Components of Software Development Risk: How to Address Them? A Project Manager Survey

Janne Ropponen and Kalle Lyytinen

Abstract—Software risk management can be defined as an attempt to formalize risk oriented correlates of development success into a readily applicable set of principles and practices. By using a survey instrument we investigate this claim further. The investigation addresses the following questions: 1) What are the components of software development risk? 2) how does risk management mitigate risk components, and 3) what environmental factors if any influence them? Using principal component analysis we identify six software risk components: 1) scheduling and timing risks, 2) functionality risks, 3) subcontracting risks, 4) requirements management, 5) resource usage and performance risks, and 6) personnel management risks. By using one-way ANOVA with multiple comparisons we examine how risk management (or the lack of it) and environmental factors (such as development methods, manager's experience) influence each risk component. The analysis shows that awareness of the importance of risk management and systematic practices to manage risks have an effect on scheduling risks, requirements management risks, and personnel management risks. Environmental contingencies were observed to affect all risk components. This suggests that software risks can be best managed by combining specific risk management considerations with a detailed understanding of the environmental context and with sound managerial practices, such as relying on experienced and well-educated project managers and launching correctly sized projects.

Index Terms—Software risk, risk management, software development, project management, system failures, process improvement.

1 Introduction

Software development suffers chronically from cost overruns, project delays, unmet user needs, and unused systems ([12], [32]). This has continued despite huge advances in development techniques, tools, and software technologies ([20]). Since the early 80s, these difficulties have been alleviated through software risk management ([38], [7]). Software risk management can be defined as "an attempt to formalize risk oriented correlates of success into a readily applicable set of principles and practices" ([8, p. 33]). It embraces techniques and guidelines to identify, analyze, and tackle software risks items. A risk item denotes a particular aspect or property of a development task, process, or environment, which, if is ignored, will increase the likelihood of a project failure, e.g., threats to successful software operation, major sources of software rework, implementation difficulty, or delay ([34]). Overall, software risk management has raised considerable hopes for improving system development ([1], [7], [8], [10], [13], [37], [20]).

Even though practitioners have increasingly followed guidelines suggested by the proponents of software risk management, information about the impact of software risk management has been sparse and anecdotal. There are only a few empirical studies about the commonality and type of software development risks. In particular, there is little empirical evidence that shows which types of positive effects software risk management can have. In this paper, our goal is to expand our knowledge in this area. Using a survey instrument, we empirically delineate six components of software development risk:

1. scheduling and timing risks,
2. system functionality risks,
3. subcontracting risks,
4. requirement management risks,
5. resource usage and performance risks, and
6. personnel management risks.

Furthermore, we examine how factors related to risk management and environment correlate with successful management of these components. These findings are derived from a survey covering more than 80 project managers. The survey instrument is based on Boehm's ([7], [8]) work on top 10 software risks and risk management techniques. The paper is organized as follows: First, we discuss earlier research, formulate the research problem, and describe the research method. Next, using principal component analysis, we derive six components of software development risk. In the fourth section, we examine which risk management practices and environmental contingencies influence these components. We conclude by suggesting some principles for successful software risk management and identifying topics that invite future research.

- *J. Ropponen is with the Finnish Evangelical Lutheran Mission, Tähtitorninkatu 18, PO Box 1554, FIN-00003, Helsinki, Finland. E-mail: janne.ropponen@mission.fi.*
- *K. Lyytinen is with the Department of Computer Science and Information Systems, University of Jyväskylä, Seminaarinkatu 15, PO Box 35, FIN-40351, Jyväskylä, Finland. E-mail: kalle@cs.jyu.fi.*

Manuscript received 1 Apr. 1997; revised 27 June 1998; accepted 11 Sept. 1998.
Recommended for acceptance by D.R. Jeffery.
For information on obtaining reprints of this article, please send e-mail to: tse@computer.org, and reference IEEECS Log Number 104751.

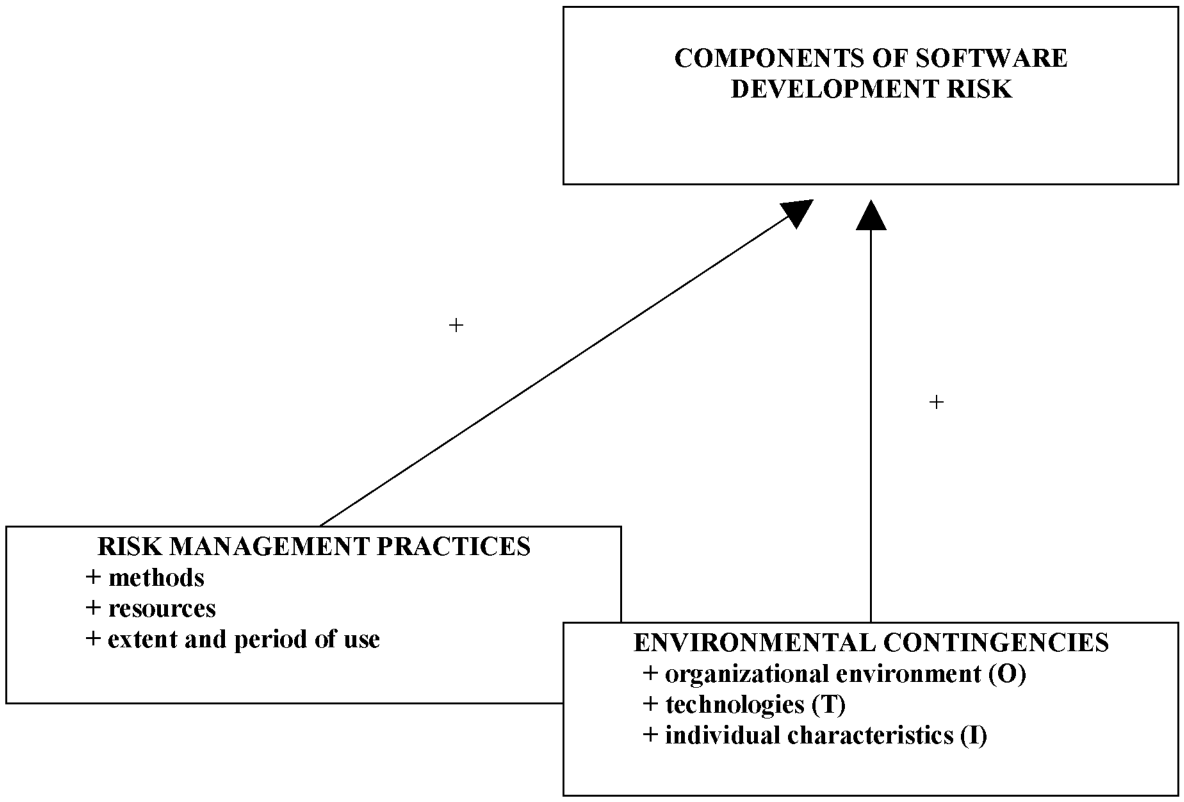

COMPONENTS OF SOFTWARE DEVELOPMENT RISK
+
+
RISK MANAGEMENT PRACTICES
+ methods
+ resources
+ extent and period of use
ENVIRONMENTAL CONTINGENCIES
+ organizational environment (O)
+ technologies (T)
+ individual characteristics (I)

Fig. 1. The research model.

2 Research Problem and Design

2.1 Related Research

The majority of risk management studies deals with normative techniques of managing risk ([7], [8], [10], [38], [13]). A few studies have classified software risk items ([2], [7], [44], [43], [45]). These studies consider software risks along several dimensions and have provided some empirically founded insights of typical software risks and their variation. Some empirical studies have tried to understand how one can effectively manage software risk. They discuss—normally by using case study data[1]—which risk management principles were (not) followed and try to learn about their (non)use. Overall, these studies provide illuminating insights into risk management deliberations, but are weak in explaining the true impact of risk management in generalizing from observations. A few studies have gone further to establish systematic models of risk management ([7], [13], [33], [37]). They all conclude that risk management efforts reduce the exposure to software risk and can thereby increase software quality and improve software development. Some studies focus solely on project delays ([19]) or deal only indirectly with software risks ([47]). Overall, our understanding of how software risk management can improve software development has remained fragmented and largely anecdotal.

2.2 Research Problem

In this paper, we investigate the impact of risk management practices on software development. We examine the following questions: 1) What are the components of software development risk? 2) What risk management practices and environmental contingencies help to address these components? The study is exploratory in nature and focuses on generation, rather than testing of, hypotheses because of the lack of well-established research models. In addressing these questions, we assume that software development risk can be decomposed into several distinct dimensions. Second, we postulate that various risk management methods and practices can influence different components of the software risk. In the same vein, we assume that there exists a connection between environmental contingencies and the capability to handle software risk. Fig. 1 summarizes the postulated connections in the research model. Each model component will be discussed next.

1. These studies have been either single site case studies ([10], [39], [36]), or multiple site case studies ([48]).

2.3 Research Model

2.3.1 Components of Software Development Risk

We define a risk as a state or property of a development task or environment, which, if ignored, will increase the likelihood of project failure. As measuring development failure is rife with both conceptual and instrumental problems ([21], [12]), we sought to measure development risk by identifying those items that have been recognized in the literature as software development pitfalls. The questions in the survey instrument (see Appendix 2 and [42] for details) were derived using the top 10 risk list of Boehm (7]). Boehm's list has been compiled by probing several large software projects and their common risks and is thus empirically grounded. It is extensive in terms of possible sources of risks ([34]) and it reflects a project manager's broad perspective on software risk (see Appendix 1). The list is also well-known and has been widely applied in practice to orchestrate risk management plans ([8], [10]). Hence, Boehm's list was chosen as it allowed us to investigate the level of exposure to various software risks.

In spite of its popularity and simplicity, Boehm's list forms an inductively derived collection of risk items and thus lacks a theoretical foundation ([34] [2]). It has multiple items which refer to the same phenomenon (like requirements related risks). It is also unclear how many distinct aspects of software risk it captures as many listed items covary with several risk management methods and also with one another ([34]). This suggests that the original list should be consolidated, in particular when some risk management techniques tackle several risk items. The benefit of this would be that a large set of risk items could be summarized into one independent component of software risk ([2]). This would provide a simpler basis for developing instruments to measure software risk. Obviously, this has a direct bearing on software risks management practices—especially on risk assessment.

We operationalized the software risk by listing a set of statements where each statement represents a claim with respect to how well this specific risk "item" has been managed in past projects. As advised by the project managers and experts in our pilot study (see Appendix 3), we extended Boehm's list with new risk items (e.g., deadline effect, completion in time, project cancellation, and managing project complexity). In the end, the survey instrument included 20 Likert scale items that measured risk levels associated with the extended Boehm's top 10 list (see Appendix 2). The questions presented are translated from Finnish. We used principal component analysis to reduce the number of chosen items into a smaller set of independent software components, as will be explained below.

2.3.2 Risk Management Practices

While investigating the use of risk management methods we utilized Boehm's (7]) classification of risk management methods.[2] We also inquired about respondents' general commitment to risk management by asking how extensively risk management methods had been used, whether the use was voluntary, and what the experiences of using methods were. These items were included because we postulated that organizations with a wider experience base have learned to apply risk management methods more effectively. We also asked about the amount of resources that had been allocated to manage risks.

2.3.3 Environmental Contingencies

We postulated that the capability to cope with the development risk is also contingent upon several external factors ([34], [10]). These include: 1) organizational characteristics, 2) technology characteristics, and 3) individual characteristics. Organizational characteristics cover items like organizational size, industry, type of system being developed (business systems/embedded systems), and contractual arrangements ([4], 5]). Technology characteristics include items like the newness of technology and the complexity and novelty of the required technological solutions ([38], [13], [48]). We also assumed that process technologies like development tools and methods affect the ability to manage risks (see [23], [47]). We also expected that project managers' characteristics and experiences (with projects of varying size and complexity) were one important predictor to successfully manage risks ([47], [29]). Similarly, software engineering education—both in computing and project management—was expected to affect the capability to handle software risks.

2.4 Data Set and Data Analysis

We collected a representative data set using a survey instrument by mailing the developed questionnaire to a selected sample of the members of the Finnish Information Processing Association (1991) whose job title was project manager or equivalent, e.g., development manager, and who had also managed projects in Finland.[3] For simplicity, we shall call all respondents project managers. In order to avoid a bias, we sent the questionnaire to at most two persons in one company. Where more than two project managers were listed for one company, the required two project managers were selected randomly. In this way, we obtained 248 people from our sample and mailed the questionnaire to them. The final data set consisted of responses from 83 project managers (response rate = 33.5 percent). The response rate was satisfactory and over the generally accepted rate (e.g., [24]). Overall, our sample included project managers both from internal IS departments and from software houses of varying size. The respondents in our sample had experiences of nearly 1,100 software projects. MIS applications covered approximately 76 percent of all projects included in the sample. The majority of their development projects were relatively small. The largest reported project was 672 man months. The average size of their last completed project was 15.24 man months (N = 75, STD 11.4).

The analysis of the data set obtained was conducted in several steps. First, we identified major software risk components by using principal component analysis (PCA) ([22]). PCA[4] is widely used to examine the underlying patterns for a large number of variables and to determine if the information (variance) can be summarized in a smaller set of factors or components for subsequent correlation or

2. Altogether, we derived 14 questions from this list. These were selected to cover those methods mentioned in Boehm's list.

3. This appeared to be a good decision as we got responses from 67 persons whose job title was exactly project manager. In addition to that, we got responses from nine development managers, five system development managers, one project director, and one project planner. All of them had experience in managing software projects.

4. In PCA, a correlation matrix is computed of all items included in the analysis, with unities inserted in the diagonal. Using factor analysis technique (with or without rotation), one extracts, for prediction purposes, as few as possible components (factors) that represent the variation of the original variables as much as possible. The resulting factor matrix displays the factor loadings (correlations) of the original items with the new extracted component. The cutting level for a statistically significant loading is at the lowest, 0.30, when sample sizes are 50 and over (as suggested by [22]). We applied a more precautionary cutting level of approximately 0.40 (two entries with loadings 0.38082 and 0.39192 were accepted). Our research setting met the criteria to use principal component analysis—prediction, finding a minimum number of factors to account for a maximum portion of variance, and the expected low unique and error variance as a portion of the total variance.

TABLE 1
Factor Matrix on Software Risks

Factors	Factor 1	Factor 2	Factor 3	Factor 4	Factor 5	Factor 6
Variables	Scheduling Timing	System function.	Sub-contract.	Requir. Manag.	Res. use perform.	Personnel manag.
Personnel shortfalls	-.00280	-.07714	-.02218	.01591	.02593	.75975
Problems in timetable	.83713	.04218	-.05162	.18396	-.15252	.08019
Resource usage and deadline	-.10617	-.11342	-.06551	.05988	.84819	.11891
Actual costs vs. estimated costs	.73074	-.27027	.16433	-.02437	.00298	.08001
Wrong size estimates	.61561	.22824	.22494	-.11404	.35527	-.01836
Estimates for personnel need	.55120	.43683	.42497	-.02526	-.00559	-.09131
Steady consumption of time	.01702	.19711	.03388	.48534	.06886	.40460
Insufficient expertise	.33951	.34520	.21398	-.31981	.07002	.39192
Managing project complexity	.52462	.08301	.23004	.24002	.48669	-.00887
Functions and properties correct	-.14384	.68828	.08495	.17152	.05864	.09647
Gold plating	-.03734	.34691	.25577	.60022	.10082	-.23089
Requirement changes	.28515	.04586	.07563	.75526	.07941	.12582
Changes in timetable	.65640	.02312	-.09419	.47302	.01124	-.08475
Satisfaction with the user interface	.07839	.79661	-.12782	.12238	-.13369	.13935
Shortfalls in externally furnished components	.12644	.09566	.79357	.04726	.06017	.03823
Unrealistic expectation of the personnel's abilities	.01215	.21776	.27694	.04472	.15203	.57234
Evaluation of performance requirements	.10498	.33530	.02009	.10129	.49317	.38082
Success in externally performed tasks	.07044	-.06682	.83802	.12630	-.02520	.16717
Estimation of hardware and software capabilities	.16448	.62809	.19168	.04457	.47688	-.06653

Legend of the table: Grayed entries denote the entries that loaded, i.e., have a high correlation with the factors defined in the column.

regression analysis. PCA analyzes the covariation of a set of variables and condenses the variation into a smaller number of underlying (latent) components. Second, we used ANOVA[5] with multiple comparisons to examine how risk management practices or environmental contingencies influence the management of identified software risk components. In most tests (if not otherwise described), we used a statistical significance level of a = 0.05. We also analyzed the validity and reliability of the survey instrument, as explained in Appendix 3.

5. The basic requirements of using variance analysis ([16])—normal distribution (Kolmogorov-Smirnov), independence of test groups, the homogeneity of variances (Levene, $\alpha = 0.01$)—were checked and the data was found appropriate for the analyses.

3 Components of Software Development Risk

By using PCA (eigenvalue 1.0, VARIMAX-rotation with Kaiser-normalization), we extracted six components from our original data set, which resulted in the best explanatory model (Table 1). These six components explained 63.2 percent of the total variation of the original variables, which can be regarded as well beyond sufficient ([22]). Moreover, 13 of the items loaded on only one factor and five items on two factors; overall, this resulted in a reasonably clean and easy to interpret model. Only one item—estimates for personnel needs—loaded on three factors. Few variables loadings on more than one factor indicate the clarity of the resulted model. One variable (project canceling) was dropped from the final analysis since it did not load to any of the components.[6] Overall, the result is statistically acceptable and represents a conservative number of factors (risk dimensions). The clarity of the model also makes its interpretation relatively straightforward. The six extracted risk components are:

1. scheduling and timing risks,
2. system functionality risks,
3. subcontracting risks,
4. requirement management risks,
5. resource usage and performance risks, and
6. personnel management risks.

We name the first factor *"Scheduling and timing risks"* since variables loading strongly to this factor relate to difficulties in scheduling the project correctly: problems in timetable, actual costs vs. estimated costs, changes in timetable (cf. "schedule risk" in [25]). All other items relating to this factor—wrong size estimates, managing project complexity, and estimates for personnel needs—are obvious reasons or consequences for problems in scheduling and timing. The second factor summarizes risk associated with getting the *"System functionality"* right. All variables loading deal with getting the system functionality correctly either from the user or from the technical point of view (satisfaction with the user interface, core functions and properties correct, and the estimation of hardware and software capabilities cf. "technical risk" [25]). It is also understandable that managing estimates of personnel needs relates to correct system functionality.

The third factor we call *"Subcontracting"* risks because success in managing externally performed tasks and shortfalls in externally furnished components both loaded strongly to this factor. Succeeding in estimating personnel needs seems to be connected to proper management of subcontracting. This is no wonder as poor management of subcontracting easily results and increased personnel needs. The fourth factor we call *"Requirements management."* This risk component deals with project managers' capability to manage the requirement change and avoid, e.g., gold plating. Both these items loaded strongly to the fourth factor. Continuous and uncontrolled changes in requirements lead to changes in timetables and make it difficult to keep resource consumption steady.

The fifth factor deals with *"Resource usage and performance"* risks. The variable loading highest to this factor is concerned with resource usage and deadline effect. The other items loading to this factor are: evaluation of performance requirements, managing project complexity, and estimation of hardware and software capabilities. Poor management of these goes together with a late and uncontrolled peak in project resource usage, i.e., the dead line effect (see, for example, [6]). The sixth factor we name *"Personnel management"* risks since the item loading most strongly deals with personnel risks (personnel shortfalls, cf. also "personnel" in [25]). Also, other items, like insufficient expertise and unrealistic expectations of the personnel's abilities, deal with the personnel. Obviously, poor mastering of performance requirements typically puts personnel under major stress in the late stages of a project and thereby increases personnel risks. It is also understandable that keeping project resource consumption (i.e., personnel load) steady relates to personnel management risks.

When we compare these six risk components with the five risk factors established in Barki et al. [2], the following can be observed. Only one of them, personnel management (which Barki et al. denote the lack of expertise), is common. The five other risk dimensions recognized in our study deal with operational project management aspects, i.e., those which a project manager can and must influence throughout the project trajectory. In contrast, Barki et al. observed risk dimensions that are beyond managers' operational control and which managers can and must recognize only before the project setup. These include the novelty of the project, the application size, or the organizational environment. It appears that our list extracts inherent risk dimensions of operative project management, whereas Barki et al.'s list extracts dimensions of IT investment risk in general.

The new list of six risk dimensions represents a more systematic set of risk components than Boehm's original list from the point of view of a project manager. Overall, the dimensions span process management aspects (components 1, 5), task related risks (2, 4), actor related risks (6), and structure/actor related risks (3) (cf. [34]). One interesting point is that none of these components is solely technical. Instead, technological aspects are embedded into software risk components (system functionality, subcontracting, and resource usage and performance). The result also demonstrates that software development risk items can be presented in a shorter and more compact set. Yet, each item in Boehm's original list can be mapped onto and presented with one component of the software development risk (see Appendix 1). For example, continuing requirement changes and gold plating are now linked to requirements management risks (see Appendix 1). The derived components also introduce new aspects of software risks (cf. resource usage and performance risks).

4 What Influences Software Risk Components

A summary of how risk management influences software risk components is shown in Table 2. Similarly, a summary of how environmental variables influence the management of software risk components is given in Table 3. These results were obtained using ANOVA with multiple comparisons. Next, we shall discuss the impact of both types of variables on each risk component.

4.1 Scheduling and Timing Risks

Mitigation of scheduling and timing risks necessitates the consideration of both risk management practices and environmental contingencies. We identified six factors altogether that influence the management of scheduling and timing risks. These were:

6. This seems to reflect the fact that project abandonment is not an item which is under the control of project managers and, therefore, it behaves in a radically different manner (see [28]).

TABLE 2
Software Risk Components Affected by Risk Management Practices

Risk components	Risk management practice variables	F	Prob.	N	
Scheduling risks	Number of projects where applied	7.2368	.0155	19	*
	Regularity of method use	7.9584	.0118	19	**
System functionality risks	Analysis of key decisions	7.6142	.0010	76	***
	Degree of method standardization	3.4295	.0506	19	*
	Degree of linkage to other methods	4.2272	.0336	19	*
Sub-contracting risks	---	---	---	---	
Requirements management risks	Analysis of poorly defined project parts	5.9383	.0041	75	**
	General use of risk management methods	6.6928	.0116	77	**
Resource usage and performance risks	Number of projects where method applied	10.6018	.0047	19	**
Personnel management risks	Extent of method application	5.6284	.0297	19	*
	Degree of method standardization	4.8792	.0222	19	*

Legend of the table: The asterisked entries denote the different significance levels observed (.05, ** .01, *** .001). Note also that the lower number of observations (19) is not due to missing data. Instead, the related questions were asked only of those respondents (19) who responded that they were following a risk management plan.*

1. experience in risk management methods,
2. regular use of risk management methods,
3. the size of the last completed project,
4. project manager's experience,
5. the industry, and
6. the type of the developed application.

Scheduling and timing risks seem to decrease linearly as more experience in using risk management methods is gathered. The group of project managers who had applied risk management methods in more than four projects performed significantly better than those who had less experience. Also, those who applied risk management methods continuously managed scheduling and timing risks significantly better than those who preferred to apply them only a few (four or less) times during a project. This finding is in line with the advice of published textbooks on the topic, i.e., risk identification "should be performed on a regular basis throughout the project" ([41]). Performance with scheduling and timing risks seems to improve with general project experience, as highlighted by the linearly increasing score of those project managers who have a larger experience base. When measured by the number of all managed projects, those project managers with only a few (one to four) projects scored significantly lower than those with 11 or more projects. If the person was managing only small projects (from six to 24 man months), an increase in scores was observed already after three projects.

The size of the project seems to significantly determine the ability to manage scheduling and timing risks. This is in line with earlier studies that smaller projects introduce less complexity and have a higher likelihood of having early problem recognition ([38], 12]). Measured by three different variables—duration, man months, and actual costs of the last completed projects—we observed that the largest projects (longer than 21 months, larger than 50 man months, and costing more than 500,000 USD) performed significantly worse than the smaller ones. This can be formulated as a risk management strategy: Keep the project size as small as possible or decompose the project into smaller units.

We also observed that project managers operating within retail business, accommodation, and nutrition services managed schedule and timing risks significantly better than those in other industries. This is due to different—presumably simpler and more standardized —application types used in these industries (cf. system functionality risks and retail business) and differences in IT maturity. An interesting finding is that scheduling and timing risks were handled significantly worse by those managing projects working on interactive systems than those working with systems involving little interactivity. We assume that this reflects the low uncertainty of specifying functionality and accordingly estimating schedules while developing batch type systems. Overall, the results suggest that projects' scheduling risk is mitigated by using experienced project managers, controlling the size of the projects, and by instituting risk management methods that direct organizations to conduct project reviews before and during the project execution.

4.2 System Functionality Risks

System functionality risks were influenced by the use of risk management methods, as well as environmental characteristics. The influencing factors are:

1. analysis of key decisions,
2. standardization level of risk management methods,
3. standardized linkages between risk management methods and other development methods,
4. industry,
5. project managers training, and
6. project manager's experience.

These results suggest that risk management can help substantially in managing functionality risks. In particular, the results suggest that project managers should learn to

TABLE 3
Software Risk Components Affected by Environmental Characteristics

Risk component	Environmental variable	Aspect	F	Prob.	N	
Scheduling and timing risks	Branch Accommodation and nutrition services[a]	O[7]	10.1511	.0021	77	**
	Branch: retail business	O	5.9243	.0173	77	**
	Number of managed projects	I	4.3589	.0162	77	**
	Number of projects. 1/2 - 2 man years	I	3.9790	.0497	77	*
	Length of the last project (months)	I	4.5484	.0141	70	**
	Man months of the last project	I	7.5653	.0011	68	***
	Actual cost of the last project	I	3.5791	.0359	49	*
	Processing mode of the application	T	3.8322	.0261	77	*
System functionality risks	Branch: telecommunications	O	5.2520	.0247	77	*
	Branch: transportation	O	8.8410	.0040	77	**
	Branch: retail business	O	3.8187	.0544	77	*
	Training for data processing	I	3.0399	.0539	77	*
	Number of managed projects	I	3.0662	.0526	77	*
Subcontracting risks	Turnover 1989	O	3.5919	.0375	40	*
	Turnover 1990	O	3.4431	.0406	48	*
	Training for project management	I	4.7347	.0116	77	**
	Number of managed projects (over 20)	I	5.8983	.0176	77	**
Requirements management risks	Degree of distributed hardware architecture[b]	T	7.5197	.0076	77	**
	Processing mode of the application	T	3.3522	.0404	77	*
	Obligation to use design and analysis procedure	T	4.8027	.0109	77	**
	Project management system	T	3.1852	.0182	77	**
Resource usage and performance risks	Branch: Wood and pulp industry	O	4.3162	.0412	77	*
	Branch: Public administration and defense	O	3.7849	.0555	77	*
	Turnover 1991	O	3.0994	.0542	51	*
Personnel management risks	Branch: Construction	O	3.6681	.0593	77	*
	Education for data processing	I	3.2910	.0427	77	*
	Degree of distributed hardware architecture	T	4.5442	.0363	77	*
	Use of analysis and design methods[c]	T	3.6502	.0308	77	*
	Project management system	T	3.3384	.0145	77	**

Legend of the table: the asterisked entries denote the different significance levels observed (.05, ** .01, *** .001).* [a]*Levene test for homogeneity of variances with 2.6767 with p-value 0.059.* [b]*Levene test for homogeneity of variancs 5.2007 with p-value 0.025.* [c]*Levene test for homogeneity of variances 4.5897 with p-value 0.013.* [7]*This acronym referes to whether the environmental item relates to individual characteristics (I), environment (O), or technology (T).*

frequently analyze key decisions, apply and standardize the use of risk management methods in projects, and link risk management methods to become an inherent part of the overall development method (i.e., the process model or project management strategy). The findings can be explained as follows: Key decisions are often made separately without thinking of the actual software design. However, selecting hardware, choosing subcontractors, and setting timetables and budgets can have a tremendous effect on the ability to develop user-friendly and functional systems. This also indicates the importance of integrating risk management activities to other systems development methods and introducing risk management as a standardized process innovation.

Not surprisingly, the industry in which systems were developed also influenced the management of system functionality risks. Telecommunications, transportation, and retail business fared significantly better than other industries. We conjecture that this finding can be explained by the types of systems being developed within these industries. For example, systems in the retail business (e.g., point of sales systems) often represent standard solutions whose functionality is well-known. This makes the management of system functionality risks easier. Similarly, the telecommunication industry often develops systems based on well-structured and detailed standards and, therefore, these industries have developed more standardized ways of managing functionality risks due to the product-like nature of project outcomes.

The study also suggests that the project manager's characteristics affect the exposure to system functionality risk. Extensive project experience seems to positively influence the mitigation of functionality risk. Those project managers who had managed more than 10 projects mastered system functionality risks significantly better than those with less experience. In addition, a project manager's level of education in computing had a clear impact on this risk. According to the results, project managers scored better if they had more education in computing. In particular, we identified a statistically significant difference between best scoring individuals, who had a major in computing, and the lowest scoring individuals, who had only on-site training. We assume that these findings demonstrate that extensive education is able to equip students with more systematic methods to deal with system functionality. Overall, the management of functionality risks can be improved by installing risk management methods (in particular, analysis of key decisions), integrating standardized risk management methods with normal development guidelines, and by hiring experienced and well-educated project managers.

4.3 Subcontracting Risks

Three environmental factors influenced the management of subcontracting risks: 1) the amount of project management training, 2) experience in managing large projects, and 3) the size of the software organization. Note that, surprisingly, none of the risk management practices directly influenced the control of subcontracting risk. This can signal a weakness in the coverage of the risk management methods used. Interestingly, general project management training did not help in mitigating subcontracting risks. In fact, for some reason, those who had almost no training scored best. However, their scores differed statistically significantly only from those with some training (not those with plenty of training). Nevertheless, this may indicate that problems of subcontracting are not widely covered in project management training. Extensive project management experience helped best to cope with the subcontracting risks, as those who had managed more than 20 projects scored significantly better than those with less experience. Improvements with subcontracting risks were related to the organizational size: project managers within middle-sized enterprises (turnover from $1,750,000 to $5,250,000) scored best. Their score also differed significantly from those project managers who worked in larger organizations. This is quite understandable as larger organizations use more subcontracting and thus face related risks. This risk item is once again best managed by building on the experience in managing projects. Another approach to consider is to explicitly improve practices around controlling contracted software management (like including it in project management training).

4.4 Requirement Management Risks

Two risk management aspects and four environmental factors help mitigate requirements management risks. This finding is interesting as requirements management has been identified as one of the most critical aspects in software development ([15], [17], [31], [43], [28]). Managing requirements are improved by a commitment to apply risk management methods and by focusing—frequently, rather than seldom—on analyzing poorly defined parts of a project plan or specifications. Interestingly, Boehm ([8]) does not relate the analysis of poorly defined parts to managing requirements changes. Instead, Boehm suggests high change threshold, information hiding, and incremental development as means to manage requirements changes. Nevertheless, ignored project factors that result from the lack of understanding obtained from a detailed decomposition of project plans and system functionality cause unwanted surprises and uncontrolled requirements changes.

Environmental factors influencing the requirement risks were

1. project management systems,
2. use of development methods,
3. hardware architecture, and
4. application type.

Interestingly, those respondents who had developed their own project management system scored significantly better in requirements management than those with a commercial project management system (MS Project, PMW, or other commercial tool). These findings can be understood as an outcome of "everything in control" illusion created by the fancy graphical illustrations found in some project management tools. In this way, they can hide requirements management problems, e.g., poorly defined parts of developed system. Moreover, an organization's own project management system may have an inherently better fit with that organization's requirements management process. Project managers employed by organizations mandating the use of development methods also scored significantly better than others. This confirms Humphrey's ([23]) observations that a disciplined systems environment helps control development processes. Project managers working with centralized architectures managed requirements risks significantly better than those developing systems with distributed architectures. Distributed systems offer more "bells and whistles" to develop fancy user interfaces and, consequently, less managerial control can be exercised. Managers working with interactive systems managed requirements better in comparison to those engaged with less interactive batch oriented software (statistical difference between groups with high and moderate interactivity). The better scores in applications with high interactivity are probably an outcome of a faster feedback loop when the development environment supports interface prototyping.

To summarize, management of requirements change can be improved by paying early attention to poorly defined parts and system functionality, standardizing the use of risk management methods, and structuring the development process. Decisions about target architectures and the type of system functionality also affect the risk component.

4.5 Resource Usage and Performance Risks

Three factors can influence resource usage and performance risks: 1) experience with risk management methods, 2) the industry, and 3) the size of the software organization. Management of resource usage and performance risks was improved with the experience of using risk management methods. The group of project managers who had applied risk management methods in more than four projects performed significantly better than those who had less experience. This highlights the importance of investing in gaining experience and developing an organizational experience base (cf. [3]). This seems to be true in particular of managing resource usage and performance risks. These risks especially troubled organizations delivering software to the wood and pulp industry, the public sector, and the defense industry. This may be due to the high performance requirements and complexity typical to these industries (process control and military systems).[7]

The results show that project managers within smaller organizations (turnover $1,750,000 or less) manage resource usage risks significantly better than those in larger organizations (turnover $5,250,000 or more). Larger organizations usually develop more complex and performance intensive systems with which this risk forms an issue.

4.6 Personnel Management Risks

Personnel risks correlate with seven different factors:

1. extent of applying risk management methods,
2. degree of standardization of risk management methods,
3. industry,
4. project managers' education,
5. hardware architecture,
6. use of analysis and design methods, and
7. the type of project management system.

Two risk management aspects relate to managing personnel risks: The wider the use of risk management methods, the better personnel risks are managed. Organizations where nearly all personnel utilized risk management methods performed significantly better than those where only one or a few applied them. Similarly, the scores were better if the risk management methods were standardized. In particular, we observed a statistically significant difference between project managers who reported voluntary use of risk management in contrast to those reporting obligatory use. Thus, general risk awareness is likely to increase the propensity to evaluate personnel risks in that it enables the identification of risks that are often organizationally sensitive. It demands more courage to air such risks if organizationally accepted risk management procedures are not available.

Interestingly, managers who worked with central computer architectures managed personnel risks better. This result reflects the greater uncertainty related to distributed systems. Complex innovations like distributed systems and client-server architectures, just becoming popular at the time of the survey, will increase unrealistic expectations of personnel's skills and the lack of expertise is a result of this. Not surprisingly, personnel risks were better managed by organizations that applied disciplined analysis and design procedures (see, e.g., [23]). Those project managers for whom the use of analysis and design methods was obligatory performed significantly better than those for whom the use was voluntary or recommended. Mature software organizations can better involve concerns related to personnel risk into their development practices. In addition, personnel management risks were better managed by those who used a tailored project management system. A significant difference was observed when comparing project managers with no project management system with those using PMW and those using another commercial system (other than MS Project). The explanation of this finding is straightforward: A tailored project management system seems to fit better to organizational needs and is thus more easily accepted and used. Hence, these systems are more likely to indicate insufficient expertise, unrealistic expectations of personnel's skills, etc.

In addition, the level of a project manager's computing education had a significant impact on the management of personnel risk. In general project managers scored better the more computing education they had. We identified a statistically significant difference between the best scoring individuals who had a major in computing in university and the lowest scoring individuals who had only on-site training. We observed also that the industry type influences the management of personnel risks. Organizations in the construction industry were weak in managing personnel risks. We assume that this relates to the types of systems being developed, sensitivity to market fluctuations, and development strategies employed.

5 Summary and Conclusions

In this paper, we have sought answers to two questions concerning software risk management: 1) What are the components of software development risk? and 2) what factors influence these components of risk? Using self-reported data from 83 project managers (covering nearly 1,100 projects), we derived six software risk components. These are:

1. scheduling and timing risks,
2. system functionality risks,
3. subcontracting risks,
4. requirements management risks,
5. resource usage and performance risks, and
6. personnel management risks.

The components embody essential dimensions of software risk and thus suggest a more manageable set of aspects that need to be heeded in software projects. The study also provides encouraging evidence of how the use of risk management methods can address some of these risks when properly aligned with other organizational procedures. At the same time, the study recommends that software organizations must tailor their risk management efforts to their development environment. The multiple comparison ANOVA analysis with the identified components reveals that risk management methods can help in managing several risk components. However, fewer relationships were detected between risk management and risk components than expected. For example, techniques that draw upon the traditional risk concept (risk exposure, risk reduction leverage) did not significantly reduce any risk

7. Strong market fluctuations typical to the wood and pulp industry can make resource allocation difficult. Difficulties with resource usage and performance risks in the public sector are eloquently exemplified by Beynon-Davis ([6]).

component.[8] This finding is analogous with empirical studies of management behavior ([35], [11]). Furthermore, no risk management measure affected subcontracting risks. Two widely used risk management techniques, analysis of key decisions and decomposition analysis, were found to be effective in mitigating system functionality and requirements management risks. Moreover, general awareness of software risks and their management was found to reduce requirements management risks. In addition, we observed that some features of risk management practices, like cumulated experience, organizational scope, their frequency of use, standardization, and linkages with other organizational procedures, had an effect. Hence, software risk management offers one important area for developing an organizational experience base ([3]).

All three environmental aspects influence the management of risk components. The analysis results suggest that software risk management is affected by the selection of target platforms, the use of disciplined development process, leveraging on experience, hiring well-educated people, and proper scoping of projects. These characteristics significantly influenced all risk components. An interesting observation is that technological attributes did not influence resource usage and performance risks in any way. This finding appears contradictory and should, therefore, especially be confirmed in future research. Another interesting finding is that the project size and domain did not relate to managing requirements risks. Overall, these results vividly illustrate the importance of understanding how environmental issues affect system development risk and lend strong support for contingency oriented systems development practices ([17], [30]).

Our advice for project managers can be formulated in simple terms. We encourage project managers to carefully identify and manage the six software risk components derived in this study. We also encourage project managers to take risk management methods seriously and develop an experience base of their use. Our results show that project managers in experienced software development organizations fared better in managing software risks. Project managers should also keep in mind that a variety of environmental contingencies affect the management of these risks. In this study, we identified several such organizational, technological, and individual characteristics. Some of these may not be in the direct control of the project manager (e.g., setting project deadlines, making decisions concerning subcontracting, or introducing improvements in the overall software process). They are, however, crucial issues to pay attention to and to take actions to reduce risks related to them.

Our findings need to be interpreted with caution. First, the selection of items in Boehm's original study on "top 10" risks may be biased due to the emphasis placed on large contracted software projects in the defense industry ([43]). Unfortunately, there is not much documentation of how the list was derived and how the risk items were ranked. Boehm only mentions that the list is "based on a survey of several experienced project managers" ([8, p. 35]) that covered interview data and project databases ([9]). As found during our pilot study, Boehm's list does not include all risk items which project managers mentioned to be important (see Appendix 3) and it was therefore expanded. We should also be aware that our study focuses exclusively on project managers' perceptions of managing project related software risks and ignores important risk items that relate to the broader management environment, including political risks, business risks, and implementation risks ([34], [45]).[9]

There are also some limitations related to the research method used. First, the study was solely based on project managers' self-reports. In future, we need to supplement these perceptual measures with factual behavioral and economic measures. The second problem is a possible sampling bias in using data from one country, though our sample was representative in terms of industries covered and types of systems developed. Third, our study shares the limitations of survey studies in establishing causal inference—it was a single period study without a control group ([21]). Moreover, the organization of the survey did not give us a mechanism to control the impact of historical incidents (both the personal history and a project phase) on respondent's responses. Therefore, respondents' evaluations may be biased due to anchoring effects and bias in viewing personal history. Changes in software risk management practices do, unfortunately, take place slowly. Therefore, we assume that sufficiently reliable measures of risk management performance were obtained. Based on statistical validity measures, we can assume that the obtained responses reflect the perceptions of project managers on overall risk management performance relatively well. The results obtained can thus be viewed as indications of causal connections.

The validity and generalizability of our findings can be improved in the future by using designs that can remove the limitations. We need to improve the instrumentation, use several measurement points (before and after the use of risk management methods), extend the study to other environments, and introduce quasi-experimental research designs (by controlling environmental variation). We invite both researchers and project managers to utilize our results when suggesting new measures to address components of software risk.

Appendix 1

Table 4 shows Boehm's top 10 risk list and the stakeholder perspectives concerned.

Appendix 2

Table 5 shows part of the questionaire used to create our measurement for risk management performance. Respondents were given the following directions:

In the following, we present a list of statements descriving your projects. Mark an appropriate alternative for each statement based on **your experience**. Choose one alternative based on how often the described situation occurs.

8. This can be due to the fact that the we had very few observations from organizations that used such methods.

9. For example, self-induced risks, such as those observed during project escalation ([26]).

TABLE 4
Boehm's Top 10 Risk List and Stakeholder Perspectives Concerned

#	Name of the risk item	Description	Stakeholder concerned[11]	Software risk component related to
1	Personnel shortfalls	Lack of qualified personnel and their change	Customer, users, subordinates, maintainers, bosses, project manager	Personnel management risks
2	Unrealistic schedules and budgets	Development time and budget estimated incorrectly (too low)	Customers, bosses, project manager	Scheduling and timing
3	Developing wrong software functions	Development of software functions that are not needed or are wrongly specified	User, project manager	System functionality
4	Developing wrong user interface	Inadequate or difficult user interface	User, project manager	
5	Gold plating	Adding unnecessary features ("whistles and bells") to software because of professional interest or pride or user's demands	Sub-ordinates, users, project manager	Requirements management
6	Continuing stream of requirement changes	Uncontrolled and unpredictable change of system functions and features	Sub-ordinates, user, project manager	
7	Shortfalls in externally furnished components	Poor quality of system components that have been delivered externally	Customers, bosses, project manager	Sub-contracting
8	Shortfalls in externally performed tasks	Poor quality or unpredictable accomplishment of tasks that are performed outside the organization.	Customers, bosses, project manager	
9	Real-time performance shortfalls	Poor performance of the resulting system	Users, customer, maintainers, project manager	Resource usage and performance
10	Straining computer science capabilities	Inability to implement the system because of lacking technical solutions and computing power.	Sub-ordinates, users, customers, project manager	

[11] *Stakeholder groups by Boehm and Ross [10]; subordinates and bosses denote those of a project manager.*

APPENDIX 3

Validation of the Survey Instrument and Control of Sampling Bias

Using Straub's ([46]) list of questions, we describe how the instrument was validated.

Content Validity

This question addresses whether instrument measures are drawn from all possible measures of the properties under investigation, i.e., are questions drawn from a representative universal pool. Because software risk measurement and software risk management performance are not well-understood areas, the issue of content validity is a serious one. We examined the available literature on software risk management, thus soliciting our questions from a representative sample. Boehm's list provided a good starting point due to its popularity and its wide empirical coverage (cf. [34]). Also, Boehm's study focused on a sample similar to that in our study. Boehm's list became our primary source as we discovered no other published studies on software risk items by the time of designing and implementing the research. We interviewed five experienced project managers and qualified researchers to find out how representative our set of questions was. This led to adding some new items to the risk management performance part as they were not captured by Boehm's notion. These related to managing process aspects such as deadline effect, project cancellation, stable time consumption, project complexity, and constant schedule changes (for details, refer to Appendices 1 and 2). Some of these items have also been identified elsewhere: cf. task and application complexity in Barki et al. ([2]), project escalation

TABLE 5
Measurement for Risk Management Performance

	Hardly ever	Rather seldom	Half	Rather often	Almost always	Boehm's risk #[12]
Your project has considerable problems due to personnel shortfalls.	1	2	3	4	5	1
Your project is completed according to the timetable.	1	2	3	4	5	2
Resource consumption reaches its top as you approach your project deadline	1	2	3	4	5	---
Actual project costs and estimated costs are nearly equal in your projects	1	2	3	4	5	2
Your project is canceled before completing it.	1	2	3	4	5	---
A failure to estimate project size interferes considerably with the implementation.	1	2	3	4	5	(2)
Demand of personnel is estimated correctly in your project	1	2	3	4	5	(1)
Time consumption of your project is constant.	1	2	3	4	5	---
Your personnel's expertise in methods, software, and equipment is insufficient	1	2	3	4	5	(1)
The complexity of your project and its effects are easy to manage.	1	2	3	4	5	---
Developed software functions and properties meet with users' needs.	1	2	3	4	5	3
Developed software includes complex, but only marginally useful properties.	1	2	3	4	5	5
Software requirements are continuously changed.	1	2	3	4	5	6
Your project timetable is changed continually.	1	2	3	4	5	---
Users are not satisfied with the implemented user interface.	1	2	3	4	5	4
Externally purchased components and equipment meet their your expectations in your project.	1	2	3	4	5	7
You have unrealistic expectations of the project members' skills.	1	2	3	4	5	(1)
Performance requirements (response time, computing efficiency) are estimated incorrectly.	1	2	3	4	5	9
Subcontracted tasks in the project are performed as expected.	1	2	3	4	5	8
Software and hardware capabilities are estimated incorrectly.	1	2	3	4	5	10

[12] *This column has been added here to clarify how to match the items with original Boehm's list of top 10 risk items presented in Appendix 1. Numbers relate to associated Boehm's risk items. The questions marked with numbers in brackets denote entries that were identified in the pilot study, but do more or less relate to a risk item in Boehm's list. Dashed entries were identified in the pilot study, but do not relate to Boehm's list.*

in Keil and Mixon ([27]), "were put under pressure to complete system quickly" by Beynon-Davis ([6, p. 1,162]), and artificial deadlines in Schmidt et al. ([45]).

Construct Validity

This question addresses whether the measures show stability across methodologies, i.e., that the data is a reflection of true scores or artifacts of the kind of instrument chosen. We sought to improve construct validity with several measures. First, we compared the data from our project manager interviews to the questionnaire data to resolve possible bias ([42]). Second, we conducted pilot tests which led to modifications in the questionnaire which improved both the content and construct validity. Moreover, during pilot testing, we asked the respondents to make suggestions to improve the questionnaire, e.g., by removing some items and adding new ones. This step introduced some amendments (for details, refer to Appendices 1 and 2). We also analyzed the possible discrepancies or variations in answers, but found none. The principal components analysis performed and reported in this paper provides itself a measure of the construct validity. As can be seen in Table 1, variables relating to each other loaded on the same factors, constituting six different constructs of software development risk. Hence, the construct validity can be regarded as sufficient.

Reliability

This question shows the measures' stability across the units of the observation, i.e., that a respondent answers the same or approximately the same questions the same way each time. The idea in improving reliability is to decrease the possibility that the measure is due to misunderstanding, error, or mistake, but instead reveals the true score. We utilized pilot tests to improve the reliability of our survey instrument. In particular, we carefully tested that all interviewees could understand the questionnaire items so that they could provide an unambiguous answer to each question. In order to improve this, technical terms (such as gold plating) were explained in the questionnaire. Moreover, we reduced potential misunderstanding by paraphrasing the questions concerning the use of specific risk management methods in a manner where a respondent did not know that he is being asked the use of a *risk* management method. This was done purposefully because many project managers do not necessarily know the technical terms of "risk identification, risk assessment" ([7]). Usually, high correlations between alternative measures or large Cronbach alphas are signs that measures are reliable ([14]). We computed Cronbach alphas for each component of software development risk (dependent variables in our study) by including the loading variables into the computation. The resulting Cronbach alphas for each component of software development risk are either close to or well over 0.60 (see Table 6) and we consider them satisfactory for our exploratory research setting (cf. [40]). The measures were also tested for homogeneity of variance (Levene test) and normal distribution (Kolmogorov-Smirnov test), which were all met. This shows that no "floor" or "ceiling" effects were observed due to misunderstanding or measurement reactivity.

Sampling Bias

We also carefully investigated reasons for not responding by calling 24 persons who had not replied. No bias in reasons for not responding was revealed. In fact, more bias would have been observed because 25 percent of non-respondents contacted had no experience in project management or their current job was not related to project management. Moreover, the investigation revealed that about 13 percent of the addresses used were out dated and we could not reach the person. The investigation also reveals that 55 percent of persons did not respond due to the fact that they had too little time or they never responded to any surveys. Similarly, one person considered the necessary background information (e.g., turnover figures) too hard to find. One person did not reply because the questions were inappropriate for his task. These findings suggest that some bias may be related to our results (e.g., an overly positive status of managing software development risks). However, we have a reasonable amount of evidence to warrant our claim that missing responses would have not considerably improved the reliability of our results.

TABLE 6
Cronbach Coefficient Alphas for the Components of Software Development Risk

Component of software development risk Loading variable	Alpha[13]
Scheduling and timing risks	**.7957**
Problems in timetable	.7307
Actual costs vs. estimated costs	.7750
Changes in timetable	.7742
Wrong size estimates	.7602
Managing project complexity	.7700
Estimates for personnel need	.7727
System functionality risks	**.6342**
Satisfaction with user interface	.5394
Functions and properties correct	.6475
Estimation of hardware and software capabilities	.4798
Estimates for personnel need	.5625
Sub-contracting risks	**.6648**
Success in externally performed tasks	.5443
Shortfalls in externally furnished components	.4620
Estimates for personnel needs	.6688
Requirement management risks	**.5972**
Requirement changes	.3479
Gold plating	.5786
Steady consumption of time	.6055
Changes in timetable	.5330
Resource usage and performance risks	**.6559**
Resource usage and deadline	.6696
Evaluation of performance requirements	.5380
Managing project complexity	.5968
Estimation of hardware and software capabilities	.5429
Personnel management risks	**.5550**
Personnel shortfalls	.5114
Unrealistic expectation of personnel's abilities	.4986
Steady consumption of time	.5361
Insufficient expertise	.5037
Evaluation of performance requirements	.4427

[13] *The alphas presented for each loading variable denote the resulting Cronbach alphas when this particular variable has been deleted.*

ACKNOWLEDGMENTS

We thank the associate editor and the four anonymous reviewers for comments that considerably helped to improve the manuscript and the analysis. We are also grateful to Esko Leskinen and Kari Heikkinen for their advice on statistical analyses. Thanks go also to Mark Keil and Lars Mathiassen for their comments to earlier drafts of the paper and Steven Kelly, who improved the language of the paper. Finally, we are indebted to project managers who voluntarily helped to validate the instrument and used their valuable time.

REFERENCES

[1] S. Alter and M. Ginzberg, "Managing Uncertainty in MIS Implementation," *Sloan Management Review*, pp. 23-31, Fall 1978.
[2] H. Barki, S. Rivard, and J. Talbot, "Toward an Assessment of Software Development Risk," *J. Management Information Systems*, vol. 10, no. 2, pp. 203-225, Fall 1993.
[3] V.R. Basili and J.D. Musa, "The Future Engineering of Software: A Management Perspective," *Computer*, pp. 90-96, Sept. 1991.
[4] C.M. Beath, "Strategies for Managing MIS Projects: A Transaction Cost Approach," *Proc. Fourth Int'l Conf. Information Systems*, pp. 133-147, Dec. 1983.

[5] C.M. Beath, "Managing the User Relationship in Information Systems Development Projects: A Transaction Governance Approach," *Proc. Eighth Int'l Conf. Information Systems,* pp. 415-427, Dec. 1987.

[6] P. Beynon-Davis, "Information Systems 'Failure' and Risk Assesment: The Case of London Ambulance Service Computer Aided Despatch System" *Proc. Third European Conf. Information Systems,* pp. 1,153-1,170, June 1995.

[7] B.W. Boehm, *Software Risk Management, Tutorial.* IEEE CS Press, 1989.

[8] B.W. Boehm, "Software Risk Management: Principles and Practices," IEEE Software, pp. 32-41, Jan. 1991.

[9] B.W. Boehm personal communication, Univ. of Technology, Helsinki, Finland, June 1995.

[10] B.W. Boehm and R. Ross, "Theory-W Software Project Management: Principles and Examples," *IEEE Trans. Software Eng.,* vol. 15, no. 7, pp. 902-916, July 1989.

[11] P. Bromiley and S. Curley, "Individual Differences in Risk Taking," *Risk Taking Behavior,* J.F. Yates, ed., pp. 87-132, Chichester: Wiley, 1992.

[12] F. Brooks, *The Mythical Man-Month: Essays on Software Engineering.* London: Addison-Wesley, Prentice Hall, 1975.

[13] R.N. Charette, *Software Engineering Risk Analysis and Management.* Intertext Publications McGraw-Hill Book Co., 1989.

[14] L.J. Cronbach, "Coefficient Alpha and the Internal Structure of Tests," Psychometrika, vol. 16, no. 3, pp. 297-334, 1951.

[15] B. Curtis, H. Krasner, and N. Iscoe, "A Field Study of the Software Design Process or Large Systems," *Comm. ACM,* vol. 31, no. 11, pp. 68-87, Nov. 1988.

[16] R.P. Cody and J.K. Smith, *Applied Statistics and the SAS Programming Language,* second ed. Elsevier Science, 1987.

[17] G.B. Davis, "Strategies for Information Requirements Determination," *IBM Systems J.,* vol. 21, no. 1, pp. 4-30, 1982.

[18] Finnish Information Processing Assoc. "Directory of Individual Business Members of the Organization,"Osto-opas Tietotekniikka 91: ATK-vuosikirja, Kustannusosakeyhtiö Otava, Keuruu, Finland, 1991.

[19] M. van Genuchten, "Why is Software Late? An Empirical Study of Reasons for Delay in Software Development," *IEEE Transactions Software Eng.,* vol. 17, no. 6, pp. 582-590, June 1991.

[20] M. Griffith and M. Newman, "Software Development Risk Management, a Special Issue," *J. Information Technology,* vol. 12, no. 4, 1996.

[21] W. Haga and M. Zviran, "Information Systems Effectiveness: Research Design for Causal Inference," *Information System J.,* vol. 4, no. 2, pp. 141-166, 1994.

[22] J. Hair, R. Anderson, R. Tatham, and B. Grablowsky, *Multivariate Data Analysis.* Tulsa, Okla.: PPC Books, 1979.

[23] W.S. Humphrey, *Managing the Software Process.* Software Eng. Inst., The SEI Series in Software Eng., Addison-Wesley, 1989.

[24] M. Igbaria, J.H. Greenhaus, and S. Parasuraman, "Career Orientations of MIS Employees: An Empirical Analysis," *MIS Quarterly,* vol. 15, no. 2, pp. 151-169, June 1991.

[25] D.W. Karolak, *Software Engineering Risk Management.* Los Alamitos, Calif.: IEEE CS Press, 1996.

[26] M. Keil, "Pulling the Plug: Software Project Management and the Problem of Project Escalation," *MIS Quarterly,* vol. 19, no. 4, pp. 421-447, 1995.

[27] M. Keil and R. Mixon, "Understanding Runaway IT Projects: Preliminary Results from a Program of Research Based on Escalation Theory," GSU CIS Working Paper, CIS-93-16, Dept. of Computer Information Systems, College of Business Administration, Georgia State Univ., 1993.

[28] M. Keil, P. Cule, K. Lyytinen, and R. Schmidt, "Against All Odds: A New Framework for Identifying and Managing Software Project Risks," *Comm. ACM,* vol. 41, no. 11, pp. 77-83, 1998.

[29] H. Kerzner, "In Search of Excellence in Project Management," *J. Systems Management,* vol. 38, no. 2, pp. 30-39, Feb. 1987.

[30] K. Lyytinen, "Different Perspectives on Information Systems: Problems and Their Solutions," *ACM Computing Surveys,* vol. 19, no. 1, pp. 5-44, 1987.

[31] K. Lyytinen, "Expectation Failure Concept and Systems Analyst's View of Information System Failures: Results of an Exploratory Study," *Information & Management,* vol. 14, no. 1, pp. 45-56, Jan. 1988.

[32] K. Lyytinen and R. Hirschheim, "Information Systems Failures—A Survey and Classification of the Empirical Literature," *Oxford Surveys in Information Technology,* Oxford Univ. Press, vol. 4, pp. 257-309, 1987.

[33] K. Lyytinen, L. Mathiassen, and J. Ropponen, "A Framework for Software Risk Management," *J. Information Technology,* vol. 11, no. 4, pp. 275-285, 1996.

[34] K. Lyytinen, L. Mathiassen, and J. Ropponen, "Attention Shaping and Software Risk—A Categorical Analysis of Four Classical Approaches," *Information Systems Research,* vol. 9, no. 3, pp. 233-255, Sept. 1998.

[35] J. March and Z. Shapira, "Managerial Perspectives on Risk and Risk-Taking," *Management Science,* vol. 33, pp. 1,404-1,418, 1987.

[36] L. Markus and M. Keil, "If We Build It, They will Come: Designing Information Systems that Users Want to Use," *Sloan Management Review,* pp. 11-25, Summer 1994.

[37] L. Mathiassen, T. Seewaldt, and J. Stage, "Prototyping and Specifying: Principles and Practices of a Mixed Approach," *Scandinavian J. Information Systems,* vol. 7, no. 1, pp. 55-72, Apr. 1995.

[38] W. McFarlan, "Portfolio Approach to Information Systems," *J. Systems Management,* pp. 12-19, Jan. 1982.

[39] B.S. Neo and S.L. Kwong, "Managing Risks in Information Technology Projects: A Case Study of TradeNet," *J. Information Technology Management,* May 1994.

[40] J.C. Nunnally, *Psychometric Theory.* New York: McGraw-Hill, 1978.

[41] Project Management Inst., "A Guide to the Project Management Body of Knowledge,"PMI Standards Committee, Project Management Institute, Upper Darby, Pa., 1996.

[42] J. Ropponen, "Risk Management in Information System Development," Technical Report TR-3, Dept. of Computer Science and Information Systems, Univ. of Jyväskylä, Finland., Lic thesis, 1993.

[43] J. Ropponen, "Software Development Risks and Management Practices: A Project Manager Survey," *Beyond The IT Productivity Paradox—Assessment Issues,* L. Willcocks, ed., to be published by John Wiley.

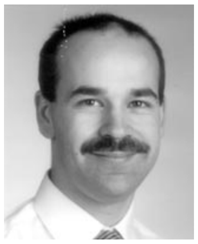
[44] J. Ropponen and K. Lyytinen, "Can Software Risk Management Improve System Development: An Exploratory Study," *European J. Information Systems,* vol. 6, pp. 41-50, 1997.

[45] R. Schmidt, K. Lyytinen, M. Keil, and P. Cule, "Identifying Software Project Risks—An International Delphi Study," Hong Kong Univ. of Science and Technology, unpublished working paper, 1998.

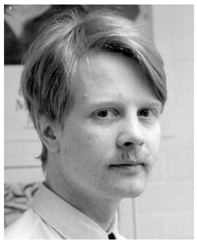
[46] D.W. Straub, "Validating Instruments in MIS Research," *MIS Quarterly,* pp. 147-165, June 1989.

[47] V. van Swede and J. van Vliet, "Consistent Development: Results of a First Empirical Study on the Relation between Project Scenario and Success," *Proc. Sixth CAiSE Conf.,* G. Wijers and S. Brinkkemper, eds., 1994.

[48] L. Willcocks and H. Margetts, "Risk Assessment and Information Systems," *European J. Information Systems,* vol. 3, no. 2, pp. 127-138, 1994.



Janne Ropponen recently finished his PhD on risk management. His research interests include software risk management and process improvement. He has published in the *European Journal of Information Systems*, *Information Systems Research*, and the *Journal of Information Technology*. He is currently on leave from Nokia Telecommunications, where his tasks include risk management and process improvement. He also works as a commercial pilot working under the Mission Aviation Fellowship.



Kalle Lyytinen is a full professor in computer science (information systems) and currently the Dean of the Faculty of Information Technology at the University of Jyväskylä, Finland. He received his PhD from computer science in University of Jyväskylä. His research interests include risk management, requirements engineering, CASE and method engineering, CSCW applications, diffusion of complex networked technologies, and electronic commerce. He has published more than 70 articles in leading refereed journals and conferences, including *Communications of the ACM*, *IEEE Transactions on Software Engineering*, *Information Systems Research*, *MIS Quarterly*, and *ACM Computing Surveys*. He is the senior editor of *MIS Quarterly* and he serves on the editorial boards of *Information Systems Research*, *Information Systems Journal*, *Requirements Engineering*, *Journal of Strategic Information Systems*, and *Information and Organization*.