Uncertainty profile and software project performance: A cross-national comparison

Kwan-Sik Na a, Xiaotong Li b, James T. Simpson c,*, Ki-Yoon Kim d

a School of Business, Seowon University, 231 Mochung-dong Hungduk-gu, Cheongju-shi Chungbuk 361-742, South Korea
b Department of Accounting and Information Systems, The College of Administrative Science, The University of Alabama in Huntsville, Huntsville, AL 35899, USA
c Center for the Management of Science and Technology, The College of Administrative Science, The University of Alabama in Huntsville, Huntsville, AL 35899, USA
d School of Business, Kwangwoon University, 447-1 Wolgye-dong, Nowon-gu, Seoul 139-701, South Korea

Received 6 June 2002; received in revised form 21 October 2002; accepted 3 November 2002

Abstract

Many software projects are inevitably associated with various types and degrees of uncertainty. It is not uncommon to see software project spiral out of control with escalated resource requirements. Thus, risk management techniques are critical issues to information system researchers. Previous empirical studies of US software firms support the adoption of development standardization and user requirement analysis techniques in risk-based software project management. Using data collected from software projects developed in Korea during 1999–2000, we conduct a comparative study to determine how risk management strategies impact software product and process performance in countries with dissimilar IT capabilities. In addition, we offer an alternative conceptualization of residual performance risk. We show that the use of residual performance risk as an intervening variable is inappropriate in IT developing countries like Korea where the role of late stage risk control remedies are critical. A revised model is proposed that generates more reliable empirical implications for Korean software projects.

Keywords: Software project performance; Requirement uncertainty analysis; Software development standards; Project uncertainty profile; Capability maturity model

1. Introduction

Most software projects inevitably involve various types and degrees of uncertainty. Without proper risk assessment and coordination, software projects can easily run out of control and consume significant additional resource. The development of the Denver International Airport baggage handling system is a recent, highly publicized example. This escalated software project delayed the airport opening by 16 months and overran the budget by almost $2 billion (Montealegre and Keil, 2000).

Software development is a highly complex and unpredictable task since many specialized groups are typically required to collaborate on one project. Unpredictable disruptions in the process can be catastrophic. The information systems (IS) literature suggests that most software development difficulties can be attributed to uncertainties caused by inadequate information (Zmud, 1980). The uncertainties inherent in software development consistently result in high software project failure rates and budget overruns that exceed 50% (Gibbs, 1994). In fact, delayed delivery of software has become an industry norm (Genuchten, 1991; Glass, 1997). Frequently, the damage caused by uncertainties is further compounded by miscommunication, information asymmetry and conflict incentives. Keil and Robey (2001), Smith et al. (2001) and Zhang et al. (2003) provide additional insights into reasons for software project escalation.
Many risk management techniques designed to minimize the negative effects of the risks embedded in software projects have been proposed in the software engineering literature (Boehm, 1991; Chittister and Haimes, 1994, 1996; Fairley, 1994). For instance, development standardization and user requirement analysis are frequently cited as important risk management techniques. It is widely accepted that the effective management of requirement uncertainty can substantially impact project performance. The level of output (i.e., milestones) and behavioral (i.e., SOPs and monitoring) standardization has also been shown to impact project performance. Techniques for managing both requirement uncertainty and standardization have become common procedures in software engineering (Ropponen, 1999). Nidumolu (1996b) tested a model that suggested relationships among these two risk management techniques and both process and product performance (see Fig. 1).

Specifically, Nidumolu proposed that both standardization and requirement uncertainty impact product and process performance through the mediating variable residual performance risk. Residual performance risk reflects only the performance risk present during the later stages of a software project. The model proposed that requirement uncertainty increases residual performance risk and that software development standards reduce such risk. Nidumolu posited that increases in residual performance risk would negatively impact process and product performance. He also suggested that increases in requirement uncertainty would have a direct negative impact on product and process performance. Data from 64 US software projects provide general support for the Nidumolu’s model. Only the proposed negative relationship between requirements uncertainty and product performance was not supported. His conclusion was that residual risk performance mediates the effects of standardization and requirements uncertainty on both process and product performance in his sample of US software firms.

Keil et al. (2000) suggests that some software risk factors are consistent across organizations in different countries while others are country specific. Further, the software organization classification taxonomy, capability maturity model (CMM), suggests that organizations differ in terms of the sophistication of their software engineering and management capabilities. CMM is a well-known comprehensive software process improvement model developed by the Software Engineering Institute at Carnegie Mellon University. It classifies software organizations into five levels based on the sophistication of their engineering and management practices. Organizations that do not have systematic software engineering methods and tools are classified as level 1 or 2. Their software development performances heavily depend on nonstandardized factors such as managers’ experience and competency. Firms classified as 4 or 5 have demonstrated process improvement capabilities that allow them to meet schedule, cost, quality and functionality targets (Paulk, 2001; Paulk et al., 1995). CMM classification is becoming increasingly important since some customers require that the software supplier achieve a CMM rating of at least 3. In fact, these customer requirements have led to the development of fast track processes designed to facilitate rapid CMM classification (Thomas and Smith, 2001).

The primary purpose of this study is to test the Nidumolu model in an IT developing country where the CMM classification is known to be lower than in the US. Thus, this exploratory study proposes to fuel the theory building process by providing descriptive results that should enhance our understanding of how risk management strategies impact software product and process performance in countries with dissimilar IT capabilities.

We test the original Nidumolu model in the Korean software industry. We begin this study by defining the variables and hypotheses derived from the Nidumolu model. Next the methodology for collecting and analyzing the Korean data is discussed, and the results from the test of the original Nidumolu model with Korean data are reported. Next, the rationale and results of a revised model are reported. Finally, the implications of our study are presented.

2. Samples and data collection

The data for this study were obtained from three of the largest software firms in Korea with total employment in excess of 25,000 employees. These firms were selected because of the autonomy of project teams and the large number of partners from other firms participating on each project team. For instance, most project teams included participation by two-to-three external organizations. In cooperation with the Software Quality Management Department Director for each firm, the research team identified 123 software development projects that were completed during 1999–2000. Each project team leader was mailed a questionnaire with an endorsement from their Software Quality Management Department Director. All 123 project teams responded.
resulting in a 100% response rate. The summary statistics for the 123 projects are presented in Table 1.

Since the purpose of the study is to replicate the Nidumolu study (1996b), we used the same instrument used in the US study. Sample items are presented in Table 2. Two bilingual Korean nationals using a double-back translation process developed the Korean version.

3. Measures

Five theoretical constructs are defined in the original model (Nidumolu, 1996b). The two dependent variables are process and product performance. Process performance is a performance metric for the software development process and can be described by the (1) learning that occurs during the course of the project, (2) the degree to which management controls the project, and (3) the quality of the interactions between the IS team and users during the development process. Product performance is a performance metric that captures the performance of the finished product and can be described by the (1) technical performance of the software, (2) the degree to which the software conforms to user needs, and (3) the degree to which the software is flexible in supporting new products and changing user needs.

The antecedent variables are requirement uncertainty and standardization. Requirement uncertainty refers to the discrepancy between the information necessary to identify the user requirements and the information available to the developers. This construct was captured by three important dimensions: (1) the instability or changes in requirements over the life of the project, (2) the heterogeneity of requirements among the users, and (3) the degree to which the process for converting user need to product requirements can be reduced to a set of objective procedures. While standardization can occur at several levels in the organization, we consider only standardized control. We consider both output control standardization and behavioral control standardization. Output control refers to the extent to which management prescribes milestones and outcomes for the development team. Behavioral control refers to the extent to which management prescribes the tools and techniques that are to be used in software development. The mediating variable residual risk performance is the risk assessed during the final stages of the project after project planning and requirement analyses have been completed. In other words, it is the difficulty in predicting performance consequences during the final stages of product development.

The purpose of this study requires that we follow the same measure evaluation and scale development process used in the original US study. The results from the measurement analysis of the Korean data presented in Table 3 are very similar to the results from the US study. The Korean data reliabilities that range from 0.65 to 0.84 are similar to the 0.65 to 0.86 reported in the US study. These reliability scores reveal an acceptable level of internal consistency for both studies. Further, the factor structures from the Korean data conform to the a priori measurement models and is consistent with the results from the US study. Finally, we used the US study process to create the composite variables. The composite variables were created by combing the appropriate dimensions for each variable. Each dimension was weighted by the factor coefficient derived from the composite variable’s first principal component. Therefore, the process for evaluating and creating the variables used in the model were identical for both the US and Korean studies.

<table>
<thead>
<tr>
<th>Table 1 Project profile statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project attributes</td>
</tr>
<tr>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Project cost (in thousand dollars)</td>
</tr>
<tr>
<td>Effort (person months)</td>
</tr>
<tr>
<td>Project duration (months)</td>
</tr>
</tbody>
</table>

\(N = 123\) projects.

<table>
<thead>
<tr>
<th>Table 2 Response formats and sample items</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measures</td>
</tr>
<tr>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>Standardization</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Requirements uncertainty</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Residual performance risk</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Process performance</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Product performance</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

\(^a\) All were five-point scales.
4. Hypotheses

Corresponding to the theoretical hypotheses proposed in Nidumolu (1996b), we test the following six hypotheses using the Korean data:

H1: Increase in requirements uncertainty will be directly associated with increase in residual performance risk for Korean software projects.

H2: Increase in requirements uncertainty will be directly associated with decrease in process performance for Korean software projects.

H3: Increase in requirements uncertainty will be directly associated with decrease in product performance for Korean software projects.

H4: Increase in residual performance risk will be directly associated with decrease in process performance for Korean software projects.

H5: Increase in residual performance risk will be directly associated with decrease in product performance for Korean software projects.

H6: Increase in standardization will be directly associated with decreases in residual performance risk for Korean software projects.

5. Results

Structural equation modeling (SEM) using AMOS 4 was used to test the preceding hypotheses. The results of the analysis of the Korean data are presented in Figs. 2 and 3.

To facilitate comparison of the US and Korean findings, we report the results of the US analysis in (Nidumolu, 1996b) in parentheses. We use plus signs to designate statistically significant ($p < 0.05$) positive path coefficients. Minus signs represent significant ($p < 0.05$) negative path coefficients. NS is used to denote nonsignificant ($p > 0.05$) coefficients. Figs. 2 and 3 reveal that all path coefficients in Nidumolu (1996b) are significant and in the proposed direction. However, the results of the Korean analysis reveal that the coefficients repre-
senting the relationship between residual performance risk and the two performance measures are not significant. Consistent with the US study findings, there is strong support for H1, H2, H3 and H6.

Tests of overall model fit for the US data revealed that the data fit the model very well. The Normed fit indices (NFI) for both US models exceeded the 0.9 recommended minimum value. The chi-square values for both models exceeded the 0.05 minimum cutoff. Finally, the Wald and Lagrange multiplier (LM) test indicated that no model modifications would significantly improve the fit statistics. Similar tests of the fit of the Korean data to the models suggest that the original models are not as robust when applied to the Korean data. Thus, the evidence from analysis of the Korean data should not be considered conclusive.

6. Revised model

Figs. 2 and 3 show that both path coefficients linking residual performance risk and the two performance constructs are nonsignificant for the Korean data. One possible explanation for this difference in the US and Korean data is that the residual performance risk is less relevant in Korean software firms. The residual performance risk concept is based on the notion that software risk assessment is a prerequisite to successful software risk management. Typically, a comprehensive assessment of ex ante risks associated with a software project requires the identification of a set of risk factors. Although many studies have attempted to compile a comprehensive list of common project risk factors, no validated and unambiguous risk factor list has been agreed upon (Barki et al., 1993; Jiang and Klein, 2000; Moymhan, 1997; Schmidt et al., 2001). One major problem with developing a common list of risk factors is that recent studies show that project managers and users frequently have different perception of risk factors (Keil et al., 2002).

These limitations of the traditional project risk management approach have led Meyer et al. (2002) to propose a more forward-thinking approach that focuses on a project’s uncertainty profile. These researchers argue that, in addition to foreseeable uncertainty that can be controlled by traditional risk management techniques, many innovative projects contain unforeseeable uncertainty and chaos. Consistent with this project risk profile perspective, we decompose residual risk performance into two parts with the following expression:

\[ \text{residual performance risk} = \text{residual controllable risk} + \text{unforeseeable risk} \]

The residual controllable risk is the uncertainty that continues to exist during the later stages of a software projects, but still can be controlled in various ways. The unforeseeable risk is the uncertainty that can be neither identified nor controlled during project planning. Based upon this decomposition, we propose that residual performance risk will be higher in Korea than in the US. First, most Korean companies are behind their US counterparts in early stage software project risk management. Na (1999) estimates that most Korean software firms are at level 1 or level 2 according to the CMM. According to Software Engineering Institute's (SEI) (2002) profile of software firms, less than 10 South Korean software firms participated in the 2001 SEI Software Capability Evaluation. Approximately 1500 US firms participated during the same period. In fact, no Korean firm participated in the CMM assessment prior to 1998 (SEI, 1998). The majority of all CMM assessments have involved US software firms. Due to the absence of systematic risk management tools, the residual controllable risk in Korean projects is likely to be greater than in US projects.

Second, the Korean data was collected during 1999–2000, a period characterized by the Internet mania. Lee (1999) found that the “Internet Rush” had begun in Korea just prior to 1999 and that the internet business has spread into all industries. Further, he...
projected that the internet market would grow at 200% per year. Many of the Korean software projects surveyed are Internet-related and quite innovative. Meyer et al. (2002) suggest that innovative projects or projects undertaken in a volatile environment tend to experience more unforeseeable uncertainties. These uncertainties are common with internet projects since the technology for integrating existing software with the internet is immature and standards are still under development. Further, as projects become more volatile, the focus of management strategies shifts from the exchange of structured information along defined interfaces to exchange of unstructured information along emerging interfaces. Simply tracking the progress of a project becomes less important than tracking the project’s basic assumptions and unknowns. Finally, these researchers report that their study of projects representing the personal-computer, telecommunications, pharmaceutical, internet startups, metal processing, and building construction industries revealed that internet startups experience the largest amount of chaos in project development.

We propose that the greater unforeseeable risk expected in Korean software projects will inflate the total residual performance risk. Thus, we offer the following hypothesis:

**H7**: Korean software projects’ mean residual performance risk is significantly greater than that of US software projects.

A one-sided $t$-test was used to determine whether the mean value of residual performance risk in Korean firms (3.10) is significantly greater than the mean value of residual performance risk in US firms (2.35). The results strongly support H7 ($t = 5.92$, $p < 0.001$). Residual performance risk is significantly higher for Korean projects.

Intuitively, the residual controllable risk should not negatively impact the project performance if there are some remedial ways to control the risk during the later development stage. For instance, it is suggested that a project manager’s experience can compensate for insufficient risk management tools (Henderson and Lee, 1992; Moynihan, 1997). In addition, project-wide coordination can improve software development performance (Andres and Zmud, 2002; Nidumolu, 1995, 1996a). Nidumolu (1996b) does not incorporate the role of late stage remedies like coordination and experience in his study of US firms. Thus, it is reasonable to expect a poorer fit of the data in the Korean since many Korean projects rely on late stage remedies to further reduce the residual controllable risk. Therefore, examining the role of residual performance risk in Korean firms without controlling late stage remedial variables is problematic. Moreover (Kitchenham and Linkman, 1997) found that uncertainty estimates are usually inaccurate for most software projects. Jiang et al. (2002) also reported having difficulty fitting their software project data to a SEM with performance risk as an intervening variable. These observations along with the nonsignificant relationship between residual performance risk and both performance measures in Korean firms suggest an alternative model. We propose a revised model that will directly examine the effectiveness of standardization and requirement analysis on the two performance constructs (see Fig. 4).

The robustness of the findings in both the US and Korean studies suggest that we omit only residual performance risk from the original model. Nevertheless, we do offer a different structural model that incorporates both process and product performance in the same model. The revised model recognizes that process performance measures temporally precede product performance measures. Thus, the revised model also examines the inter-temporal relationship between process and product performance.
7. Analysis of revised model

The nonsignificant $\chi^2$-value ($p$-value = 0.124) derived from the revised model suggests that the estimated covariance matrix does not differ significantly from the observed sample covariance matrix. Further the NFI of 0.998 exceeds the recommended 0.90 minimum and provides additional evidence that the model is a good representation of the data. These findings suggest that the revised model is empirically superior to the original models. The path coefficients presented in Fig. 4 represent the relationships among the variables in the model. The significance level for each path coefficient is in parentheses. The results provide strong support H8, H9, H10 and H11. Increased standardization positively impacts both process and product performance. Requirement uncertainty negatively impacts product performance, and increases in process performance is associated with increased product performance.

These results suggest that the revised model provides a more robust tool for examining the effectiveness of requirement analysis and development standardization in software project management in developing IT countries. However, user requirement analysis and standardization are both examples of traditional software risk management tools, and they are generally inadequate techniques for addressing unforeseeable uncertainty or chaotic situations. Managers of software projects that are either innovative or are developed in a highly volatile environment must depend on supplement traditional risk management tools with more forward-thinking approaches.

8. Summary and discussion

Most software projects are associated with various types and degrees of uncertainties. Many studies have proposed various risk management techniques to achieve effective risk identification and control. An important study of US software companies has shown that development standardization and user requirement analysis are effective techniques for reducing latter stage risk which impacts software project performance (Nidumolu, 1996b).

The present study addresses the fundamental question of whether IT capability impacts the relationships among risk management techniques, risk and project performance. Using identical structural models and data collected from Korean software projects during 1999–2000, we compared the findings from the previous US study with the finding from the Korean study. Using the same analysis techniques, we found that the original theoretical model developed for the US study is not very robust when applied to the Korean data. Further, the analysis revealed that the mean value of residual risk performance and its impact on project performance differed significantly in the two countries. The authors suggest that IT capability helps to explain these differences.

Compared to the US, systematic risk management is still considered to be very immature in IT developing counties like Korea. Therefore, modeling a project’s residual performance risk without controlling for other later stage risk control remedies that are important in developing countries is likely the primary reason for the differences.

We offer a revised structural equation model for developing IT counties that eliminates residual performance risk. The revised model was tested using the Korean data. The results revealed that software development standardization enhances both development process performance and software product performance. Further, requirement uncertainty analysis is effective in improving software product performance. Finally, there is significant inter-temporal correlation between development process performance and software product performance.

Although the residual performance risk was excluded from the revised model for developing countries, we believe that it is a theoretically important concept. Our analysis shows that the mean level of residual performance risk is significantly greater in Korean software
projects than in US software projects. The decomposition of residual performance risk into residual controllable risk and unforeseeable risk provides one potential explanation for this difference. The Korean data was collected during a period described as Internet mania. We believe that the highly volatile external environment and the innovative nature of many software projects developed during this period increased the unforeseeable uncertainties. In addition, the relative immaturity of many Korean software companies’ risk management practices also may have inflated the residual controllable risk. This interpretation recognizes that some types of uncertainties or chaos go beyond the realm of traditional risk management techniques. Moreover, when significant amount of unforeseeable uncertainty exists, learning and flexibility become critical parts of a project manager’s risk planning and reduction strategies.

Since the Korean survey was designed to replicate the previous US study, we did not collect data that would allow us to decompose residual performance risk. Thus, we could not determine how much of the residual risk was unforeseeable uncertainty and how much was controllable through other risk control techniques or remedies during the later software development stages. As Korean software project managers continue to improve their risk management practice, it is reasonable to expect that the average residual controllable risks will decrease over time. However, as software development becomes more innovation-driven and the technologies continue to advance rapidly, more unforeseeable uncertainties or even chaotic situations may emerge. We believe that future studies should examine both components of residual performance risk.

Acknowledgement

The authors would like to thank Ms. Lynsey Delane for her assistance in the preparation of this manuscript.

References


Dr. Kwan-Sik Na is an Associate Professor in the Business at the Seowon University in Korea and Visiting Scholar in the Center for the Management of Technology at the University of Alabama in Huntsville. He received his Ph.D. in Management Information Systems and Management Science at the Kwangwoon University in Korea. His research focuses on software and information system risk management, decision support system and software industry research. He has published over 30 articles in these areas.

Dr. Xiaotong Li is an assistant professor of MIS at the University of Alabama in Huntsville. He received his Ph.D. in MIS from the University of Mississippi and worked as a research assistant at the Robert M. Hearn Center for Enterprise Sciences. His research has appeared in Communications of the ACM, Information Resource Management Journal and International Journal of Information and Management Sciences. His major research interests are in real options application in IT and economics of technology.

Dr. James T. Simpson is Professor of Marketing, Chairman of the Department of Management and Marketing and Director of the Center for the Management of Science and Technology at the University of Alabama in Huntsville. He received his Ph.D. in Marketing and Applied Statistics at the University of Alabama. His research focuses on technology management, marketing high technology product and services, and marketing channel structure and behavior. His research has appeared in numerous journals including the Journal of Marketing Research, Marketing Letters, Journal of Business Research, and European Journal of Innovation Management. He has lectured and served as a visiting scholar at universities in Russia, Byelorussia, China, Romania, and England.

Dr. Ki-Yoon Kim is Professor of Business Administration, Chairman of the Department of Business Administration at the Kwangwoon University in Korea. He received his Ph.D. in Management Science at the Korea University in Korea. His research focuses on security management, software and information system risk management. He has published over 50 articles in these areas including Journal of Information System Education.