Decision Sciences
Volume 35 Number 2
Spring 2004
Printed in the U.S.A.

How Software Project Risk Affects Project Performance: An Investigation of the Dimensions of Risk and an Exploratory Model*

Linda Wallace†
Department of Accounting and Information Systems, Virginia Polytechnic Institute and State University, 3007 Pamplin Hall (0101), Blacksburg, VA 24061, e-mail: wallacel@vt.edu

Mark Keil
Department of Computer Information Systems, J. Mack Robinson College of Business, Georgia State University, Atlanta, GA 30303, e-mail: mkeil@gsu.edu

Arun Rai
Center for Process Innovation and Department of Computer Information Systems, J. Mack Robinson College of Business, Georgia State University, Atlanta, GA 30303, e-mail: arunrai@gsu.edu

ABSTRACT

To reduce the high failure rate of software projects, managers need better tools to assess and manage software project risk. In order to create such tools, however, information systems researchers must first develop a better understanding of the dimensions of software project risk and how they can affect project performance. Progress in this area has been hindered by: (1) a lack of validated instruments for measuring software project risk that tap into the dimensions of risk that are seen as important by software project managers, and (2) a lack of theory to explain the linkages between various dimensions of software project risk and project performance. In this study, six dimensions of software project risk were identified and reliable and valid measures were developed for each. Guided by sociotechnical systems theory, an exploratory model was developed and tested. The results show that social subsystem risk influences technical subsystem risk, which, in turn, influences the level of project management risk, and ultimately, project performance. The implications of these findings for research and practice are discussed.

Subject Areas: Sociotechnical Systems Theory, Software Project Risk, and Structural Equation Modeling.

---

*The authors would like to thank the Project Management Institute's Information Systems Special Interest Group (PMI-ISSIG) for supporting this research. We would also like to thank Georgia State University for their financial support through the PhD research grant program. The authors gratefully acknowledge Al Segars and Ed Rigdon for their insightful comments and assistance at various stages of this project.

†Corresponding author.

INTRODUCTION

With information technology playing an increasing role in the economy, companies have grown more heavily dependent on the successful delivery of information systems (IS). Yet, many software projects result in systems that do not function as intended, are not used, or are never delivered (Gordon, 1999; Johnson, 1999). As organizations continue to invest time and resources into strategically important software projects, managing the *risk* associated with such projects becomes a critical concern. Failure to understand, identify, and manage risk is often cited as a major cause of IS project problems such as cost and schedule overruns, unmet user requirements, and the production of systems that do not provide business value (Alter & Ginzberg, 1978; Barki, Rivard, & Talbot, 1993; Boehm, 1991; Charette, 1989; McFarlan, 1981).

Advocates of IS risk management claim that by identifying and analyzing threats to success, actions can be taken to reduce the chance of project failure. Researchers have often stressed the importance of empirically categorizing the sources and types of risks associated with software development projects to better prioritize and assess the possible exposure and losses that may result (e.g., Boehm, 1991; McFarlan, 1981). Unfortunately, relatively few tools are available for identifying software project risk factors. Moreover, there is a lack of theory to explain the linkages between various dimensions of software project risk and project performance. While various risk checklists (e.g., Boehm, 1991) and frameworks (e.g., Keil, Cule, Lyytinen, & Schmidt, 1998) have been proposed, the software project risk construct, its underlying dimensions, and the relationship of these dimensions to project performance remain largely unexplored. Until there is a better understanding of risk and its effects on software projects, project managers will face difficulties in developing the appropriate risk mitigation strategies that enable them to produce successful information systems.

The purpose of the research was: (1) to identify underlying dimensions of software project risk and to develop and validate an instrument for measuring software project risk, and (2) to build and test a model guided by theory that relates the risk dimensions to project performance.

The remainder of the paper is organized into five sections. First, we present a brief review of the literature, highlighting six dimensions of software project risk. Second, guided by the project management literature and sociotechnical systems theory, we introduce a model of risk and project performance that builds on the six dimensions. Third, we describe the development and validation of a risk instrument designed to measure the six dimensions. Fourth, the model of risk and performance is tested using structural equation modeling and the results are discussed. Finally, we summarize the implications of our work for both research and practice.

LITERATURE REVIEW

In decision theory a risk may lead to either positive or negative consequences, and the concept of risk reflects the variation in the distribution of possible outcomes (Arrow, 1970). Thus, a risky alternative is one for which the variance is large. The view of risk used in decision theory, however, is *not* consistent with the findings from empirical studies of how managers define risk (March & Shapira, 1987). For managers, the uncertainty associated with positive outcomes is not considered to be a risk at all; only the threat of negative outcomes is considered a risk. Thus, managers seek to influence their environment so as to bring risk (i.e., possible negative outcomes) under control (MacCrimmon & Wehrung, 1984). Consistent with the views of March and Shapira (1987), we define software project risk factors as conditions that can pose a serious threat to the successful completion of a software development project.

Advocates of software project risk management suggest that project managers should identify and control risk factors to reduce the chance of project failure. If the key elements to be controlled are the project risk factors (Karolak, 1996), then the process of risk assessment must begin with the identification of these factors. Though several lists of risk factors have been published in the literature (e.g., Alter & Ginzberg, 1978; Boehm, 1991; Heemstra & Kusters, 1996; McFarlan, 1981; Moynihan, 1997; Schmidt, Lyytinen, Keil, & Cule, 2001), there has been little attempt to move beyond checklists to establish the underlying dimensions of the software project risk construct or to develop good instruments for assessing software project risk.

Unfortunately, there is no general agreement as to the dimensionality of the software project risk construct. For example, in his widely cited article on managing software project risk, McFarlan (1981) identifies three dimensions of risk: project size, project structure, and experience with the technology. The derivation of these particular dimensions is not specified, and McFarlan does not provide an instrument specifically designed to measure these dimensions of risk. As an example of how a manager might measure risk, McFarlan introduces a sample of a 54-item risk assessment questionnaire used by a company for measuring software project risk. While the Dallas Tire case (HBS case no. 9-180-006) is cited as the source for the questionnaire items presented in the article, the exact origin of the questionnaire and its psychometric properties remain unknown. Even McFarlan (1981, p. 144) himself points out that, "no analytic framework lies behind these questions."

There has been only one other previous attempt to provide a measure of software project risk that we are aware of (Barki et al., 1993). Barki et al. (1993) reviewed the IS literature and compiled a list of 35 risk variables, which formed the basis for a questionnaire consisting of 144 items. They collected data, compared several factor solutions, and found the most interpretable solution to consist of five factors, which they labeled: technological newness, application size, lack of expertise, application complexity, and organizational environment. In refining their instrument, they retained 23 variables related to uncertainty, measured by 83 items.

While the instrument developed by Barki et al. (1993) represents a significant advance in software risk measurement, there was no attempt to involve practicing project managers in the identification or validation of risk items or the factors that emerged from their analysis. In their initial factor analysis, a nine-factor solution emerged, but was judged to be uninterpretable. Therefore, a five-factor solution was imposed because it was easiest to interpret. However, a recent study based on a subset of the same variables suggested a six-factor solution (Jiang & Klein, 2001). Thus, the dimensionality of the software project risk construct remains open to question.

The variables on the Barki et al. (1993) instrument were measured with a range of single-item and multi-item binary scales, ratio scales, interval scales, and semantic differential scales. The large number of single-item measures makes it impossible to assess the measurement reliability for many of the variables in the instrument, and the broad range of different measurement scales makes it cumbersome to calculate an overall measure of project risk. In order to assess overall risk, each of the variables can be converted to a binary scale and averaged, however, this reduces the precision of the instrument.

Given the limitations of both the checklist approach and existing instrumentation for measuring software project risk, a need remains for a rigorously validated instrument that is grounded in the IS literature and validated with practicing software project managers. Therefore, our first step was to develop a comprehensive picture of the underlying dimensions of software project risk (Smith, Milberg, & Burke, 1996). First, an extensive literature review was performed to gain more insight into the factors that may increase the riskiness of a software project. In addition to examining the IS academic literature, articles written by practitioners detailing their experiences with software development projects were reviewed (e.g., Casher, 1984; Keider, 1984). Both types of literature were used to produce an extensive list of factors that have been identified as potentially contributing to IS failure. Similar threats were grouped together to get a clearer conceptualization of the general types of software project risk factors. Commonly cited problems in software development projects were sorted in an iterative and interpretive manner, and categories were combined and modified in an intuitive manner to suggest underlying dimensions of software project risk. This approach is consistent with how other researchers have gone about the process of defining the domain of a construct (e.g., Moore & Benbasat, 1991; Smith et al., 1996). Several iterations were performed to develop a classification scheme in which the categories or dimensions of risk were as distinct as possible. This procedure resulted in the identification of six dimensions of risk, which are described briefly in Table 1 along with representative references.

A MODEL OF RISK AND PERFORMANCE

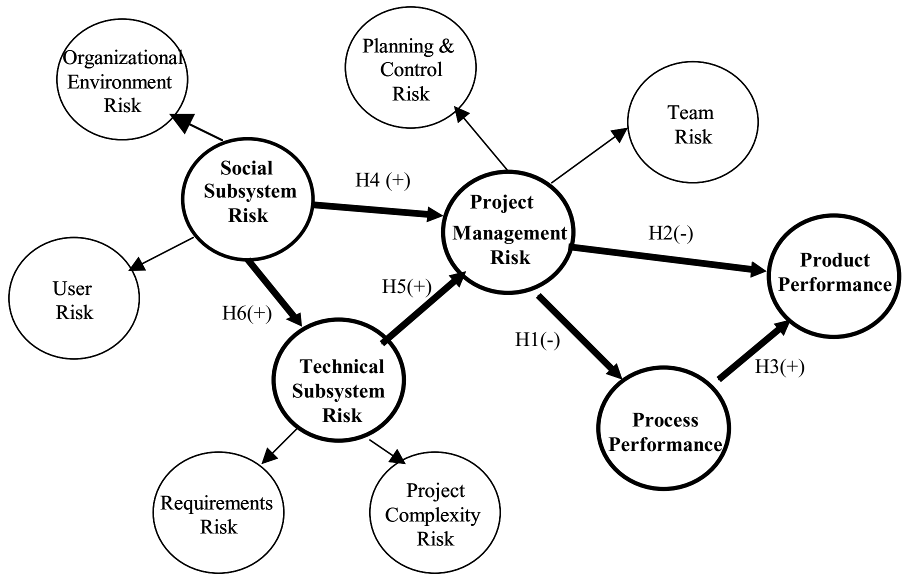
In this section, we present and describe a model (see Figure 1) that uses the six dimensions of software project risk as building blocks for creating higher-level latent constructs, which are then used to examine the relationship between risk and project performance. The model's development was guided by the project management literature and sociotechnical systems theory.

Central to our model is the notion that project management has an impact on project performance, or outcome. Our conceptualization of project performance includes both product and process performance (Nidumolu, 1996). Product performance refers to the successfulness of the system that was developed, whereas process performance refers to the successfulness of the development process itself (i.e., extent to which the project was delivered on schedule and within budget). Project management includes a number of key processes, such as planning and control (Project Management Institute, 2000), and requires an effective project manager to ensure that these processes are in place. To be effective, the project

Table 1: Six dimensions of risk.

Dimension	Description	Representative References
Organizational Environment Risk	The risk or uncertainty surrounding the organizational environment in which a software project takes place was identified as a major area of project risk. Factors such as organizational politics, the stability of the organization environment, and organizational support for a project have been shown to impact project performance.	(Ewusi-Mensah & Przasnyski 1991; Jarvenpaa & Ives 1991; Jones, 1994; O'Toole & O'Toole, 1966)
User Risk	The lack of user involvement during system development is one of the most often cited risk factors in the literature. If the attitudes of users toward a new system are unfavorable, then it is likely that they will not cooperate during a development effort, leading to an increased risk of project failure	(Davis, 1982; Keider, 1984; Robey & Farrow, 1982; Tait & Vessey, 1988)
Requirements Risk	Uncertainty surrounding system requirements is another major factor that can impact project performance. Frequently changing requirements are not the only possible requirements-related problem associated with system development projects. Incorrect, unclear, inadequate, ambiguous, or unusable requirements may also increase the problems, or risks, associated with a software development project	(Boehm, 1991; Charette, 1989; Schmidt et al., 2001, Thayer, Pyster, & Wood, 1980)
Project Complexity Risk	The inherent complexity of a software project, in terms of the difficulty of the project being undertaken, represents another dimension of software project risk. There are several attributes of a project that can indicate how complex it is, such as whether new technology is used, if the processes being automated are complex, and if there are a large number of required links to existing systems and external entities.	(Barki et al., 1993; Kemerer & Sosa, 1988; McFarlan, 1981; Schmidt et al., 2001)
Planning and Control Risk	The planning and control of the software development process adds another dimension to the riskiness of a project. Poor planning and control often leads to unrealistic schedules and budgets and a lack of visible milestones to assess whether the project is producing the intended deliverables. Without accurate duration estimates, managers do not know what resources to commit to a development effort. The net result is often excessive schedule pressure or unrealistic schedules that can increase project risk.	(Jones & McLean, 1970; Keider, 1984; Metzger, 1981; Moore, 1979, Thayer et al., 1980)
Team Risk	Team risk refers to issues associated with the project team members that can increase the uncertainty of a project's outcome, such as team member turnover, insufficient knowledge among team members, cooperation, motivation, and team communication issues.	(Abdel-Hamid, 1989; Alter & Ginzberg, 1978; Brooks, 1987; Jiang, Klein, & Means, 2000; Schmidt et al., 2001)

Figure 1: A model of risk and performance.



manager must coordinate the activities of a diverse project team and he or she must ensure that the team members have the skills to carry out the project. Thus, the Project Management Risk construct can be modeled as a construct measured by two key underlying risk dimensions: Planning/Control and Team.

One would expect that project management influences both product and process performance. The linkage to process performance is readily apparent in that poorly managed projects are more likely to exhibit cost and schedule overruns. The linkage to product performance is subtler. Nevertheless, it is reasonable to expect that poorly managed projects are more likely to yield systems that fail to satisfy user needs or are otherwise judged to be less successful on dimensions such as satisfaction or quality. Furthermore, we posit a linkage between process and product performance; projects that incur cost and schedule overruns are simply less likely to deliver a successful product. Thus, we state the following hypotheses:

H1: Project Management Risk will have a significant negative impact on Process Performance.

H2: Project Management Risk will have a significant negative impact on Product Performance.

H3: Process Performance will have a significant positive impact on Product Performance.

To guide the development of our model further, we turned to concepts from sociotechnical systems theory, which emphasizes the fit between technical and social subsystems (Pava, 1983; Trist, 1981). Sociotechnical systems theory (SST) arose in the early 1950s from some field research on the British coal mining industry. Researchers from the Tavistock Institute discovered a South Yorkshire coal-field where new technology had allowed the formation of relatively autonomous groups of miners with interchanging roles. This work arrangement contrasted sharply with the atmosphere that surrounded other coal mines where miners were assigned to one-man-one-task roles in accordance with the principles of scientific management (Trist, 1981).

Trist and others at the Tavistock Institute were fascinated with what they had found. The autonomous work groups seemed to enjoy a more positive work environment. Moreover, they experienced higher productivity with lower absenteeism and fewer accidents than workers at conventional mines. What intrigued Trist and his colleagues was that management and labor had made a conscious choice to modify the work design to accommodate the shift in technology. This was interesting to the Tavistock researchers because the mining company might just have easily instituted the new technology without examining or changing the work practices.

Several IS researchers have embraced the sociotechnical approach, applying the concepts to the design and implementation of information systems (Bostrom & Heinen, 1977; Eason, 1988; Mumford, 1981; Pava, 1983). Sociotechnical design requires a high degree of participation from users, an idea that has been discussed heavily in the IS literature. Consistent with sociotechnical systems theory, we posit that there are two risk elements that must be simultaneously managed in order to achieve favorable project outcomes. We label these two elements *Social Subsystem Risk* and *Technical Subsystem Risk*.

Social Subsystem Risk captures the notion that a software project is embedded within a social context that may be unstable or highly politicized, causing reductions in commitment and resources needed to successfully complete the project. The social context in which the project is situated may also be characterized by users that are resistant to change, or not committed to the project. Thus, Social Subsystem Risk can be modeled as a construct consisting of two underlying risk dimensions: Organizational Environment and User.

Technical Subsystem Risk captures the notion that a software project involves the creation of a technical artifact of some complexity that is described by a set of requirements. The technical artifact may be more difficult to create when the requirements are unclear or highly volatile. The technical artifact can also be characterized by its level of complexity. The use of new technology or unfamiliar technology can have an effect on the overall complexity of the technical subsystem. Thus, Technical Subsystem Risk can be modeled as a construct represented by two underlying risk dimensions: Requirements and Project Complexity.

Prior literature has acknowledged a contingent relationship between the nature of the project and how it should be managed. McFarlan (1981), for example, has suggested that the selection and use of formal planning and control techniques, as well as internal and external integration tools, should be tailored to the type of project. Extrapolating this notion to the domain of software project risk, we posit a contingent relationship between the type of project (i.e., the social and technical subsystems within which the project is situated) and the level of project management risk. Thus, we state the following hypotheses:

H4: Social Subsystem Risk will have a significant positive impact on Project Management Risk.

H5: Technical Subsystem Risk will have a significant positive impact on Project Management Risk.

Moreover, since the technical subsystem is situated within a particular social context, we posit that the social subsystem can affect the technical subsystem. As noted earlier, the social subsystem consists of Organization and User Risk. If the organizational environment is unstable or the users are not cooperative, this will likely make it more difficult to produce an accurate and stable set of requirements, thus making it more challenging to create the technical artifact. Thus, we state the following hypothesis:

H6: Social Subsystem Risk will have a significant positive impact on Technical Subsystem Risk (i.e., more unstable environments will be associated with more complex and uncertain requirements, thus leading to greater risks associated with creation of the artifact itself).

Before testing the model and the above hypotheses, we turn to the process that was used to develop and validate measures of the modeled constructs.

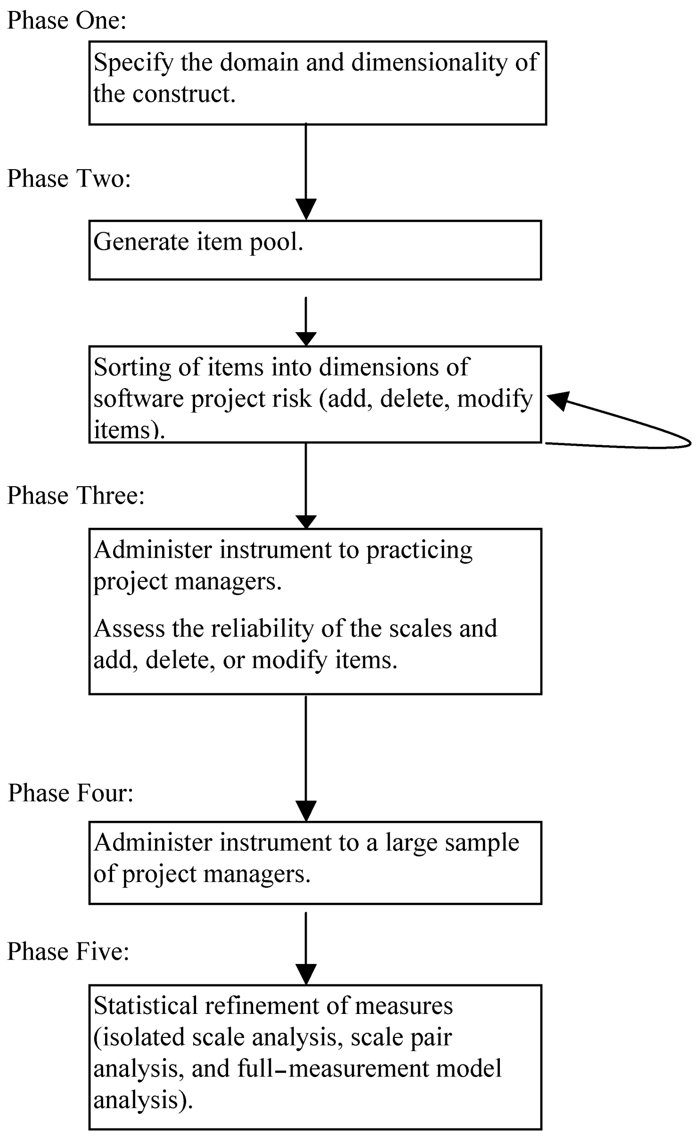
INSTRUMENT DEVELOPMENT AND VALIDATION

Figure 2 provides an overview of the methodology used to develop and validate an instrument to measure software project risk, and Appendix 1 provides more details about the phases (based on previous work by Churchill [1979] and Smith et al. [1996]). As seen in Phase One of Figure 2, the purpose of the initial phase was to specify the domain and dimensionality of the construct. After culling the six dimensions from the literature (as described in the literature review), we conducted interviews with practicing software project managers to determine if they were consistent with how practitioners regard the software project risk construct.

The focus of Phase Two was item creation and scale development. A pool of possible items was generated to measure software project risk. These items were subjected to various sorting techniques to validate the six underlying dimensions of software project risk that were proposed in Phase One and to refine the scale items intended to tap into the dimensions. In Phase Three, the items were pretested through the administration of a preliminary version of the instrument to a small sample of practicing project managers. The analysis of the pretest data was used to add, delete, or modify items. The survey instrument consisted of 53 items designed to tap into six different dimensions of software project risk. In Phase Four, the survey instrument was administered to a large sample ($n = 507$) of software project managers and tested for reliability and validity. The sample covered a wide variety of projects and varied from very small to very large projects, thereby strengthening the generalizability of the findings.

Phase Five focused on scale refinement using statistical analysis. Each scale model was examined in isolation, in pairs, and in a full-measurement model containing all indicators. The analysis was performed with AMOS 4.01 (Arbuckle,

Figure 2: Overview of methodology used to develop and validate the instrument.


Phase One:
Specify the domain and dimensionality of the construct.
Phase Two:
Generate item pool.
Sorting of items into dimensions of software project risk (add, delete, modify items).
Phase Three:
Administer instrument to practicing project managers.
Assess the reliability of the scales and add, delete, or modify items.
Phase Four:
Administer instrument to a large sample of project managers.
Phase Five:
Statistical refinement of measures (isolated scale analysis, scale pair analysis, and full-measurement model analysis).

1999). Maximum Likelihood Estimation (MLE) was selected as the most appropriate estimation procedure given the large sample size (Bentler & Bonett, 1980; Hatcher, 1994; Joreskog & Sorbom, 1989).

There has been some debate in the structural equation modeling community regarding whether a measurement model should be analyzed separately, before the analysis of a structural model. Recommendations from researchers in the area have ranged from a single process combining the analysis of the measurement and structural model (Hayduk, 1987; Hayduk & Glaser, 2000) to a two-step process that separates the measurement model and the structural model (Anderson & Gerbing, 1988; Herting & Costner, 2001), to a four-step process consisting of a factor model, a confirmatory factor model, the proposed structural model, and a more constrained model (James, Mulaik, & Brett, 1982; Mulaik & Millsap, 2000; Mulaik, 1998).

For this research we chose to separate the measurement model from the structural model. This approach is consistent with prior research (Carr, 2002; Chong & Chong, 2002; Frohlich, 2002; Rai, Lang, & Welker, 2002) and was appropriate here because it allowed us to refine our measures before testing the structural model. Following a suggestion by Bollen (2000), we applied some of the concepts behind his "jigsaw piecewise technique," in which pieces of the model are fitted individually and then together until the whole model is complete. We did this by adding a preliminary step where each latent variable was modeled separately to identify problems with model fit. This helped to identify psychometric issues so that our measures could be refined before a full confirmatory factor analysis was performed. Each of the steps we followed is discussed below.

Analysis of Isolated Models

An analysis of an isolated model of each dimension was performed to assess the ability of the set of items to tap into their associated dimension of risk. After assessing the fit of each model, steps were taken to identify and consider areas of possible model improvement. Specifically, items with loadings less than .60 were considered as candidates for deletion. All deletion decisions were assessed to ensure that content validity was preserved to the maximum extent possible. The scales were refined through repeated model fittings based upon an examination of the standardized loadings, modification indices, overall fit indices, interpretability, and content validity implications (Frohlich, 2002). Table A1-1 indicates the items that were deleted as a result of this analysis.

In the process of refining the instrument, we encountered some instances in which construct measures exhibited less than optimal psychometric properties because they tapped into multiple facets of a construct. In these cases, we found that the only way to improve the psychometric properties of our measures was to sharpen the focus of a construct. Accordingly, we decided to narrow the domain of coverage of the complexity and team constructs, so as to achieve reasonable measurement properties.

Initially, the measurement items pertaining to the complexity construct attempted to capture information concerning technological complexity, as well as other elements that might contribute to overall project complexity (e.g., complex requirements). However, the model analysis showed that only the four items dealing with technical complexity were loading strongly on the complexity construct. Thus, a decision was made to eliminate the other items, thereby focusing the measure of complexity around issues involving technical complexity.

Similarly, the initial measurement items pertaining to the team construct attempted to capture information concerning whether the team members had sufficient training and experience, as well as other elements that might contribute to team risk (e.g., conflicts or turnover within the team). The isolated model analysis revealed, however, that only the three items dealing with the expertise of the

Table 2: Final survey items.

Organizational Environment Risk	
Org1	Change in organizational management during the project
Org2	Corporate politics with negative effect on project
Org3	Unstable organizational environment
Org4	Organization undergoing restructuring during the project
User Risk	
User1	Users resistant to change
User2	Conflict between users
User3	Users with negative attitudes toward the project
User4	Users not committed to the project
User5	Lack of cooperation from users
Requirements Risk	
Req1	Continually changing system requirements
Req2	System requirements not adequately identified
Req3	Unclear system requirements
Req4	Incorrect system requirements
Project Complexity Risk	
Comp1	Project involved the use of new technology
Comp2	High level of technical complexity
Comp3	Immature technology
Comp4	Project involves use of technology that has not been used in prior projects
Planning & Control Risk	
P&C1	Lack of an effective project management methodology
P&C2	Project progress not monitored closely enough
P&C3	Inadequate estimation of required resources
P&C4	Poor project planning
P&C5	Project milestones not clearly defined
P&C6	Inexperienced project manager
P&C7	Ineffective communication
Team Risk	
Team1	Inadequately trained development team members
Team2	Inexperienced team members
Team3	Team members lack specialized skills required by the project

team were loading strongly onto the team construct. Thus, a decision was made to eliminate the other items, thereby focusing the measure of team risk around skill-related issues. Table 2 shows the items that were ultimately retained for each risk dimension.

Overall, the fit measures for the isolated models based on the refined scales indicated that the proposed models were a good fit with the observed data. The goodness of fit indices (GFIs) for all the models were .96 or above and the adjusted goodness of fit indices (AGFIs) were all .86 or above, which both indicate very good fit (Bollen, 1989). The lowest comparative fit index (CFI) and the lowest normed fit index (NFI) were .96 and .95 respectively, and the lowest nonnormed fit index (NNFI) was .91, which are all above their targets of .90 (Bentler, 1990; Bentler & Bonett, 1980). The highest standardized root mean square residual (RMR) was .05, which is an acceptable level (Bagozzi & Yi, 1988). All six of the calculated composite scale reliabilities were .78 or above; higher than the recommended minimum level of .70 (Fornell & Larcker, 1981). In general, the fit statistics suggested that the scales provided an adequate and reliable measure of fit for the six dimensions of software project risk.

In addition to measures for the six risk dimensions, the final instrument included five items designed to measure Product Performance and two items designed to measure Process Performance (as shown in Appendix 2). The measures for Product Performance and Process Performance were adapted from Rai and Al-Hindi (2000) and Nidumolu (1996) and the calculated composite reliabilities for both measures were above .80. A model of the two performance constructs and their items was also analyzed and the fit statistics were all well above the levels described above.

Discriminant and Convergent Validity

In order to assess the discriminant validity of the six risk scales, models of each pair of constructs were estimated. With six constructs of interest, there are fifteen possible combinations of construct pairs. Two models, a constrained model and an unconstrained model, were estimated for each possible pair of constructs. A significant value in an χ^2 difference test suggests that the unconstrained model is a better fit to the data than the constrained model (where the constructs are proposed to correlate perfectly) (Bagozzi & Phillips, 1982; Gerbing & Anderson, 1988). The χ^2 difference tests for every pair of constructs was significant at $p < .001$, indicating that each scale was measuring a construct that was significantly different from all other constructs, thus establishing discriminant validity.

To further establish discriminant validity, a confidence interval test was performed. A confidence interval of plus or minus 2 standard errors was calculated around the correlations between each pair of constructs. None of the confidence intervals included 1.0, thereby providing additional evidence of discriminant validity (Anderson & Gerbing, 1988; Hatcher, 1994).

The variance-extracted test can be used to establish both discriminant and convergent validity (Fornell & Larcker, 1981; Netemeyer, Johnston, & Burton, 1990). This test compares the variance-extracted estimates for two constructs of interest with the square of the correlation of the two constructs. Validity is demonstrated if both variance-extracted estimates are greater than this squared correlation (Hatcher, 1994). Table 3 shows the results of this test for the six dimensions. The diagonal elements in Table 3 are the variance-extracted estimates for each dimension, and the off-diagonal elements are the squared correlations between constructs. The diagonal elements are all larger than their corresponding correlation coefficients, which indicate that they exhibit both convergent and discriminant validity (Gefen, Straub, & Boudreau, 2000).

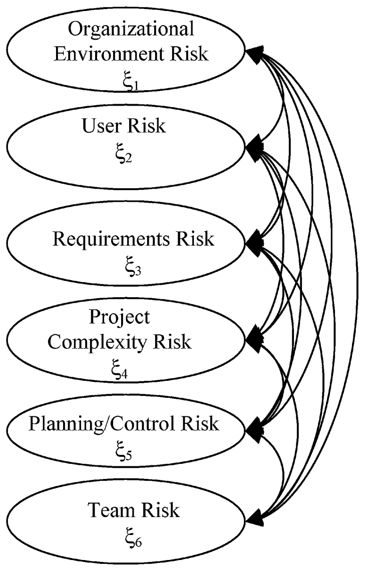
Adequacy of Model Fit

Confirmatory factor analysis (CFA) helps to test that indicator variables load highly on predetermined factors, but do not load highly on unrelated factors (Hatcher, 1994). AMOS 4.01 was used to perform a CFA to assess the measurement model

Table 3: Variance-extracted test.

	Organizational Environment	User	Requirements	Project Complexity	Planning and Control	Team
Organizational Environment	**0.478**					
User	0.189	**0.617**				
Requirements	0.143	0.175	**0.675**			
Project Complexity	0.006	0.007	0.061	**0.524**		
Planning & Control	0.125	0.102	0.419	0.036	**0.589**	
Team	0.072	0.059	0.203	0.070	0.283	**0.606**

Figure 3: Measurement model of software project risk.


Organizational Environment Risk ξ_1
User Risk ξ_2
Requirements Risk ξ_3
Project Complexity Risk ξ_4
Planning/Control Risk ξ_5
Team Risk ξ_6

$\chi 2$ (309)= 725.42 (p = 0.00)
Normed χ^2 = 2.3
GFI = .91
AGFI = .89
CFI = .94
NFI (Bentler/Bonnet) = .91
NNFI (Tucker-Lewis) = .94
RMR = .05
RMSEA = .05
Lower bound for RMSEA = .047
Upper bound for RMSEA = .056
Number of iterations = 9

Scale Reliabilities (composite reliabilities)

Org. Environ.	.78
User	.89
Requirements	.89
Complexity	.81
Planning/Control	.91
Team	.82

of software project risk (Arbuckle, 1999). The measurement model (see Figure 3) consisted of six latent variables: Organizational Environment risk (ξ_1), User risk (ξ_2), Requirements risk (ξ_3), Project Complexity risk (ξ_4), Planning/Control risk (ξ_5), and Team risk (ξ_6). The covariance matrix of the items comprising the six dimensions of risk was used as the input for the analysis and a reference variable for each latent construct was set to one in order to set the scale for the analysis. Each latent variable had between three and seven indicator variables (see Table 2). Each indicator variable represented a questionnaire item that was expected to load on only one factor.

The six-factor measurement model was estimated and Figure 3 contains the fit statistics and scale reliabilities that were obtained. The GFI, CFI, NFI, and NNFI are all higher than the recommended minimum values of .90 (Bentler, 1990; Bentler

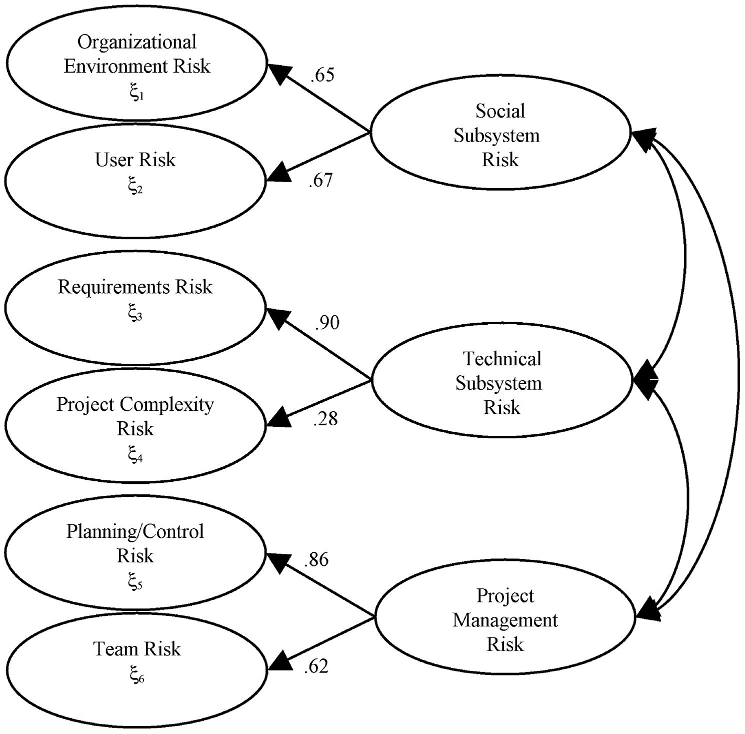
& Bonett, 1980; Hair, Anderson, Tatham, & Black, 1998). The AGFI is also well above the recommended minimum value of .80, suggesting good model fit. The standardized RMR is .05, which was judged to be an acceptable value (Bagozzi & Yi, 1988). The root mean square error of approximation (RMSEA) is well under the threshold of .08, indicating good model fit after adjusting for the large sample size (Browne & Cudeck, 1993).

The standardized factor loadings were all significant, with values ranging from .62 to .91. The t-values, representing the significance of the loadings of the indicators on the constructs, indicated that all paths were significant at the $p < .001$ level. This provides evidence to support the convergent validity and unidimensionality of the indicators (Anderson & Gerbing, 1988). Furthermore, the loadings between the indicators and their underlying constructs were almost identical to the loadings that were obtained during the paired comparisons, thereby indicating a high degree of stability of the estimates. Finally, the composite reliability index (see Fornell & Larcker, 1981) of each scale was also acceptable with all values in excess of .78 (Nunnally & Bernstein, 1994). Overall, the fit measures and item loadings we obtained were found to meet accepted thresholds, thus supporting the reliability and validity of the refined instrument (Hair, Anderson, Tatham, & Black, 1992; Henry & Stone, 1994).

The measurement model is analogous to a first-order model, where the six dimensions of risk are allowed to covary. This first-order model can serve as a baseline model against which a theoretical model, such as a second-order factor model, can be tested. A chi-square difference test can be used to determine whether there is a significant difference between the fit provided by the measurement model and the fit provided by a theoretical model. If the theoretical model is successful in accounting for the observed relationships between the variables in the measurement model then there will not be a significant difference between the chi-square for the theoretical model and the chi-square for the measurement model (Anderson & Gerbing, 1988; Hatcher, 1994). The second-order factor (theoretical) model in this case includes the three higher-level latent constructs—social subsystem risk, technical subsystem risk, and project management risk—that are proposed to govern the relationship among the six first-order constructs. The second-order factor model is shown in Figure 4.

As shown in Figure 4, the six dimensions of risk are not allowed to correlate, but rather their covariation is explained by the second-order constructs, making it a more parsimonious model (i.e., fewer paths) than the measurement model. With the exception of the chi-square statistic (χ^2 (315) = 737.30), the measures of fit from the second-order model were identical to the first-order model (see Figure 3 for the fit statistics), and the loadings of the variables onto their respective constructs were the same, thereby demonstrating the stability of the second-order model. To statistically compare the two models, a chi-square difference test was performed, resulting in a chi-square difference value of 11.88 with 6 degrees of freedom. With 6 degrees of freedom, the critical value of the chi-square is 22.458 at $p < .001$. The chi-square difference value of 11.88 is less than the critical value, meaning that there is no significant deterioration in the fit of the theoretical model as compared to the measurement model, given the additional constraints that were imposed.

Figure 4: Second-order model of software project risk.

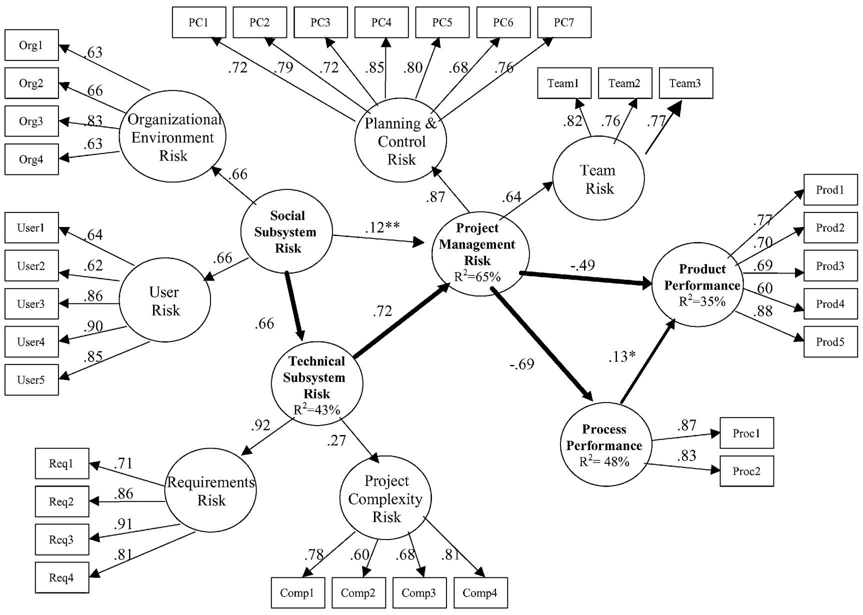

Organizational Environment Risk ξ_1
User Risk ξ_2
Requirements Risk ξ_3
Project Complexity Risk ξ_4
Planning/Control Risk ξ_5
Team Risk ξ_6
Social Subsystem Risk
Technical Subsystem Risk
Project Management Risk
.65
.67
.90
.28
.86
.62

It has been suggested that if two models have similar fit, the more parsimonious model should be tentatively selected as the superior model (Anderson & Gerbing, 1988; Hatcher, 1994). However, the choice of a model based solely on parsimony considerations can lead to the selection of an incorrect model (Marsh & Hau, 1996; Raykov & Marcoulides, 1999), so other considerations must accompany the selection of the model in a situation where multiple models produce similar fit. Our preference of the theoretical model shown in Figure 4 over the measurement model is based on the empirical confirmation of the expected theoretical relationships between the first- and second-order constructs, which were derived from sociotechnical systems theory. The theoretical model is more parsimonious than the measurement model; it does not show any significant deterioration of fit when compared to the measurement model, and it is grounded in an established theory. Furthermore, other second-order factor models were tested using different combinations of the first-order factors, and all of them showed significant deterioration of fit when compared to the measurement model, thereby providing additional validation of the theoretical model shown in Figure 4.

TESTING THE MODEL OF RISK AND PERFORMANCE

Structural equation modeling using AMOS 4.01 was selected to evaluate the relationships between indicators and latent constructs, as well as the structural

Figure 5: SEM analysis results.



All paths were significant at p < .001 unless noted.
*denotes that the path was significant at p < .05.
**denotes an insignificant path.

relationships between the second-order latent constructs. Figure 5 shows the standardized loadings of the items on each latent construct, as well as the path loadings between constructs. The paths for all six risk dimension constructs loading on their respective latent constructs (technical subsystem risk, project management risk, and social subsystem risk) were significant at $p < .001$. The path loadings for the six risk dimension constructs are very similar to the results of the second-order factor model (Figure 4), thereby further supporting the stability of the model.

All but one of the path coefficients associated with the relationships between the latent constructs were significant at $p < .001$. The only insignificant path ($p > .05$) was between Social Subsystem Risk and Project Management Risk. The percent of variance explained by the model as it relates to Technical Subsystem Risk, Project Management Risk, Process Performance, and Product Performance were 43%, 65%, 48%, and 35%, respectively. These values are relatively high, thus providing confidence that the proposed model has a fairly high degree of explanatory power.

Overall, the fit statistics for the structural model were judged to be acceptable, with GFI, CFI, NFI and NNFI values of .89, .94, .89 and .93 respectively. The AGFI was .87 and the normed χ^2 was 2.15. The RMSEA was .048 (with a lower bound of .04 and upper bound of .05) and the standardized RMR was .05. Several alternative models were run for comparison to ensure that the proposed model was the most satisfactory explanation for the relationship between the constructs of interest. Models where either Technical Subsystem Risk or Social Subsystem Risk were linked directly to the performance variables resulted in insignificant paths between those variables and the performance constructs. All of the alternative models performed more poorly than the proposed model, thereby substantially supporting our research model.

DISCUSSION AND IMPLICATIONS

Our results suggest that Social Subsystem Risk increases Technical Subsystem Risk, as determined by requirements and technical complexity. Operating projects within unstable organizational environments or with resistant users increases Social Subsystem risk, which, in turn, can increase Technical Subsystem risk. Increased Technical Subsystem Risk, as determined by such enhanced requirements as uncertainty and technological complexity, has a dramatic impact on Project Management risk. Managerial processes, such as planning and control mechanisms, and assembling a highly skilled project team can be used as interventions to mediate the potentially detrimental effects of Technical Subsystem Risks on project performance. The importance of such mediation becomes critical for projects in risky organizational environments or during the development of artifacts with significant requirements volatility.

In general, our results support the view that a sociotechnical perspective provides a parsimonious approach to conceptualize software project risk and understand interrelationships between the six risk dimensions that were identified. Reasonably high levels of explained variance with relatively few constructs suggest that the theoretical framing is focused on key constructs and their interrelationships. In addition, the logical appeal of the theoretical framing is compelling in that it recognizes that systems are developed in a specific social context and that the contextual setting impacts the riskiness of the project. From a managerial standpoint, such risks can be proactively managed by implementing processes and structures that are designed to counter the risks associated with the organizational environment, users, requirements, and project complexity. For example, training the project team on methods that will increase user involvement may contribute to clearer and more stable requirements, thereby leading to better project outcomes.

Limitations

Before discussing the implications of our research further, it is important to discuss potential limitations. Risk is a complex construct and this research may not have captured every aspect of software project risk, even though a rigorous method was applied to ensure that the specifications of the construct domain were as complete as possible and their measures had acceptable measurement properties. The project complexity scale was refined to focus on technical complexity. Similarly, the team risk scale was refined to focus on skills. Other facets of project complexity and team risk should be investigated in future research.

Our study focused on the internal risks associated with software development projects. External shocks to an organization can also represent significant risks for software projects. While our organizational environment construct may capture some of the effects associated with changes in the external environment, it does so only in an indirect manner. It is also possible that different types of projects may carry different kinds of risk. For example, outsourced projects may exhibit a different risk profile as compared with in-house development projects and some additional risk constructs (e.g., external relationship quality) may need to be incorporated to fully gauge the risk of these projects.

Finally, since convenience samples were used throughout this study and the scales themselves were geared toward internal risks, the external validity and generalizability of this study may be limited. The findings of this study should be tested with other groups of software project managers as well as with different types of projects.

Implications for Research

Previous literature has stressed the importance of software project risk assessment and control. To date, however, there has been no widely accepted measure or comprehensive definition of the factors associated with software project risk. Previous studies have proposed that risk is a complex construct, comprised of many components. They have suggested that software project risk is a multidimensional construct that cannot be captured by a single scale. But surprisingly, only one other study (Barki et al., 1993) has attempted to identify the dimensions of software project risk and undertake an empirical assessment of the identified dimensions. At the beginning of this paper we noted some limitations associated with the Barki et al. instrument, including some statistical issues and the omission of practitioners in the identification and validation of risk items or the factors that emerged from their analysis.

Our study addressed these limitations by employing a rigorous methodology for the development of a grounded instrument that taps into the major dimensions of software project risk, as reflected in both the literature and in the eyes of practitioners. Interestingly, the dimensions identified differ somewhat from the factors identified by Barki et al. (1993). Specifically, our parsing of the risk construct emphasizes the importance of requirements risk, planning and control risk, and user risk to a much greater degree than is captured by Barki et al. Conversely, Barki et al. place considerable emphasis on application size, which was originally included in our project complexity construct, but subsequently dropped in the process of refining the complexity scale to focus on technical complexity.

One contribution of this study is that it provides a definition of the dimensions of risk and a validated set of measures for their operationalization. The result is an understanding of risk as a function of six dimensions. This study used different levels of model testing to validate the measures of the six dimensions which, taken together, represent software project risk. Furthermore, the rigorous evaluation of the measures suggests that the six dimensions are distinct aspects of the underlying construct, software project risk. The measures developed in this research represent reliable and valid indicators of each of the six dimensions.

A second contribution of this study is that we have articulated a model that builds on the six dimensions of risk and explains a significant amount of the variance in project performance. This model, which is grounded in the project management literature and sociotechnical systems theory, along with the underlying risk measures we have developed, provides a solid foundation for future work in the area.

One potential area of research involves using the instrument developed here to compare how different stakeholders perceive risk. In the current study there was only one participant from each project who completed the risk assessment, but it is reasonable to believe that different project participants would view risk differently and that the differences in their assessments may provide insight on why many software projects go awry. Another avenue for future research would be to use the constructs and measures described here to study how risk perceptions change during the course of a project. To relate risk to project performance, the current study was limited to retrospectively assessing risk at the conclusion of the project. Taking measures of risk throughout a project and monitoring variations in the dimension levels would help to produce an understanding of how the risk profile of a project typically changes over time. The measures of software project risk developed in this study can also be used to learn more about the effectiveness of various risk mitigation tactics designed to reduce the severity of a risk factor's impact and to increase the likelihood of successful software development.

Additionally, this study did not take into account risk magnitude. Risk is often defined as the product of the uncertainty surrounding a project times the magnitude of the potential loss associated with project failure. This research focused only on the uncertainty, or risk factors, and did not address any additional impact caused by the size of possible losses due to failure. Future research could investigate whether magnitude issues play into perceptions of risk. For example, someone working on a technically complex project with a high potential for loss if a problem occurs (such as nuclear control software system) might have a much higher perception of risk than someone working on an equally complex system that would not result in large losses.

Implications for Practice

This study has described six dimensions of software project risk that project managers may use for identifying and managing the risks associated with software development projects. The measures developed in this research can be used to provide an early warning of potential project problems so that corrective action can be taken to avoid project failure.

Practitioners can use the instrument to develop historical databases of the risks associated with different projects and their outcome. Compilation of this information could provide a means of assessing future projects. The constructs and measures developed here could then be used to create a risk profile for each project. These profiles could then be compared to historical norms. Potentially high-risk projects could be flagged at an early stage so that appropriate decisions could be made about whether or not to continue with a high-risk project, or to select an alternative course of action. Practitioners could also administer the instrument at multiple points during a project and track the changes in the riskiness of a project as it progresses from beginning to end. In keeping with good project management practices, this would help to ensure that risk assessment is an ongoing process and not something that happens once at the outset of a project.

The design of an effective project risk management strategy must recognize the relationships between risks associated with the social subsystem, technical subsystem, and project management, and the implications of these relationships for project performance. Projects situated in contexts with high social subsystem risk and high technical subsystem risk should aggressively use project planning and control approaches and involve a highly skilled team. Such project management interventions provide for an essential adaptive capability required to achieve high levels of project performance in such contexts.

The constructs and measures developed here could also be used to monitor the effects of risk mitigation strategies used to counteract risks that are identified. Perhaps some mitigation strategies would be more effective than others when certain types of risk were assessed to be unusually high. For example, if organizational environment risk is high, there may be certain actions that a project manager could take to reduce the impact on the uncertainty of the project requirements (i.e., by keeping the system simple and on a short development time-frame). Over time, this information could be used to develop an understanding of the effectiveness of various risk mitigation strategies, possibly leading to a contingency framework for managing project risk. Such knowledge would be a very valuable resource for identifying the appropriate risk management tools given a particular risk profile. [Received: December 2002. Accepted: November 2003.]

REFERENCES

Abdel-Hamid, T. K. (1989). A study of staff turnover, acquisition, and assimilation and their impact on software development cost and schedule. *Journal of Management Information Systems, 6*(1), 21–39.

Alter, S., & Ginzberg, M. (1978). Managing uncertainty in MIS implementation. *Sloan Management Review, 20*(1), 23–31.

Anderson, J. C., & Gerbing, D. W. (1988). Structural equation modeling in practice: A review and recommended two-step approach. *Psychological Bulletin, 103,* 411–423.

Arbuckle, J. L. (1999). *AMOS for Windows: Analysis of moment structures*. Chicago: Small Waters.

Arrow, K. (1970). *Essays in the theory of risk-bearing*. Amsterdam: North-Holland.

Bagozzi, R. P., & Phillips, L. W. (1982). Representing and testing organizational theories: A holistic construal. *Administrative Science Quarterly, 27,* 459–489.

Bagozzi, R. P., & Yi, Y. (1988). On the evaluation of structural equation models, *Journal of the Academy of Marketing Science, 16,* 74–94.

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

Bentler, P. M. (1990). Comparative fix indexes in structural models. *Psychological Bulletin, 107*(2), 238–246.

Bentler, P. M., & Bonett, D. G. (1980). Significance tests and goodness of fit in the analysis of covariance structures. *Psychological Bulletin, 88,* 588–606.

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

Bollen, K. A. (1989). *Structural equations with latent variables*. New York: Wiley.

Bollen, K. A. (2000). Modeling strategies: In search of the holy grail. *Structural Equation Modeling, 7*(1), 74–81.

Bostrom, R. P., & Heinen, J. S. (1977). MIS problems and failures: A socio-technical perspective—Part I: The causes. *MIS Quarterly, 1*(3), 17–32.

Brooks, F. P. (1987). No silver bullet: Essence and accidents of software engineering. *Computer* (April), 10–19.

Browne, M. W., & Cudeck, R. (1993). Alternative ways of assessing model fit. In K. A. B. J. S. Long (Ed.), *Testing structural equation models*. Newbury Park, CA: Sage, 136–162.

Carr, C. L. (2002). A psychometric evaluation of the expectations, perceptions, and difference-scores generated by the IS-adapted SERVQUAL instrument. *Decision Sciences, 33*(2), 281–296.

Casher, J. D. (1984). How to control risk and effectively reduce the chance of failure. *Management Review, 73*(6), 50–54.

Charette, R. N. (1989). *Software engineering risk analysis and management*. New York: Intertext.

Chong, V. K., & Chong, K. M. (2002). Budget goal commitment and informational effects of budget participation on performance: A structural equation modeling approach. *Behavioral Research in Accounting, 14,* 65–86.

Churchill, G. A., Jr. (1979). A paradigm for developing better measures of marketing constructs. *Journal of Marketing Research, 16*(1), 64–73.

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly, 13*(3), 318–340.

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

Eason, K. (1988). *Information technology and organizational change*. London: Taylor & Francis.

Ewusi-Mensah, K., & Przasnyski, Z. H. (1991). On information systems project abandonment: an exploratory study of organizational practices. *MIS Quarterly, 15*(1), 67–85.

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

Frohlich, M. T. (2002). E-Integration in the supply chain: Barriers and performance. *Decision Sciences, 33*(4), 537–556.

Gefen, D., Straub, D., & Boudreau, M. (2000). Structural equation modeling and regression: guidelines for research practice. *Communications of the Association of Information Systems, 4*(7), 1–79.

Gerbing, D. W., & Anderson, J. C. (1988). An updated paradigm for scale development incorporating unidimensionality and its assessment. *Journal of Marketing Research, 25,* 186–192.

Gordon, P. (1999, January 18). To err is human, to estimate, divine. *Information Week,* 65–72.

Hair, J. F., Anderson, R. E., Tatham, R., & Black, W. C. (1992). *Multivariate data analysis with readings*. New York: Macmillan.

Hair, J. F., Anderson, R. E., Tatham, R., & Black, W. C. (1998). *Multivariate data analysis with readings*. Englewood Cliffs, NJ: Prentice Hall.

Hatcher, L. (1994). *A step-by-step approach to using the SAS system for factor analysis and structural equation modeling*. Cary, NC: SAS Institute.

Hayduk, L. A. (1987). *Structural equation modeling with LISREL: Essentials and advances*. Baltimore, MD: John Hopkins University Press.

Hayduk, L., & Glaser, D. N. (2000). Jiving the four-step, waltzing around factor analysis, and other serious fun. *Structural Equation Modeling, 7*(1), 1–35.

Heemstra, F. J., & Kusters, R. J. (1996). Dealing with risk: A practical approach. *Journal of Information Technology, 11*(4), 333–346.

Henry, J. W., & Stone, R. W. (1994). A structural equation modeling of end-user satisfaction with a computer-based medical information system. *Information Resources Management Journal, 7*(3), 21–33.

Herting, J. R., & Costner, H. L. (2001). Another perspective on the proper number of factors and the appropriate number of steps. *Structural Equation Modeling, 7*(1), 92–110.

James, L. R., Mulaik, S. A., & Brett, J. M. (1982). *Causal analysis: Assumptions, models, and data*. Beverly Hills, CA: Sage.

Jarvenpaa, S. L., & Ives, B. (1991). Executive involvement and participation in the management of information technology. *MIS Quarterly*, *15*(2) 205–277.

Jiang, J. J., & Klein, G. (2001). Information system success as impacted by risks and development strategies. *IEEE Transactions on Engineering Management, 48*(1), 46–55.

Jiang, J. J., Klein, G., & Means, T. (2000). Project risk impact on software development team performance. *Project Management Journal* (December), 19–26.

Johnson, J. (1999). Turning chaos into success. *Software Magazine, 19*(3), 30.

Jones, C. (1994). *Assessment and control of software risks*. Englewood Cliffs, NJ: Yourdon.

Jones, M. M., & McLean, E. R. (1970). Management problems in large-scale software development projects. *Industrial Management Review, 11*(3), 1–15.

Joreskog, K. G., & Sorbom, D. (1989). *LISREL 7: A Guide to the Program and Applications*. Chicago: SPSS.

Karolak, D. W. (1996). *Software engineering risk management*. Los Alamitos, CA: IEEE Computer Society Press.

Keider, S. P. (1984). Why systems development projects fail. *Journal of Information Systems Management, 1*(3), 33–38.

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

Kemerer, C. F., & Sosa, G. L. (1988). Barriers to successful strategic information systems. *Planning Review, 16*(5), 20–23, 46.

MacCrimmon, K. R., & Wehrung, D. A. (1984). The risk in-basket. *Journal of Business, 57*(3), 367–387.

March, J. G., & Shapira, Z. (1987). Managerial perspectives on risk and risk taking. *Management Science, 33*(11), 1404–1418.

Marsh, H. W., & Hau, K. T. (1996). Assessing goodness of fit: Is parsimony always desirable? *Journal of Experimental Education, 64*(4), 364–390.

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

Metzger, P. W. (1981). *Managing a programming project*. Englewood Cliffs, NJ: Prentice Hall.

Moore, G. C., & Benbasat, I. (1991). Development of an instrument to measure the perceptions of adopting an information technology innovation. *Information Systems Research, 2*(3), 192–222.

Moore, J. H. (1979). A framework for MIS software development projects. *MIS Quarterly, 3*(1), 29–38.

Moynihan, T. (1997). How experienced project managers assess risk. *IEEE Software, 14*(3), 35–41.

Mulaik, S. A., & Millsap, R. E. (2000). Doing the four-step right. *Structural Equation Modeling, 7*(1), 36–73.

Mulaik, S. A. (1998, November 20). Four-step model. *SEMNET Discussion List*.

Mumford, E. (1981). Participative systems design: Structure and methods. *Systems Objectives Solutions, 1*(1), 5–19.

Netemeyer, R. G., Johnston, M. W., & Burton, S. (1990). Analysis of role conflict and role ambiguity in a structural equations framework. *Journal of Applied Psychology, 75,* 148–157.

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.

Nunnally, J. C., & Bernstein, I. H. (1994). *Psychometric theory*. New York: McGraw-Hill.

O'Toole, R. J. W., & O'Toole, E. F. (1966). Top executive involvement in the EDP function. *PMM & Co-Management Controls* (June), 125–127.

Pava, C. H. P. (1983). *Managing new office technology*. New York: Free Press.

Project Management Institute. (2000). *A Guide to the project management body of knowledge*. Newtown Square, PA: Project Management Institute.

Rai, A., & Al-Hindi, H. (2000). The effects of development process modeling and task uncertainty on development quality performance. *Information and Management, 37,* 335–346.

Rai, A., Lang, S. S., & Welker, R. B. (2002). Assessing the validity of IS success models: An empirical test and theoretical analysis. *Information Systems Research, 13*(1), 50–69.

Raykov, T., & Marcoulides, G. (1999). On the desirability of parsimony in structural equation model selection. *Structural Equation Modeling, 6,* 292–300.

Robey, D., & Farrow, D. L. (1982). User involvement in information systems development: A conflict model and empirical test. *Management Science, 28*(1), 73–85.

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.

Smith, H. J., Milberg, S. J., & Burke, S. J. (1996). Information privacy: Measuring individuals' concerns about organizational practices. *MIS Quarterly, 20*(2), 167–196.

Tait, P., & Vessey, I. (1988). The effect of user involvement on system success: A contingency approach. *MIS Quarterly* (March), 91–110.

Thayer, R. H., Pyster, A., & Wood, R. C. (1980). The challenge of software engineering project management. *IEEE Computer* (August), 51–59.

Trist, E. (1981). The sociotechnical perspective. In A. H. V. d. Ven & W. F. Joyce (Ed.), *Perspectives on organizational design and behavior*. New York: Wiley, 19–75.

APPENDIX 1

Instrument Validation Process

Phase one: Specifying the domain and dimensionality of the construct

Because the six dimensions described in the main body of the paper evolved from an iterative and interpretive reading of the literature, steps were taken to verify the completeness of the listing with a panel of experts. Specifically, interviews were conducted with four individuals with software project management experience and four currently practicing software project managers. The experts were asked to list factors that appeared to cause problems in their previous software development projects. The purpose of this exercise was to capture any nuances of the dimensions that may have been overlooked during the literature review, but are considered to increase the riskiness of a project by experts in the area. The feedback from the experts was used to modify the descriptions of the six dimensions.

The experts were then asked to review descriptions of the six dimensions and comment on the how adequately the dimensions captured the domain of software project risk. During the interview process many of the experts shared examples of how a particular risk factor had specifically impacted one of their projects. This information was used during the item generation portion of Phase Two of the research.

Phase two: Generating the item pool and sorting them into dimensions

The second phase of instrument development involved further elaboration and verification of the underlying dimensions of software project risk. The primary focus of this phase was to begin formally converting the construct dimensions into measurable scales. The steps and techniques chosen for Phase Two were similar to the procedures followed by Davis (1989), Moore and Benbasat (1991), and Smith (1996).

In preparation for this phase, short descriptive statements were written to describe as many aspects of the six dimensions identified in Phase One as possible. There were 76 statements (see Table A1.1 for the original list of items, as well as a trace of all the modifications made to the items in later phases of the research) written to cover the content of the domain of software project risk.

The 76 statements were then used in a preliminary sorting exercise to validate the construct validity of the six dimensions of software project risk discussed earlier. The process used was very similar to that used by Moore and Benbasat (1991) to validate a set of proposed dimensions. Specifically, the 76 items were each printed on 3 × 5-inch index cards and randomly shuffled. Then 150 graduate students (majoring in computer information systems) at a large southeastern university were given a set of the 76 cards and asked to individually sort through them and group similar risks together into piles. The subjects were also given blank index cards and asked to write a description, or label, for each pile that they created. The subjects were given no upper or lower limits as to the number of piles or dimensions they could create. Moore and Benbasat (1991) had only four individuals participate in their preliminary sort. By including 150 subjects in the current study it was hoped to gain an even greater confidence about the identified dimensions and the items' ability to measure them.

The respondents created anywhere from 3 to 13 piles based on their interpretation of how the items could best be grouped together. The average number of piles created by the subjects was approximately 6, thereby lending some initial support to the number of dimensions identified in Phase One of the research. The subject's piles were then analyzed in more detail in order to identify how frequently the items intended to measure each dimension were grouped together. A 100-subject threshold was used as the guideline because it indicated that two-thirds or more of the subjects believed the items to share something in common. This exercise helped to verify that the items were being grouped in a manner that was consistent with the proposed dimensions, thereby indicating a high level of construct validity.

Two members of the research team analyzed the data to verify the clustering of the items into the six dimensions and to determine how well the items were

Table A1.1: Summary of item changes.

	Original Item	S1	S2	S3	P	I	Scale	Final Item
1	Frequent user turnover				D			
2	Inadequate development tools selected for the project	M		D				
3	Project with highly complex requirements			D				
4	Lack of cooperation from users						User5	Lack of cooperation from users
5	Frequent conflicts between project team members	M				D		
6	Inadequate support from supplier(s)	D						
7	Use of inadequate development methodology	D						
8	Project requires a change in currently utilized technology		M				Comp4	Project involves use of technology that has not been used in prior projects
9	Responsibilities for project task assignments not clearly defined	D						
10	Project affects a large number of departments or organizational units		M		M	D		
11	Team members continually added to the project after it began	D						
12	Low level of staff motivation	D						
13	Team members not familiar with the type of application being developed	D						
14	Project progress not monitored closely enough						P&C2	Project progress not monitored closely enough
15	System intended to service a large number of users	D						
16	Large number of links to other systems required by the project		M			D		
17	Lack of top management support for the project					D		
18	Lack of a full-time project manager				D			
19	Project milestones not clearly defined						P&C5	Project milestones not clearly defined
20	Users resistant to change						User1	Users resistant to change
21	Inexperienced project manager						P&C6	Inexperienced project manager
22	Change in organizational management during the project						Org1	Change in organizational management during the project

Table A1.1: (continued) Summary of item changes.

	Original Item	S1	S2	S3	P	I	Scale	Final Item
23	Project staff not familiar with technology required for the project	D						
24	Project with high level of technical complexity			M			Comp2	High level of technical complexity
25	Incorrect system requirements						Req4	Incorrect system requirements
26	One of the largest projects attempted by the organization					D		
27	Inadequately trained project team members	M					Team1	Inadequately trained development team members
28	Lack of effective project management methodology			M			P&C1	Lack of an effective project management methodology
29	Project involves the use of new technology			M			Comp1	Project involved the use of new technology
30	Project goals not agreed upon		M		D			
31	Unrealistic user expectations for the system under development		M		M	D		
32	Failure to gain user commitment		M				User4	Users not committed to the project
33	Frequent team member turnover		M		M	D		
34	Team members lack specialized skills required by the project						Team3	Team members lack specialized skills required by the project
35	Insufficient resources provided for the project	M			D			
36	Undefined project success criteria					D		
37	Use of new development methodology	D						
38	Many external suppliers involved in the development project				D			
39	Organization undergoing layoffs or downsizing during the project				M		Org4	Organization undergoing restructuring during the project
40	Conflicting requirements provided by users		M		M	D		
41	Conflict between user departments		M				User2	Conflict between users
42	Project goals are not identified		M			D		
43	Goldplating (i.e., the addition of unneeded features to the system specifications)	D						
44	Incorrect assumptions made by project team about user needs	D						

Table A1.1: (continued) Summary of item changes.

	Original Item	S1	S2	S3	P	I	Scale	Final Item
45	Poor project planning						P&C4	Poor project planning
46	Users not involved in development process		M		M	D		
47	Inexperienced team members						Team2	Inexperienced team members
48	Lack of team commitment to the project		M			D		
49	Unstable organizational environment						Org3	Unstable organizational environment
50	Inappropriate technology selected for the project	M		M	M		Comp3	Immature technology
51	Continually changing scope/objectives		M			D		
52	Project involving the automation of unstructured task(s)				M	D		
53	Team members not familiar with the task(s) being automated					D		
54	System will result in a large number of changes to the organization	D						
55	Negative attitudes by project staff	M				D		
56	Lack of "people skills" in project leadership					D		
57	Lack or loss of organizational commitment to the project					D		
58	Unclear system requirements						Req3	Unclear system requirements
59	Shortfalls in externally furnished components	D						
60	No change management practices in place	D						
61	Users not able to identify system requirements		M				Req2	System requirements not adequately identified
62	Resources shifted away from the project because of changes in organizational priorities					D		
63	Unrealistic budget	M				D		
64	Continually changing system requirements						Req1	Continually changing system requirements
65	Large number of people working on the project	D						
66	Highly complex task being automated					D		
67	Inexperienced users				D			

Table A1.1: (continued) Summary of item changes.

	Original Item	S1	S2	S3	P	I	Scale	Final Item
68	Poor risk management practices	D						
69	Poor reward structure for team member performance	D						
70	Organization involved in a merger during the project				D			
71	Ineffective project team communication				M		P&C7	Ineffective communication
72	Inadequate estimation of project budget, schedule and required resources	M					P&C3	Inadequate estimation of required resources
73	Ineffective project manager					D		
74	Excessive schedule pressure	M				D		
75	Corporate politics with negative effect on project						Org2	Corporate politics with negative effect on project
76	Users with negative attitudes toward the project						User3	Users with negative attitudes toward the project
New1	Use of relatively new development tools that have not been widely used before	A	M	M		D		
New2	Complicated contractual arrangements with outside suppliers	A	M			D		

Column Headings:
S1 = Changes made after first sorting exercise
S2 = Changes made after second sorting exercise
S3 = Changes made after third sorting exercise
P = Changes made after the pretest
I = Changes made after the isolated model analysis

Scale = Item to measure one of the following scales:
User = User
Comp = Project Complexity
Team = Development Team
Org = Organizational Environment
P&C = Planning/Control
Req = Requirements

Legend:
A = New item added
M = Item modified
D = Item deleted

grouping together into the six dimensions proposed in Phase One. Table A1.1 identifies the items modified in this first sorting exercise.

The next part of Phase Two involved a different type of sorting exercise to further refine the item measures. A sorting instrument was developed that contained a short paragraph description of each of the hypothesized dimensions of software project risk as well as a random listing of the 61 items remaining from the earlier sort.

Three IS faculty, three PhD students and two practicing project managers were asked to evaluate and critique the dimension descriptions to ensure that they were accurate representations of each dimension. Changes were made to the definitions based on the feedback from these eight individuals. Then, the sorting instrument was administered to a convenience sample of 46 practicing project managers. The respondents were asked to assign each of the 61 remaining items to the dimension that they felt was best represented by that item.

The results were first analyzed to identify those items that were assigned to the same category by at least 32 of the 46 respondents (70%). This frequency measure revealed 30 items that were placed into the same category by at least 70% of the subjects. These 30 items were retained without any modifications for the next phase of the research.

However, there were 31 items that were not placed into the same category by at least 70% of the subjects. Modifications to the 31 items were considered. Fifteen of the 31 items were not modified either because it was not clear how best to reword them or because, although they were not placed into the same category by 70% of the respondents, they were placed into a category by a relatively high percentage of the respondents (i.e., 60%). The remaining sixteen items were reworded in an attempt to improve their classification rate and to provide a better fit with the other items in their measurement scale. No items were deleted as a result of this analysis. Table A1.1 identifies the items that were altered after this second sort.

After the changes described above were made, the sorting exercise was repeated to validate the modifications. The subjects who performed the sort were a convenience sample of 20 PhD students majoring in IS. The responses to this exercise were analyzed in a similar manner as the previous exercise. Because item refinement had taken place, the results were much improved. This time, 53 of the 61 items were consistently placed into the same category by at least 70% of the respondents. Of the remaining eight items, two items were deleted, five items were reworded, and one remained unmodified. The output of Phase Two was a comprehensive list of 59 risk items designed to measure the six dimensions of the software project risk construct (see Table A1.1).

Phase three: Pre-testing the instrument and assessing its reliability

The goal of Phase Three was to produce an instrument containing a set of reliable and internally consistent scales to measure each of the dimensions of software project risk and to further validate the measure of software project risk. A pretest was performed to obtain a general assessment of the instruments' appearance, to further eliminate items that did not contribute significantly to the value of the instrument, and to initially assess the reliability of the measurement scales. The 59 remaining items from Phase Two were placed on an instrument and the respondents were asked to read each statement and indicate the extent to which it characterized their most recently completed project. The response format for each item was a seven-point Likert-type scale ranging from "strongly disagree" to "strongly agree."

The pretest was administered to a convenience sample of 43 software project managers. The survey and a cover letter were distributed to 90 participants of a software project management conference along with their registration packets. Fourteen of these individuals completed the survey and returned them to the registration desk. The remainder of the sample of 43 in this phase came from personal contacts as well as surveys that were distributed to graduate IS students and completed by software project managers in the organizations where they were employed.

The pretest analysis consisted of calculating coefficient alpha and item–total correlations for each scale. The coefficient alphas and item–total correlations were used to identify problematic items that were either reworded or deleted from the scale, resulting in a set of six internally consistent scales. Table A1.1 identifies the items that were deleted or modified as a result of the pretest analysis.

Phase four: Large-scale administration of the instrument

The purpose of the fourth phase of the instrument development process was to further examine the reliability and validity of the instrument, which now consisted of 53 items, and to test a measurement model of software project risk. The final instrument was administered via a web-based survey. As with the pretest, the subjects were asked to complete the survey as it related to their most recently completed project. The 53 items were randomly placed onto the final survey and respondents were again asked to indicate on a seven-point Likert-type scale the extent to which each statement characterized their most recently completed project.

The subjects were members of the Information Systems Special Interest Group (ISSIG) of the Project Management Institute (PMI). At the time of the survey's administration, there were approximately 7,200 members of the Information Systems Special Interest Group (PMI-ISSIG). The members of this organization regularly receive an electronic newsletter or a paper newsletter that updates them on the activities of the PMI-ISSIG. The electronic newsletter is sent to those individuals who have shared their e-mail addresses with PMI-ISSIG (approximately 3,800 of the 7,200 members) and the paper newsletter is sent to all 7,200 members of the PMI-ISSIG. PMI-ISSIG agreed to include information about this research effort in both the electronic and paper versions of their newsletter. The newsletters asked project managers to fill out the web survey. In return, the respondents were told that their names would be entered into a drawing to win a polo-style shirt with the new PMI-ISSIG logo on it. The e-mail version of the newsletter had an electronic link directly to the survey web site.

The electronic newsletter was sent on July 23, 1998, and the paper newsletter was mailed out a few weeks later. As of July 28, 1998, there were approximately 85 responses received from the PMI-ISSIG. At this time a follow-up e-mail to the approximately 3,800 individuals with known e-mail addresses was sent out in an attempt to increase the number of responses. It was felt that the request for participation might have been overlooked in the first newsletter as it focused on several other issues in addition to the request for participation in the survey. The second request elicited a much higher response rate than the first request. The web survey was accessible until August 31, 1998, and at that time there were 507 responses.

There was no practical way to determine whether the respondents completed the web-based survey as a result of the request in the paper newsletter they received or if they were responding to the electronic newsletter. However there was a significant delay in the mailing of the paper newsletter to the 7,200 PMI-ISSIG members, and the majority of the 507 responses were obtained before the paper newsletter was mailed out. Therefore most of the respondents are assumed to be from the group of 3,800 members who received an electronic request for participation. Given that there were 507 respondents, this indicates a response rate of approximately 13.34%.

The 507 project managers who responded to the final survey ranged in age from 24 to 62, with an average age of 40.5. There were 371 males (73.2%), 127 females (25.0%), and 9 missing values (1.8%). Their previous software project management experience ranged from 1 to 38 years, with an average of 6.6 years. The average amount spent on a project was approximately 11 million dollars. The distribution of the amount spent on the projects was bi-modal with the largest number of respondents stating that they had spent either 500 thousand dollars or 1 million dollars on their most recently completed project.

Almost three-fourths of the respondents were the project manager of their most recently completed project. Another 10% classified themselves as a member of the project team, and 17% classified themselves as something other than the project manager or a member of the project team. The diversity of project roles should improve the generalizability of the results of this study as project risk was assessed from the viewpoint of individuals with many different responsibilities during a software development effort.

APPENDIX 2

Product and Process Performance Measures

Product performance measure

(assessed on a 7-point Likert scale with strongly disagree and strongly agree as anchors)

PROD1	The application developed is reliable.
PROD2	The application is easy to maintain.
PROD3	The users perceive that the system meets intended functional requirements.
PROD4	The system meets user expectations with respect to response time.
PROD5	The overall quality of the developed application is high.

Process performance measures

(assessed on a 7-point Likert scale with strongly disagree and strongly agree as anchors)

PROC1	The system was completed within budget.
PROC2	The system was completed within schedule.

Linda Wallace is an assistant professor in the Department of Accounting and Information Systems at Virginia Polytechnic Institute and State University. She obtained her PhD in computer information systems from Georgia State University in 1999. Her research interests include software project risk, project management, and agile software development. Her research has been accepted for publication in *Communications of the ACM, Journal of Systems and Software*, and the *Proceedings of the Academy of Management*.

Mark Keil is a professor in the Department of Computer Information Systems at Georgia State University. His research focuses on software project management, with particular emphasis on understanding and preventing software project escalation. His research is also aimed at providing better tools for assessing software project risk and removing barriers to software use. His research has been published in *MIS Quarterly, Sloan Management Review, Communications of the ACM, Journal of Management Information Systems, Information & Management, IEEE Transactions on Engineering Management, Decision Support Systems,* and other journals. He has served as an associate editor for the *MIS Quarterly* and as co-editor of the *DATA BASE for Advances in Information Systems.* He currently serves on the editorial board of *IEEE Transactions on Engineering Management.*

Arun Rai is the Harkins Professor in the Center for Process Innovation and Department of Computer Information Systems at Georgia State University. His research interests include digitally enabled supply chain management, diffusion and impacts of information technology, and management of systems delivery. His research has been published in *Accounting, Management and Information Technologies, Annals of Operations Research, Communications of the ACM, Decision Sciences, Decision Support Systems, European Journal of Operations Research, IEEE Transactions on Engineering Management, Information Systems Research, Journal of Management Information Systems, MIS Quarterly, Omega,* and other journals. He has served, or serves, on the editorial boards for *IEEE Transactions on Engineering Management, Information Systems Research, MIS Quarterly,* and other journals. Leading corporations, including A. T. Kearney, Bozell Worldwide, Daimler-Chrysler, Comdisco, SAP, IBM, among others, have sponsored his research work.