Structural Equation Modeling*, *9*(2), 233–255.

Chin, W. W. (1998). Commentary: Issues and opinion on structural equation modeling. *MIS Quarterly*, *22*(1), vii–xvi.

Cohen, J., & Cohen, P. (1983). *Applied multiple regression/correlation analysis for the behavioral sciences*. Hillsdale, NJ: Erlbaum.

Cooper, R. B., & Swanson, E. B. (1979). Management information requirements assessment: The state of the art. *SIGMIS Database*, *11*(2), 5–16.

Curtis, B., Krasner, H., & Iscoe, N. (1988). A field study of the software design process for large systems. *Communications of the ACM*, *31*(11), 1268–1287.

Daft, R. L., & Lengel, R. H. (1986). Organizational information requirements, media richness and structural design. *Management Science*, *32*(5), 554–571.

Davis, G. B. (1982). Strategies for information requirements determination. *IBM Systems Journal*, *21*(1), 4–30.

Fairley, R. (1994). Risk management for software projects. *Software, IEEE*, *11*(3), 57–67.

Faraj, S., & Sproull, L. (2000). Coordinating expertise in software development teams. *Management Science*, *46*(12), 1554–1568.

Field, J. M., Ritzman, L. P., Safizadeh, M. H., & Downing, C. E. (2006). Uncertainty reduction approaches, uncertainty coping approaches, and process performance in financial services. *Decision Sciences*, *37*(2), 149–175.

Fink, L., & Neumann, S. (2009). Exploring the perceived business value of the flexibility enabled by information technology infrastructure. *Information & Management*, *46*(2), 90–99.

Fornell, C., & Larcker, D. F. (1981). Evaluating structural equation models with unobservable variables and measurement error. *Journal of Marketing Research*, *18*(1), 39–50.

Galbraith, J. (1977). *Organization design*. Reading, MA: Addison-Wesley.

Galbraith, J. R. (1973). *Designing complex organizations*. Reading, MA: Addison Wesley.

Gemino, A., Reich, B. H., & Sauer, C. (2007). A temporal model of information technology project performance. *Journal of Management Information Systems*, *24*(3), 9–44.

Ghosh, B., & Scott, J. E. (2009). Relational alignment in offshore IS outsourcing. *MIS Quarterly Executive*, *8*(1), 19–29.

Guinan, P. J., Cooprider, J. G., & Faraj, S. (1998). Enabling software development team performance during requirements definition: A behavioral versus technical approach. *Information Systems Research*, *9*(2), 101–125.

Han, W.-M., & Huang, S.-J. (2007). An empirical analysis of risk components and performance on software projects. *Journal of Systems and Software*, *80*(1), 42–50.

Henderson, J. C., & Lee, S. (1992). Managing I/S design teams: A control theories perspective. *Management Science*, *38*(6), 757–777.

Hickey, A. M., & Davis, A. M. (2004). A unified model of requirements elicitation. *Journal of Management Information Systems*, *20*(4), 65–84.

Hoegl, M., & Gemuenden, H. G. (2001). Teamwork quality and the success of innovative projects: A theoretical concept and empirical evidence. *Organization Science*, *12*(4), 435–449.

Hofmann, H. F., & Lehner, F. (2001). Requirements engineering as a success factor in software projects. *Software, IEEE*, *18*(4), 58–66.

Holbrook, H. (1990). A scenario-based methodology for conducting requirements elicitation. *ACM SIGSOFT Software Engineering Notes*, *15*(1), 95–104.

Hsia, P., Samuel, J., Gao, J., Kung, D., Toyoshima, Y., & Chen, C. (1994). Formal approach to scenario analysis. *Software, IEEE*, *11*(2), 33–41.

Hsu, J. S.-C., Chan, C.-L., Liu, J. Y.-C., & Chen, H.-G. (2008). The impacts of user review on software responsiveness: Moderating requirements uncertainty. *Information & Management*, *45*(4), 203–210.

Jones, C. (1994). *Assessment and control of software risks*. Burlington, MA: Prentice Hall.

Jones, M. C., & Harrison, A. W. (1996). IS project team performance: An empirical assessment. *Information & Management*, *31*(2), 57–65.

Kaiser, K. M., & Hawk, S. (2004). Evolution of offshore software development: From outsourcing to cosourcing. *MIS Quarterly Executive*, *3*(2), 69–81.

Keil, M., Cule, P. E., Lyytinen, K., & Schmidt, R. C. (1998). A framework for identifying software project risks. *Communications of ACM*, *41*(11), 76–83.

Kerlinger, F. N. (1986). *Foundations of behavioral research*. Fort Worth, TX: Holt Rinehart and Winston.

Kotonya, G., & Sommerville, I. (1998). *Requirements engineering: Processes and techniques*. New York: John Wiley & Sons.

Lederer, A. L., & Prasad, J. (1992). Nine management guidelines for better cost estimating. *Communications of ACM*, *35*(2), 51–59.

Lee, D. M. S., Trauth, E. M., & Farwell, D. (1995). Critical skills and knowledge requirements of is professionals: A joint academic/industry investigation. *MIS Quarterly*, *19*(3), 313–340.

Liang, H., Saraf, N., Hu, Q., & Xue, Y. (2007). Assimilation of enterprise systems: The effect of institutional pressures and the mediating role of top management. *MIS Quarterly*, *31*(1), 59–87.

Lyytinen, K., & Hirschheim, R. (1987). Information systems failures—A survey and classification of the empirical literature. In P. Zorkoczy (Ed.), *Oxford surveys in information technology* (257–309). New York, NY: Oxford University Press, Inc.

Mata, F. J., Fuerst, W. L., & Barney, J. B. (1995). Information technology and sustained competitive advantage: A resource-based analysis. *MIS Quarterly*, *19*(4), 487–505.

Mathiassen, L., Saarinen, T., Tuunanen, T., & Rossi, M. (2007). A contigency model for requirements development. *Journal of the Association for Information Systems*, *8*(11), 569–597.

McFarlan, F. W. (1981). Portfolio approach to information systems. *Harvard Business Review*, *59*(5), 142–150.

Mittermeir, R., Hsia, P., & Yeh, R. (1987). Alternatives to overcome the communications problem of formal requirements analysis. In R. D. Galliers (Ed.), *Information analysis: Selected readings*. Reading, MA: Addison-Wesley, 153–165.

Newman, M., & Robey, D. (1992). A social process model of user–analyst relationships. *MIS Quarterly*, *16*(2), 249–266.

Nidumolu, S. (1995). The effect of coordination and uncertainty on software project performance: Residual performance risk as an intervening variable. *Information Systems Research*, *6*(3), 191–219.

Nidumolu, S. R. (1996). A comparison of the structural contingency and risk-based perspectives on coordination in software-development projects. *Journal of Management Information Systems*, *13*(2), 77–113.

Pavlou, P. A., & Gefen, D. (2005). Psychological contract violation in online marketplaces: Antecedents, consequences, and moderating role. *Information Systems Research*, *16*(4), 372–399.

Podsakoff, P. M., MacKenzie, S. B., Jeong-Yeon, L., & Podsakoff, N. P. (2003). Common method biases in behavioral research: A critical review of the literature and recommended remedies. *Journal of Applied Psychology*, *88*(5), 879–903.

Preston, D. S., & Karahanna, E. (2009). Antecedents of is strategic alignment: A nomological network. *Information Systems Research*, *20*(2), 159–179.

Ravichandran, T., & Rai, A. (2000). Quality management in systems development: An organizational system perspective. *MIS Quarterly*, *24*(3), 381–415.

Robertson, S., & Robertson, J. C. (1999). *Mastering the requirements process*. New York: Addison-Wesley.

Roman, G. C. (1985). A taxonomy of current issues in requirements engineering. *Computer*, *18*(4), 14–23.

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

Ross, J. W., Beath, C. M., & Goodhue, D. L. (1998). Develop long-term competitiveness through IT assets. *IEEE Engineering Management Review*, *26*(2), 37–47.

Sage, A., & Rouse, W. (2009). *Handbook of systems engineering and management*. New York: Wiley-Interscience.

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

Schwalbe, K. (2002). *Information technology project management*. Boston, MA: Course Technology.

Sherif, K., & Menon, N. M. (2004). Managing technology and administration innovations: Four case studies on software reuse. *Journal of the Association for Information Systems*, *5*(7), 247–281.

Sutcliffe, A. G., & Ryan, M. (1998). Experience with SCRAM, a scenario requirements analysis method. *Proceedings of the IEEE 3rd international conference on requirements engineering*, 164–171. doi:10.1109/ICRE.1998.667822.

Tesch, D., Sobol, M. G., Klein, G., & Jiang, J. J. (2009). User and developer common knowledge: Effect on the success of information system development projects. *International Journal of Project Management*, *27*(7), 657–664.

Tiwana, A. (2009). Governance-knowledge fit in systems development projects. *Information Systems Research*, *20*(2), 180–197.

Tiwana, A., Keil, M., & Fichman, R. G. (2006). Information systems project continuation in escalation situations: A real options model. *Decision Sciences*, *37*(3), 357–391.

Tiwana, A., & McLean, E. R. (2005). Expertise integration and creativity in information systems development. *Journal of Management Information Systems*, *22*(1), 13–43.

Wallace, L., & Keil, M. (2004a). Software projects risks and their effect on outcomes. *Communications of the ACM*, *47*(4), 68–73.

Wallace, L., Keil, M., & Rai, A. (2004b). How software project risk affects project performance: An investigation of the dimensions of risk and an exploratory model. *Decision Sciences*, *35*(2), 289–321.

Weick, K. (1979). *The social psychology of organizing*. Wokingham, England: Addison-Wesley.

Williams, L. J., Edwards, J. R., & Vendengerg, R. J. (2003). Recent advances in causal modeling methods for organizational and management research. *Journal of Management*, *29*(6), 903–936.

Wold, H. (1979). Model construction and evaluation when theoretical knowledge is scarce: An example of partial least squares. In J. Kmenta, & J. B. Ramsey (Eds.), *Evaluation of econometric models*. New York, NY: Seminar Press, 47–74.

Wood, J., & Silver, D. (1995). *Joint application development*. New York, NY: Wiley.

Jack Shih-Chieh Hsu is an assistant professor in the Department of Information Management, National Sun Yat-Sen University, Taiwan. He received his PhD from the University of Central Florida. His research interests include knowledge management and team management in information system development projects. His research has been accepted or published in academic journals such as *Decision Support Systems*, *Information Systems Journal, Information & Management, Communications of the AIS,* and *International Journal of Project Management.*

Tung-Ching Lin is a professor of information management at National Sun Yat-Sen University in Taiwan. He received his PhD in MIS from the University of Wisconsin-Milwaukee. He has published two books (*Management Information Systems: The Strategic Core Competence of e-Business*, and *Knowledge Management*) and numerous papers in professional journals such as *Decision Sciences*, *Information System Journal*, *Information and Management*, *Computers & Education*, *Electronic Commerce Research and Application*, *Journal of Information Science*, and others. His current research interests include knowledge management, Web2.0, electronic commerce, and management of information systems.

Kuang-Ting Cheng received his PhD from the Department of Management Information Systems, National Sun Yat-Sen University in Taiwan. His current research focuses on knowledge management and project teams. He has presented his papers in several prestigious international conferences, including PACIS, WDSI, and SAIS.

Lars P. Linden received his PhD in management information systems from the University of Central Florida. His research interests include inquiring systems and Web development with free software. His research has been published in several journals, including the *Journal of Educational Technology & Society*, the *International Journal of Actor-Network Theory and Technological Innovation*, and *Communications of the Association for Information Systems*.