Journal of the Association for Information Systems

Volume 8 | Issue 11 Article 33

11-2007

A Contigency Model for Requirements Development

Tuure Tuunanen
The University of Auckland, New Zealand, tuure@tuunanen.fi

Matti Rossi
Helsinki School of Economics, Finland, mrossi@hkkk.fi

Timo Saarinen
Helsinki School of Economics, Finland, timo.saarinen@hse.fi

Lars Mathiassen
Georgia State University, lmathiassen@gsu.edu

Recommended Citation
Tuunanen, Tuure; Rossi, Matti; Saarinen, Timo; and Mathiassen, Lars (2007) "A Contigency Model for Requirements Development," *Journal of the Association for Information Systems*, 8(11), .
DOI: 10.17705/1jais.00143
Available at: https://aisel.aisnet.org/jais/vol8/iss11/33

Journal of the Association for Information Systems

JAIS

Research Article

A Contingency Model for Requirements Development

Lars Mathiassen
Georgia State University
lmathiassen@gsu.edu

Tuure Tuunanen
The University of Auckland, New Zealand
tuure@tuunanen.fi

Timo Saarinen
Helsinki School of Economics, Finland
timo.saarinen@hse.fi

Matti Rossi
Helsinki School of Economics , Finland
matti.rossi@hse.fi

Abstract:

Drawing upon the requirements and software development literature, the present study proposes an integrative contingency model for requirements development. Based on 116 quality journal articles, we analyze requirements development risks, requirements development techniques, and heuristics for how they are effectively related. Subsequently, we synthesize the insights from the identified literature into a model for requirements development that relates patterns of risk resolution to archetypical risk profiles. The model integrates the literature on requirements and software development; sets the scene for future research; and, finally, proposes how practitioners can manage risks in requirements development projects.

Keywords: *Requirements development; contingency theory; risk management.*

* Juhani Iivari was the accepting senior editor. Ann Hickey, Suzanne Rivard, and Uolevi Nikula were the reviewers. This was submitted on July 31 2006, and went through 4 revisions.

Volume 8, Issue 11, Article 2, pp. 569-597, November 2007

Introduction

The requirements for the first software applications were often easy to identify, since most applications were developed by scientists to support their own needs and purposes. However, as businesses started to develop software for users, it soon became necessary to gather, explicate, and understand user needs and have users participate in software development (Lamb and Kling, 2003; Markus and Mao, 2004; McKeen et al., 1994). This has resulted in a considerable variety of techniques (Byrd et al., 1992; Chatzoglou and Macaulay, 1996; Davis, 1982; Keil and Carmel, 1995) to support development of requirements for software applications. Some would argue that the constant stream of techniques has turned into a methodology jungle (Jayaratna, 1994).

Researchers have responded by creating contingency models for adopting techniques to specific contexts. These models offer categories of contextual factors, 7portfolios of techniques, and approaches to fit techniques to contexts (Iivari, 1992; Kickert, 1983). The models integrate knowledge within particular disciplines (Andrea and Zmud, 2002; Barki et al., 2001; Hickey and Davis, 2004) and are often based on risk management models, i.e., the context is analyzed in terms of risks, techniques are seen as means for risk resolution, and fit is expressed as heuristics that link techniques to contexts based on their capacity to resolve risks (Lyytinen et al., 1998).

As a first attempt to integrate available knowledge, Alter et al. (1978) introduced a contingency model to help develop software for decision support. McFarlan (1981, 1982), in turn, suggested that software applications should be developed through appropriate integration internally amongst developers and externally between developers and users. Davis (1982) focused on development of requirements for software applications and integrated available knowledge into a contingency model to reduce the uncertainty of the task. Davis' model was later revised and extended by Fazlollahi and Tanniru (1991). Davis (1982) and Fazlollahi and Tanniru (1991) have a narrow focus on requirements, whereas other models address requirements development as part of a broader approach to software development (Alter et al., 1978; McFarlan, 1981, 1982)).

Unfortunately, there have been no attempts to integrate new knowledge that has become available (Hickey and Davis, 2004) even though requirements development practice and research have changed considerably over the past fifteen years (Neill and Laplante, 2003). During this period, ubiquitous computing, increased emphasis on inter-organizational applications, and demands for shorter project life-cycles have introduced new techniques and changed the risk profile of requirements development projects. Developers are often challenged to establish effective interaction with would-be users who are outside organizational reach (Duggan and Thachenkary, 2004, Frolick and Robichaux, 1995, Peffers et al., 2003), a challenge that increases when users do not know how to describe their needs (Walz et al., 1993).

The purpose of this paper is therefore to propose a model for requirements development that integrates current research. Concerning the status of the model, it is hypothetical in nature and purely based on a literature review; empirical validation and further development of the model requires additional research. Concerning the focus of the model, the software engineering literature emphasizes elicitation and representation of requirements and issues related to management of requirements changes (Dubois and Pohl, 2003, Pohl, 1994). The information systems literature emphasizes users and the negotiation and interpretation of requirements to develop software applications of feasible quality (Byrd et al., 1992, Keil and Carmel, 1995, Markus and Mao, 2004, McKeen et al., 1994). The proposed model's focus on requirements development covers, in this way, a cyclic process (Nguyen and Swatman, 2003) starting with vague ideas that are presented informally and often inconsistently, leading toward a desired end state where there is a common agreement on a set of relatively formalized requirements that can serve as a blueprint for software design and implementation (Pohl, 1994). The process encompasses activities ranging from requirements elicitation and analysis to specification and conflict resolution (Pohl, 1994).

Our interest lies in understanding how requirements development techniques can be applied to different contexts. We therefore adopt contingency theory (Barley, 1990, Iivari, 1992, Pugh, 1998, Weill and Olson, 1989) and risk management (Barki et al., 2001, Lyytinen et al., 1998) as our basis. Specifically, we seek to develop a risk-strategy model (Iversen et al., 2004) based on the assumption that requirements development outcomes depend on the way in which techniques are applied to resolve risks. As risk-strategy models relate a few types of software risks to a few types of software resolution techniques, we seek to synthesize current research into a simple, yet comprehensive understanding of requirements development risks and techniques.

We first present our method for reviewing the literature. Next, we analyze the identified literature guided by specific research questions and synthesize the results into an integrative contingency model that proposes how patterns of risk resolution can be used to address archetypical risk profiles. Finally, we discuss the proposed model, outline avenues for future research, and suggest implications for practice.

Review Methodology

A literature review should be complete and focus on concepts. Two of the key issues are, therefore. how to identify the relevant literature and how to structure the analysis of the included literature (Webster and Watson, 2002, Weill and Olson, 1989, Zhang and Li, 2005).

Identifying the Literature

Focusing on journals within information systems and other computing areas, we consider requirements development research and those parts of software development research that cover requirements development activities. Both streams of literature offer relevant insights for proposing a contingency model for requirements development: one from a specialized point of view, as expressed in Davis' model (1982) of requirements determination, and the other from a broader, more general point of view, as expressed in McFarlan's (1981, 1982) general model of software development with clear implications for requirements development.

Webster and Watson (2002) suggest that the process of identifying relevant articles in leading journals requires going backward by reviewing citations used by the selected articles, and then going forward by identifying articles citing key articles identified in the previous steps. In addition to these generic steps, we have identified articles in specialized journals that publish high quality articles about requirements development. Finally, Weill and Olson (1989) suggest using structured critique to further guide article selection. We have implemented these recommendations into a seven-step approach as summarized in Table 1.

In the first step, we used the Web of Science–service with access to scientific literature from 1990 and onward to identify relevant requirements and software development research. In this process, we used broad key words to identify the 500 most relevant articles within requirements development and the 500 most relevant articles within software development. Initially, using the keywords 'requirements and determination' or 'requirements and elicitation' allowed us to identify research with a particular focus on requirements development issues across information systems, software engineering, and other computing areas. Subsequently, because of the strong emphasis on requirements development issues within the software engineering literature, we used the keywords 'software development methods', 'software engineering management', 'software process management', and 'software life cycle' to identify additional relevant research. This keyword search was done May 15th 2004. [1]

In the second step, we selected those articles from step one that were published in leading journals. Several articles identify leading journals (Gillenson and Stutz, 1991; Hardgrave and Walstrom, 1997; Holsapple et al., 1994; Mylonopoulos and Theoharakis, 2001, Whitman et al., 1999). We chose two recent lists published in 2003. One focuses on information systems journals (Peffers and Tang, 2003), and the other on journals with primary focus on information systems and computing areas (Katerattanakul et al., 2003). By combining these lists, we arrived at leading journals relevant for our study (see Appendix I for details). We then used the aggregate list to select articles from leading journals.

The two first steps led to a total of 135 articles. Many of these turned out to be of little or peripheral relevance to our study because of the broad key word search adopted in the first step. Therefore, we conducted a third step in which we manually filtered the articles based on specific criteria of relevance, see Table 1. As the Web of Science does not include articles written before 1990, in the fourth step we went backward through the reference lists of all articles included in step three. Within both streams of literature, we compiled an aggregate reference list sorted according to first author and included those articles that had two or more citations in the newer articles in leading journals, i.e., we included those older articles that had most impact in the newer literature.

In the fifth step, we then manually filtered the two lists of older literature according to the rules of step three. Subsequently in a sixth step, we added two specialized sources, Requirements Engineering Journal and IEEE Software. These are not

[1] The literature search was limited in two ways. First, we did not use keywords such as 'requirements and analysis', 'requirements and determination', 'requirements and engineering', 'requirements and gathering' and 'requirements and specification'. Many articles use some of these keywords, which may have led to the exclusion of some of the relevant literature. Second, while using keywords such as 'software development methods' and 'software life cycle', we did not use corresponding keywords such as 'information systems development methods' and 'information systems development lifecycle'. This may bias our initial screening of the literature toward software engineering rather than information systems.

Journal of the Association for Information Systems

considered leading journals in information systems according to the adopted lists, but both journals publish high quality articles on requirements development from a software engineering perspective. We used the same keywords as in step one and the same heuristics as in step three to identify relevant articles. This step resulted in the selection of 25 additional articles. In the final seventh step, we combined the lists of steps three, five, and six to generate the list of considered articles, see Appendix II.

Table 1 Literature selection

Selection Step	Requirements Development	Software Development	Sum of Articles
Step 1: Broad search in Web of Science (May 15th 2004)	Keywords: 'requirements and determination' or 'requirements and elicitation'. **2,633** of 18,860,525 articles. Search limited to 500 most relevant.	Keywords: 'software development methods', 'software engineering management', 'software process management', 'software life cycle'. **4,320** of 18,684,867 articles. Search limited to 500 most relevant.	6953
Step 2: Selecting articles in ranked journals (See Appendix I for the list of journals)	Result: **40** articles.	Result: **97** articles.	137
Step 3: Selecting most relevant articles	Criteria: 1) Should evaluate techniques for requirements development. Result: **32** articles.	Criteria: Should cover requirements development activities by either 1) theorizing about the software development process or product over the whole life-cycle or by 2) taking a comprehensive approach to software development that includes requirements problems and their solutions. Result: **24** articles.	56
Step 4: Identifying pre-1990 articles	Result: list containing 56 articles with two or more citations.	Result: list containing 62 articles with two or more citations.	174
Step 5: Selecting most relevant articles	Result: **14** new articles out of the 56 articles with two or more citations.	Result: **21** new articles out of the 62 with two or more citations.	91
Step 6: Selecting articles in Requirements Engineering Journal and IEEE Software	Result: **17** articles	Result: **7** articles.	116
Step 7: Combining results from step 3, 5, and 6	Result: **63** articles.	Result: **52** articles.	116
Number of reviewed articles: **116** (see Appendix 2 for details).			

Although we do not claim that this initial screening of the relevant literature is complete and unbiased, we believe that it provides rich enough material for our analysis of requirements risks in the existing literature.

Structuring the Review

Our choice of contingency theory as the basis for the review is supported by Hickey and Davis's unified model of requirements elicitation (2004) in which they suggest using contextual factors as a basis for adoption of requirements techniques. Historically, contingency theory has sought to develop generalizations about the fit between context and organizational design (Barley, 1990; Pugh, 1998). Contingency theory originated in work by Burns and Stalker (1961) that studied how external circumstances affected mechanistic and organic approaches to organization and in Woodward's (1965) study of how patterns of management varied according to technical differences (Lawrence and Lorsch, 1974). While classical contingency theory is useful as a basis for management, Barley argues that it is limited in two ways (1990). First, it largely ignores how context-technology relations arise and change. Second, it tends to ignore human action and, therefore, fails to explain variations in observed relationships between organizational context and technology.

To overcome these limitations, we adopt risk management as a specific form of contingency theory that has been successfully applied within the Information Systems discipline (Lyytinen, 1988; Lyytinen et al., 1998). Risk management

generalizes patterns of relations between organizational contexts (in the form of risk items) and use of technologies (in the form of resolution techniques) in ways that support human action and include change over the software development life-cycle (Lyytinen et al., 1998). In line with Bergman et al. (2002), we consequently see requirements development as a practice that is driven by iterations between functional and political issues, in which decision making is restricted by bounded rationality (Simon, 1955, Simon, 1957), and in which the actual orchestration of actors, problems, solutions, and decisions can take arbitrary forms (Cohen et al., 1972).

Specifically, we have structured our review according to the basic elements of contingency theory and risk management in information systems research as shown in Table 2. First, we focus on contextual factors as risk items that have the potential to lead to wrong or inadequate software solutions, rework, implementation difficulty, delay, or uncertainty (Boehm, 1991; Lyytinen et al., 1996). Requirements development involves risks in the form of loss or uncertainty that require managerial intervention (Barki et al., 1993; Barki et al., 2001; Lyytinen et al., 1996). Keeping in mind that our goal is to synthesize a risk-strategy model (Iversen et al., 2004) for requirements development, this motivates the first research question:

> How can we characterize requirements development risks in a simple, easy to apply, and yet comprehensive way?

Table 2 Structuring the Review

Contingency theory	Risk management	Review question
Context	Risk item	How can we characterize requirements development risks in a simple, easy to apply, and yet comprehensive way?
Technology	Resolution technique	How can we characterize requirements development techniques in a simple, easy to apply, and yet comprehensive way?
Fit	Heuristics	How can we effectively apply patterns of risk resolution to risk profiles in requirements development?

Second, we focus on technological options as resolution techniques that in the context of requirements development represent ways of addressing risks through formalization of activities and notations. Following Hickey and Davis (2004), requirements techniques describe what to do in specific phases of the process, and they are usually accompanied by tools and notations. This motivates the second research question:

> How can we characterize requirements development techniques in a simple, easy to apply, and yet comprehensive way?

Third, contingency models and risk management approaches build on particular views of how to effectively respond in specific contexts. This is why we analyze the identified literature to understand how to apply patterns of requirements development techniques to archetypical risk profiles. This motivates the third research question:

> How can we effectively apply patterns of risk resolution to risk profiles during requirements development?

As the goal of the literature review is to propose a model for managing requirements development risks, the analysis is further guided by the type of risk management model we seek to develop. Iversen et al. (2004) identified four types of models. First, there are risk lists (e.g., Barki et al., 1993) that contain generic risk items (often prioritized) to help managers focus on possible sources of risk; they do not offer appropriate resolution techniques. Second, there are risk-action lists (e.g., Boehm, 1991) that contain generic risk items (often prioritized), each with one or more related risk resolution technique. Third, there are risk-strategy models (e.g. McFarlan, 1982) that relate archetypical risk profiles to specific patterns of resolution techniques. The risk profile is assessed along a few risk factors (e.g., into high or low), making it possible to classify the project as having one of a few archetypical risk profiles. For each profile, the model offers a specific pattern of resolution techniques. Finally, there are risk-strategy analysis models (e.g., Davis, 1982). These are similar to risk-strategy models, but they apply different heuristics. There are no direct links from archetypical risk profiles to patterns of resolution techniques. Instead, these models offer a stepwise process in which risks are identified and linked to techniques to form an overall resolution strategy.

Iversen et al. (2004) suggest that risk-strategy models are the most advantageous from a usage point of view, but they are harder to build and modify than the other models. Accepting these difficulties, we choose to synthesize the identified literature into a risk-strategy model that in a straightforward and simple way links patterns of risk resolution to outcomes. This choice has, as indicated in the three research questions, specific implications for how we should analyze the literature.

Journal of the Association for Information Systems

First, we should develop a simple, but still comprehensive conception of requirements development that will allow us to identify a few archetypical risk profiles and a few patterns of risk resolution techniques. Second, we should identify heuristics for relating requirements development risk profiles to patterns of resolution techniques.

Analysis

In the following, we present the results of the analysis of the identified literature, structured according to the three research questions. As this analysis is based on the articles selected for review, it might in some cases not give credit to the original sources on which these articles were based.

Requirements Development Risks

Davis' (1982) contingency model for requirements development adopts a one-dimensional understanding of risk, i.e., task uncertainty, and proposes to use different techniques according to the level of uncertainty. Fazlollahi and Tanniru (1991) extended this model by differentiating between two sources of risks: uncertainty and equivocality. Mathiassen et al. (1995) proposed a similar two-dimensional conception of risks distinguishing between uncertainty and complexity of requirements. While these models are well aligned and build on classical contingency theory, they fail to capture that requirements development increasingly involves external stakeholders and that the gap between developers and would-be-users as a consequence has increased (Coughlan and Macredie, 2002; Damian and Zowghi, 2003; Lang and Duggan, 2001; Lausen and Vium, 2005; Mich et al., 2005). If a project cannot effectively identify, connect to, and interact with would-be users, it is difficult to discover requirements about the software and its practical use. This analysis suggests a view of requirements development risks in line with available models (Davis, 1982; Fazlollahi and Tanniru, 1991; Mathiassen et al., 1995) and extended with requirements identity risks to reflect the recent emphasis in the identified literature on the possible gaps between developers and would-be-users. Introducing identity risks as a third category, however, creates an overlap with the emphasis on uncertainty risks in existing models. In fact, both identity risks and complexity risks imply some element of uncertainty in the form of unknown or poorly understood requirements. As a consequence, we suggest changing the notion of uncertainty risks to volatility risks, thereby making the three categories more clearly distinct. The resulting view of requirements development risks is elaborated below and summarized in Table 3.

Table 3. Requirements development risks

Risks	Definition	References	Mapped Articles
Requirements identity	The availability of requirements; high identity risk indicates requirements are unknown or indistinguishable	(Damian and Zowghi, 2003, Hirschheim and Newman, 1991, Keil and Carmel, 1995, Lang and Duggan, 2001, Lyytinen, 1988, Pai, 2002, Rai and Al-Hindi, 2000, Ravichandran and Rai, 1999, Ravichandran and Rai, 2000, Regnell et al., 2001, Salaway, 1987, Walz et al., 1993, Watson and Frolick, 1993, Zultner, 1993)	13
Requirements volatility	The stability of requirements; high volatility risk indicates requirements easily change as a result of environmental dynamics or individual learning	(Davis, 1982, Fazlollahi and Tanniru, 1991, Houston et al., 2001, Kraut and Streeter, 1995, Lyytinen, 1987, Mathiassen et al., 1995, Nidumolu, 1995, Willcocks and Margetts, 1994)	8
Requirements complexity	The understandability of requirements; high complexity risk indicates requirements are difficult to understand, specify, and communicate	(Boehm and Ross, 1989, Brooks, 1987, Fazlollahi and Tanniru, 1991, Glass et al., 1992, Lyytinen, 1987, Mathiassen et al., 1995, Mills, 1999)	7

The identified literature discusses risks related to **requirements identity** in situations where there is a communication gap between developers and would-be users, e.g., in development of generic applications or mass market software (Regnell et al., 2001). Furthermore, requirements identity risks relate to the availability of requirements; high risk means that requirements are unknown or indistinguishable (see Table 3). Identity risks depend on the physical, conceptual, and cultural distance between developers and would-be users. The shift from internal users toward external users occurs as software is increasingly developed to markets and used by customers and business partners (Damian and Zowghi, 2003; Lang and Duggan, 2001). The voice of customers and other external users has, consequently, become an important factor in requirements development (Pai, 2002; Ravichandran and Rai, 1999; Ravichandran and Rai, 2000; Zultner, 1993). Several researchers have highlighted the problem of identifying and reaching external users (Hirschheim and Newman, 1991; Keil and Carmel, 1995). In addition, Salaway (1987) argue that it is more problematic to communicate with external than with

internal users. While users typically do not understand the requirements of software applications (Lyytinen, 1988; Walz et al., 1993; Watson and Frolick, 1993), these factors increase the risks related to making requirements readily accessible.

The identified literature accentuates **requirements volatility** as another key risk in software development (Davis, 1982; Fazlollahi and Tanniru, 1991; Lyytinen, 1987). Requirements volatility relates to the stability of requirements (Daft and Macintosh, 1981; Davis, 1982; Mathiassen et al., 1995); high risk indicates that requirements change easily (see Table 3). Such dynamics occur when stakeholders learn more about the software during development or when internal or external conditions for using the software change. Curtis et al. (1992) identified several sources of volatility (e.g., market impacts, company impacts, and hidden impacts) in their classic study. They noticed that the requirements process often is dialectic in nature so that when designers list requirements, these reveal new possibilities for the customer, thus feeding a new discovery cycle. An additional source of volatility risk is that user needs seldom are evident to developers (Houston et al., 2001; Kraut and Streeter, 1995; Nidumolu, 1995; Willcocks and Margetts, 1994). Mills (1999) further reminds us that software evolves over time and requirements therefore inevitably change.

Finally, the identified literature points out **requirements complexity** as a key risk in requirements and software development (Brooks, 1987; Fazlollahi and Tanniru, 1991). Requirements complexity is a measure of how easy it is to understand requirements (Mathiassen et al., 1995); high risk indicates that requirements are difficult to understand, specify, and communicate (see Table 3). Brooks (1987) argues that software is inherently complex to understand and describe. Digital computers are themselves more complex than most other human artifacts, and software has order-of-magnitude more states than computers. Boehm and Ross (1989) identified additional sources of complexity based on the varying, and often conflicting, views implied by different stakeholders in the development process.

We can summarize the rationale for this conception of requirements development risks as follows. Initially, Davis' model (1982) was based on a one-dimensional understanding of uncertainty risks; this conception is rooted in the classical notion of task uncertainty (Galbraith, 1973). Fazlollahi and Tanniru's model (1991) and the model by Mathiassen et al. (1995) developed this further into a two dimensional understanding by adding equivocality and complexity risks respectively; this conception is rooted in more elaborate contingency models that extend the classical notion of task uncertainty. Our proposed model adds identity risks as a third dimension; this conception is rooted in recent developments in requirements development practice. The simplicity resulting from only identifying three risk categories makes it feasible to synthesize the analysis into a risk-strategy model (Iversen et al., 2004). Hence, our analysis of requirements development risks leads to the following response to the first research question (Q1):

> **Risk Proposition**: requirements development risks can be characterized as identity, volatility, and complexity risks.

Requirements Development Techniques

The response to complex requirements was historically the initial focus of the software discipline (Larman and Basili, 2003), and it led to numerous documentation-centric specification techniques that are based on abstraction and textual or graphic representations (Byrd et al., 1992, Davis, 1982, Hevner and Mills, 1995, Mathiassen et al., 1995, Vessey and Conger, 1994). Beginning in the late seventies (Basili and Turner, 1975), the software discipline started to increasingly develop iterative approaches (Larman and Basili, 2003). The traditional focus on complex requirements was hence complemented with a focus on volatile requirements and an emphasis on various forms of experimentation (Boehm, 1988, Brooks, 1987, Davis, 1982, Lyytinen, 1987, Mathiassen et al., 1995, Ramamoorthy and Tsai, 1996, Zmud, 1980). Experimental techniques usually revolved around the design of the new software artifact.

In the identified literature, we find two additional trends in requirements development techniques. First, there is a focus on applying user-centric discovery techniques to identify or predict user needs (Byrd et al., 1992; Davis, 1982; Jenkins et al., 1984). These techniques emphasize communication between stakeholders (Curtis et al., 1992; Curtis et al., 1988; Davidson, 2002; Keil and Carmel, 1995) and involvement of different stakeholder groups in the design process (Bostrom, 1977; Bostrom, 1989; Elboushi and Sherif, 1997). Second, there is a considerable emphasis on various forms of prioritization techniques to select among and assess the importance of identified requirements (Byrd et al., 1992; Davis, 1982; Duggan, 2003; Ravichandran and Rai, 1999; Ravichandran and Rai, 2000; Stallinger and Grunbacher, 2001). These techniques range from simple (Byrd et al., 1992; Maiden and Hare, 1998) to rather elaborate (Duggan, 2003; Frolick and Robichaux, 1995; Ravichandran and Rai, 1999; Ravichandran and Rai, 2000; Stallinger and Grunbacher, 2001; Zultner, 1993) approaches to support decision-making in requirements development.

This analysis of the identified literature suggests extending the two-dimensional understanding of requirements development techniques in the available models (Davis, 1982; Fazlollahi and Tanniru, 1991) with requirements discovery and

Journal of the Association for Information Systems

requirements prioritization techniques as a reflection of recent trends in requirements development practice. The resulting classification of requirements development techniques is elaborated below and summarized in Table 4. We have applied the proposed categories to classify the 85 requirements development techniques found in the reviewed literature (see Appendix III for details). Below, we discuss the proposed categories of requirements development techniques.

Table 4 Requirements Development Techniques	
Techniques	**Definition**
Requirements discovery	Customer-centric and based on identification or prediction of customer needs
Requirements prioritization	Resource-centric and based on analysis of and choice between identified requirements
Requirements experimentation	Software-centric and based on iterative processes involving end-users
Requirements specification	Documentation-centric and based on abstraction and textual or graphic representations

The identified literature offers several requirements **discovery techniques** (Arthur and Gröner, 2005; Byrd et al., 1992; Davis, 1982; Halling et al., 2003; Jenkins et al., 1984). These techniques are user-centric and focus on would-be users, either internally or externally (see Table 4). Discovery techniques facilitate explication and identification of requirements, but take less rigid approaches to documentation and do not necessarily rely on software artifacts as communication tools. Discovery techniques emphasize the initial identification and appreciation of emerging requirements. Hence, they resemble forecasting or prediction techniques. Some techniques focus on understanding or predicting the beliefs and needs of individual users or stakeholders (Browne and Ramesh, 2002; Browne and Rogich, 2001; Byrd et al., 1992). Many of these techniques have been inspired by approaches in marketing, like quality function deployment (Pai, 2002; Ravichandran and Rai, 1999; Ravichandran and Rai, 2000; Zultner, 1993), Delphi (Davis, 1982) and laddering (Browne and Ramesh, 2002; Browne and Rogich, 2001; Davidson, 2002). The main goal of these techniques is to rely on personal contact between developers and potential users. Typical examples are use case-based (Antón et al., 2001) requirements development or interviews (Byrd et al., 1992; Wetherbe, 1991). Other discovery techniques take advantage of group dynamics to discover requirements through interaction with groups of users or stakeholders. Examples of these techniques include focus group interviews (Keil and Carmel, 1995; Leifera et al., 1994; Telem, 1988), group support systems (Chen and Nunamaker Jr., 1991; Dennis et al., 1988; Duggan, 2003; Duggan and Thachenkary, 2004; Halling et al., 2003; Liou and Chen, 1993; Ramesh and Sengupta, 1995), and videotaping (Jirotka and Luff, 2006).

Prioritization techniques are resource-centric, and they apply various types of analysis and decision support to help focus the development of requirements (see Table 4). The rationale for using these techniques is that software development is faced with limited resources — e.g., cost, technologies, skills, and time — which challenge projects to make decisions about which requirements to emphasize and de-emphasize. Some techniques are rather simple and straightforward to apply, such as critical success factors technique (Byrd et al., 1992), which provides a way of selecting the most important requirements from the top management's point of view. A similarly easy to use technique is card sorting (Byrd et al., 1992; Maiden and Hare, 1998), where developers categorize requirements in relation to the problem domains. Yet another technique is business process planning information (Davis, 1982), which uses the derived business objectives and business processes to come up with logically related categories of requirements. Based on the outcome of this analysis, it is possible to establish priorities in connection to the proposed architecture. Other techniques are more elaborate and require higher skill levels, for example the strategic business objective technique (Frolick and Robichaux, 1995) proposes using group support systems to assist in prioritizing requirements. Another group support system approach is EasyWinWin (Stallinger and Grunbacher, 2001). The Delphi method (Davis, 1982) is also related to these more elaborate techniques. This approach, however, is usually done off-line through asynchronous data collection from the participants, with analysis conducted after each data collection round. This is repeated until a satisfactory level of consensus is reached. The nominal group technique (Duggan, 2003, Duggan and Thachenkary, 2004) uses the focus group setting and timeframe to do the same, with participants independently rating and ranking requirements by voting on and pooling of individual rankings without using group support system software. Finally, perhaps the most elaborate technique for prioritizing requirements is quality function deployment (Duggan, 2003, Elboushi and Sherif, 1997, Pai, 2002, Ravichandran and Rai, 1999, Ravichandran and Rai, 2000, Zultner, 1993), which uses detailed matrices to represent priorities, how these could be met, and who in the development team will be responsible for them. A modified version of the quality function deployment prioritization matrix is part of the contextual design technique (Holtzblatt, 1995, Jones et al., 1993, Kujala, 2003).

Experimentation techniques are software-centric, and they employ designs of the software artifact as a key means for communication with users (see Table 4). Hence, it is essential that there is some representation of the software available. These techniques differ in their level of user interaction. Some experimentation techniques are driven mainly by user reactions. They facilitate learning by confronting users or other stakeholders with specifications, prototypes, and preliminary versions of software modules. Prototyping of software applications and user interfaces can help developers receive direct feedback from users (Davis, 1982, Keil and Carmel, 1995, Lyytinen, 1987, Ravid and Berry, 2000, Sutton, 2000, Watson and Frolick, 1993). These techniques can also help stabilize requirements by having users explain or demonstrate their work process in context, like in contextual design (Holtzblatt, 1995, Jones et al., 1993). Other experimentation techniques involve users actively in the development process (Chen and Nunamaker Jr., 1991, Darke and Shanks, 1997, Liou and Chen, 1993). The objective of these techniques is to have user knowledge and experience shape requirements development activities. Experimentation techniques that help users and developers jointly solve problems and debate requirements through various forms of structured workshops have been widely used by practitioners (Baskerville et al., 2001, Blackburn et al., 1996).

Finally, the identified literature advocates requirements **specification techniques** as a way of resolving complexity risks in requirements and software development (Bowen and Hinchey, 1995, Byrd et al., 1992, Curtis et al., 1992, Davis, 1982, Hevner and Mills, 1993, Hevner and Mills, 1995, van Lamsweerde and Letier, 2000, Vessey and Conger, 1994). These techniques are documentation-centric, and their distinguishing feature is to provide an explicit and agreed upon basis for continued development (see Table 4). Some of these techniques are highly formalized based on rigorously defined concepts and notation schemes with precise semantics (Bowen and Hinchey, 1995, Byrd et al., 1992, García-Duque et al., 2006, Hevner and Mills, 1993, van Lamsweerde and Letier, 2000). Box structures offer one such formal approach to represent requirements with execution semantics that allow for simulation of specifications (Hevner and Mills, 1995). Other examples are KAOS (van Lamsweerde and Letier, 2000) and Z (Liu et al., 1998). Specification techniques can also be more informal with less rigid ways of documenting requirements and relying primarily on the use of natural language (Bowen and Hinchey, 1995, Curtis et al., 1992). However, many informal techniques are augmented with formalized concepts and notations. These techniques focus either on acquiring information from users, on studying existing software, or on developing graphical representations of requirements, and they adopt natural language as the basic means for defining semantics. Prominent examples of informal techniques are entity-relationship modeling (Haumer et al., 1998, Pedersen et al., 2001) and data flow diagramming (Atkinson, 2000, Larsen and Naumann, 1992, Marakas and Elam, 1998, Ramesh and Browne, 1999).

There are several techniques that do not naturally fall into a single category of techniques. [2] CREWS (Haumer et al., 1998) and other scenario-based techniques (Breitman et al., 2005, Haumer et al., 1998, Jarke et al., 1998, Rolland et al., 1998, Saiedian et al., 2005, Shin et al., 2005) are prime examples of this. They provide means for both specification and discovery. Other similar techniques are Attributed Goal-Oriented Analysis (AGORA) (Kaiya et al., 2005) and the Inquiry Cycle Model (Potts et al., 1994).

In summary, based on the identified literature and in response to the second research question (Q2), we propose the following classification of requirements development techniques:

> **Resolution Proposition**: requirements development techniques can be characterized as discovery, prioritization, experimentation, and specification techniques.

Applying Techniques to Risks

Applying requirements techniques to resolve risks is an important issue (Ramamoorthy and Tsai, 1996), and the identified literature offers several suggestions. For example, Fitzgerald (1996) distinguished between levels of analysis, Trammel et al. (1996) noted that projects should be incremental to ensure continuous user feedback from each new version of the software, and other researchers recommended stepwise refinement of software solutions (Drehmer and Dekleva, 2001; Hausler et al., 1994).

Our focus is on how different techniques should be prioritized and adopted during the project life-cycle to resolve risks (Bersoff and Davis, 1991; Nguyen and Swatman, 2003). There is general agreement in the identified literature that projects seldom rely on a single technique (Chatzoglou and Macaulay, 1996; Davis, 1982). Instead, projects adopt a mixture of techniques in response to the context they face (Watson and Frolick, 1993). Moreover, the use of each technique is often adapted to the context (Basili and Rombach, 1988; Ropponen and Lyytinen, 1997; Ropponen and Lyytinen, 2000). Many writers cite the spiral model (Boehm, 1988; Iivari, 1990) for the way it applies different techniques through a sequence of

[2] We thank the anonymous reviewer for this valuable comment

Journal of the Association for Information Systems

iterative cycles (Apte et al., 1990; Bersoff and Davis, 1991; Lyytinen, 1987;Lyytinen, 1988; Lyytinen et al., 1998; Mathiassen et al., 1995; Nguyen and Swatman, 2003; Ropponen and Lyytinen, 1997; Ropponen and Lyytinen, 2000). Later discussions of the model exemplify important principles for resolving requirements development risks: to use several techniques (e.g., prototyping and specification); to give priority to certain issues over others as the life-cycle evolves (e.g., first focus on reducing risks, then focus on constructing software); and to adapt the approach to the specific development context (e.g., the number of iteration cycles depend on the context).

Generally, the identified literature suggests initially identifying and connecting to users in order to discover requirements (Duggan and Thachenkary, 2004; Elboushi and Sherif, 1997; Frolick and Robichaux, 1995) and possibly involving users in the development effort (Kujala, 2003). This approach bridges the communication gap and makes it possible to identify requirements based on the voice of users and other stakeholders (Curtis et al., 1992; Curtis et al., 1988; Davidson, 2002; Keil and Carmel, 1995; Pai, 2002; Ravichandran and Rai, 1999; Ravichandran and Rai, 2000; Zultner, 1993). From a strong initial position in which users are connected and the context of use is appreciated, the identified literature suggests an increasing focus on explicating and validating requirements through various forms of experimentation (Byrd et al., 1992; Keil and Carmel, 1995; Leifera et al., 1994; Ravid and Berry, 2000; Salaway, 1987; Sawyer and Guinan, 1998). Subsequently, as requirements stabilize, the identified literature suggests an increasing focus on detailing and specifying requirements as a basis for constructing the software (Breitman et al., 2005; Cysneiros and Leite, 2004; Darke and Shanks, 1997; Franch and Carvallo, 2003; Haumer et al., 1998; Hevner and Mills, 1995; Lee et al., 1998; van Lamsweerde and Letier, 2000). Finally, when requirements are discovered, and throughout the development effort, the identified literature suggests prioritizing requirements in response to resource constraints (Byrd et al., 1992; Davis, 1982; Duggan, 2003; Ravichandran and Rai, 1999; Ravichandran and Rai, 2000; Stallinger and Grunbacher, 2001). Such efforts help focus the effort and facilitate its convergence toward a feasible solution.

Hence, while all types of techniques can be used at any point, and different types of techniques can be combined, the identified literature suggests putting initial *primary* emphasis on requirements identity risks and discovery techniques (Browne and Ramesh, 2002; Browne and Rogich, 2001; Duggan and Thachenkary, 2004; Holtzblatt, 1995; Jones et al., 1993; Nunamaker et al., 1991; Ravichandran and Rai, 1999; Ravichandran and Rai, 2000; Stallinger and Grunbacher, 2001), subsequent *primary* emphasis on volatility risks and experimentation techniques, and final *primary* emphasis on complexity risks and specification techniques (Apte et al., 1990; Bersoff and Davis, 1991; Boehm, 1988; Lyytinen, 1987; Lyytinen et al., 1998; Mathiassen et al., 1995; Nguyen and Swatman, 2003). In addition, the identified literature suggests applying prioritization techniques once requirements have been identified, i.e., when identity risks have been appropriately resolved (Hickey and Davis, 2004; Peffers et al., 2003).

These heuristics are summarized in the following response to the third research question (Q3):

> **Priority Proposition:** the following evolution in primary emphasis over the project life-cycle is effective: 1) identity risks combined with discovery techniques; 2) volatility risks combined with prioritization and experimentation techniques; 3) complexity risks combined with prioritization and specification techniques.

In addition, the identified literature stresses the importance of managing the interaction between different techniques (Lyytinen et al., 1998; Mathiassen et al., 1995). Interaction occurs when adoption of a specific technique influences other types of risks than it was intended to resolve. For example, a developer might ask users to prioritize identified requirements to reduce complexity risks; this might, however, trigger discussions amongst users leading to new requirements and changes in existing requirements, hence increasing volatility risks. Because requirements development risks interact, the identified literature recommends that risk profiles should be continuously assessed and managed (Chen and Chou, 1999; Lyytinen et al., 1996; McFarlan, 1982; Quintas, 1994). Risk management, if practiced in this way, involves continuous sense-and-respond activities in which risk profiles are updated and the portfolio of adopted techniques is modified or changed (Lyytinen et al., 1996). Also, it involves addressing risks systemically because adoption of certain techniques might require adoption of complementary techniques to address adverse effects.

These heuristics are summarized in the following response to the third research question (Q3):

> **Interaction Proposition**: adoption of a requirements development technique can require compensating actions to reduce the adverse effect on risks other than the ones targeted by the technique.

Synthesis

With this section, we seek to synthesize the analysis of the identified literature into a proposal for an integrative contingency model for requirements development. Contingency theory and, more specifically, risk management (cf. Table 2) served as

the basis for the analysis of the literature. We identified identity, volatility, and complexity risks as a simple, yet comprehensive way to understand contextual factors (Table 3). Similarly, we identified discovery, prioritization, experimentation, and specification as ways to understand technological options (Table 4). Finally, we identified the Priority Proposition and the Interaction Proposition as key heuristics for how to effectively apply requirements development techniques to specific contexts. Further support for synthesizing the findings into an integrative contingency model can be found in the contingency literature and in existing models for managing risks in information systems development (Barley, 1990; Iivari, 1992; Lyytinen et al., 1996).

A Systems Approach to Fit

Iivari (1992) identified three types of fit between context and technology in contingency models: namely, selection, interaction, and systems approaches. If we translate these into risk management of requirements development, the selection approach suggests that requirements development risks determine which technique to adopt. Context is considered as given, and techniques are adopted through managerial selection. The interaction approach suggests that the challenge is to design appropriate relationships between risks and techniques. A design influences not only which technique to adopt in response to a specific risk, but also the way in which techniques interact with and shape a project's risk profile. The focus is, however, still on optimizing the application of individual techniques to specific risks. Finally, the systems approach suggests that the challenge is to create overall consistency between multiple requirements development techniques, requirements development risks, and the resulting performance characteristics.

The unidirectional causality implied by the selection approach is simplistic (Iivari, 1992), and it contradicts the dynamics implied by the Priority and Interaction Propositions. The interaction and systems approaches offer more comprehensive views of the relationship between techniques and risks consistent with the findings from the identified literature and our view of requirements development. While the interaction approach views causality dialectically, its focus on specific pairs of risks and resolution techniques can lead to unintended sub-optimizations, and it is not consistent with the insights underlying the Interaction Proposition. For these reasons, we choose the systems approach as the basis for synthesizing a model that links patterns of risk resolution to archetypical risk profiles. Assuming there is no one best way to approach requirements development in a given context, we therefore propose to combine patterns of risk resolution to achieve an internal consistency or harmony, as well as a basic consistency with the risks that a project faces (Minzberg, 1983, pp. 2-3). This choice of a systems approach to fit is consistent with our proposal of a risk-strategy model (Iversen et al., 2004) to support requirements development.

The proposed systems approach to fit is illustrated in Figure 1. First, adoption of a systems approach to fit involves consideration of performance (Iivari, 1992). We suggest the following as overall measures of success: the cost of requirements development is appropriate and leads to requirements that are complete, understood, and represent user or market needs. These measures focus on requirements development rather than more broadly on software development; they take into account the need to consider limited resources available for requirements development; and they reflect our emphasis on identity, volatility, and complexity risks. Second, we suggest managing requirements risks continuously over the development life-cycle. Specifically, we suggest that risk assessments be conducted iteratively with appropriate intervals; the *a priori* risk profile at time T then represents contextual factors and guides management decisions on how to move forward; in contrast, the *a posteriori* risk profile at time T+1 offers performance indicators of emerging success or failure. This approach is in line with Lyytinen et al.'s (1998) general notion of continuous (as opposed to discrete) risk management and with Iivari's iterative meta-model of information systems development (Iivari, 1990). Finally, we suggest applying patterns of resolution techniques to fit specific risk profiles as summarized in Table 5 and Figure 2. The detailed arguments for the proposed approach are presented below based on the literature analysis.

Journal of the Association for Information Systems

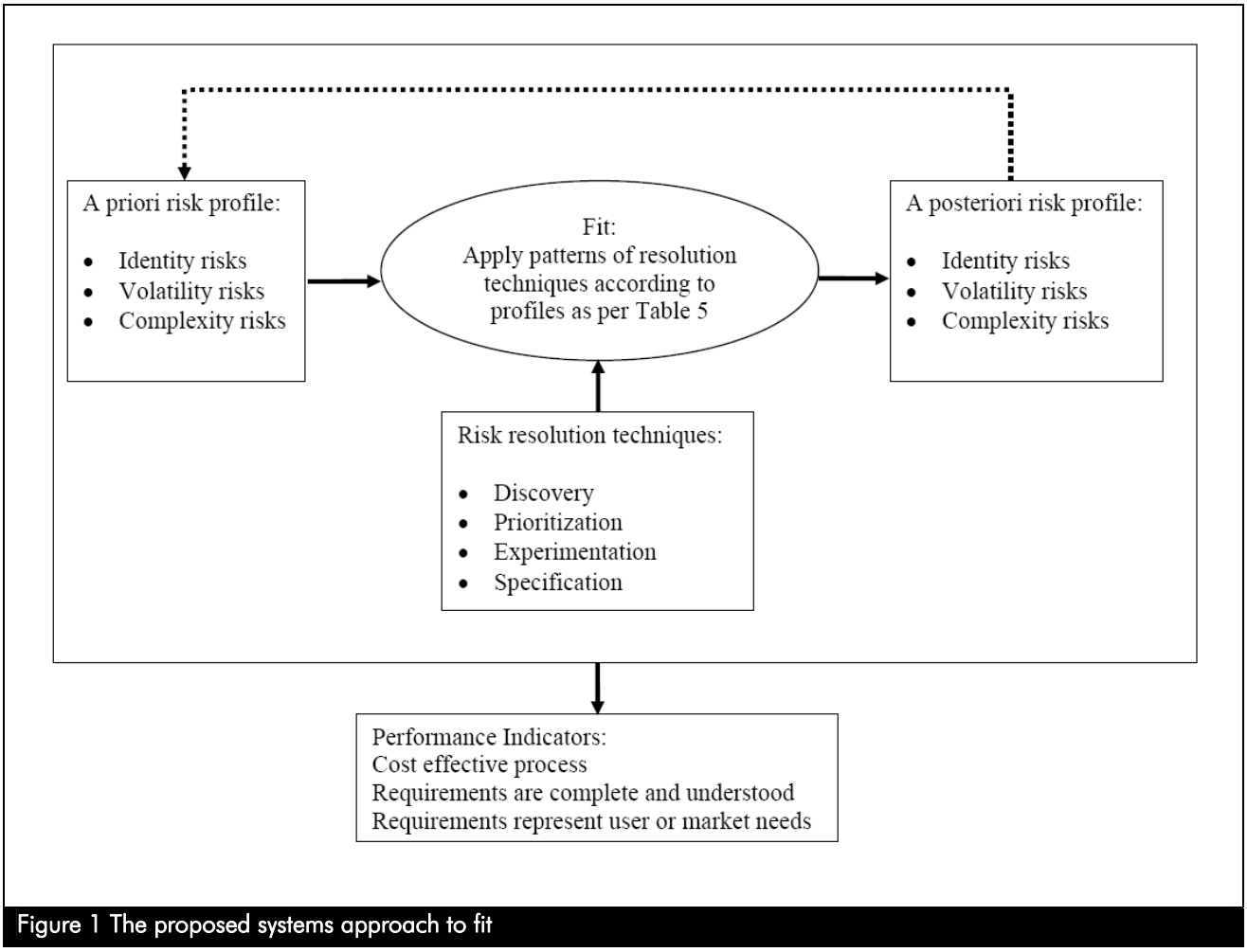

Figure 1 The proposed systems approach to fit

Contingency Model Development

McFarlan (1982) provided an example of an integrative risk-strategy model for software development; other risk-strategy models were proposed by Barki et al. (2001), Donaldson et al. (2001), and Keil et al. (1998). McFarlan's model (1982) distinguished between three types of software development risks (size of project, experience with technology, and understanding of task); suggesting an assessment of each risk using a high-low scale; and emphasizing four types of techniques to resolve risks (external integration, internal integration, project planning, and project control) based on a high-medium-low scale. As a consequence, the model suggested $2^3=8$ archetypical risk profiles and proposed for each of them a particular pattern of emphasis on resolution techniques. The model can be used repeatedly over the project life-cycle as the risk profile of a project changes. Our proposed model uses McFarlan's model (1982) as a template.

Table 5 Linking risk profiles to resolution patterns. High-Low on the left indicates risk level. High-Medium-Low on the right indicates technique emphasis.

Profile	Identity	Volatility	Complexity	Discovery	Prioritization	Experimentation	Specification
1	High	High	High	High	Low	Medium	Medium
2	Low	High	High	Low	High	High	Medium
3	High	Low	High	High	Low	Low	Medium
4	High	High	Low	High	Low	Medium	Low
5	Low	Low	High	Low	High	Low	High
6	Low	High	Low	Low	High	High	Low
7	High	Low	Low	High	Low	Low	Low
8	Low	Low	Low	Low	Medium	Low	Low

Adopting a high-low scale for assessing the level of identity, volatility, and complexity risks leads to eight archetypical risk profiles. Table 5 summarizes these and our suggestions for relating them to different emphases on requirements development techniques (adopting a high-medium-low scale). First, the basic relationship between risks in the Risk Proposition and techniques in the Resolution Proposition suggests: identity risks are resolved primarily by using discovery techniques, volatility risks primarily by combining prioritization and experimentation techniques, and complexity risks primarily by combining prioritization and specification techniques. Second, as each risk profile is characterized by identity-volatility-complexity risks (HI=high; LO=low), the Priority Proposition suggests the following heuristics:

> If only identity risks are high, put high emphasis on discovery techniques.
>
> If only volatility risks are high, put high emphasis on prioritization and experimentation techniques.
>
> If only complexity risks are high, put high emphasis on prioritization and specification techniques.
>
> If two or more risk items are high, follow the Priority Proposition.

Finally, the Interaction Proposition suggests how techniques can be emphasized further to avoid the adverse effects of one-sidedly emphasizing one particular technique, thus putting medium emphasis on experimentation techniques in profiles 1 and 4 and on specification techniques in profiles 1, 2 and 3.

As a consequence, we suggest that a project will transition between different archetypical risk profiles as risks are resolved by applying resolution patterns for the identified risk profiles according to Table 5. This is illustrated in Figure 2. In the following, we review each of the archetypical risk profiles and apply the propositions to explain in detail the rationale for the proposed contingency model as it is summarized in Table 5 and Figure 2.

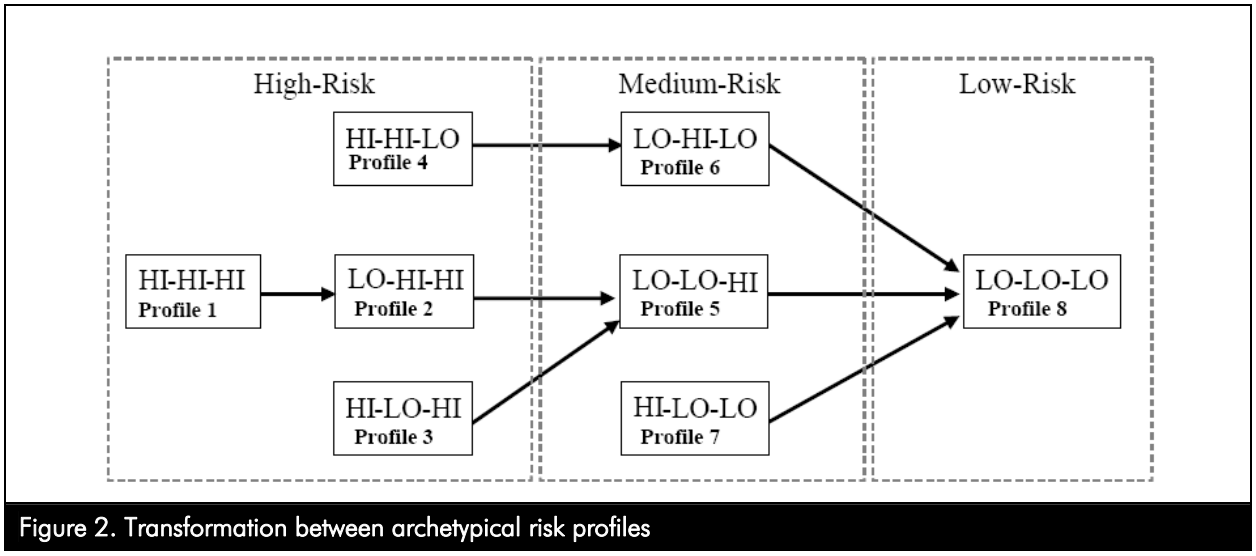

Figure 2. Transformation between archetypical risk profiles

High-risk profiles cover profiles 1-4, and they include at least two high-risk items. All three risks are high in **profile 1** (HI-HI-HI). Based on the Risk, Resolution, and Priority Propositions, the model therefore suggests first focusing primarily on requirements discovery to ensure strong connections to would-be-users and the context in which they operate. At the same time, the Interaction Proposition suggests that these projects also adopt experimentation and specification techniques from the outset to help capture and assess requirements as they are discovered. It is, however, important that these complementary techniques are more lightly emphasized initially to avoid creating barriers to effective discovery of requirements.

Profile 2 can be the result of successful use of discovery techniques (cf. profile 1) or an independent starting point for a project. In this situation, requirements development risks are assessed as LO-HI-HI. Hence, the project is well connected to would-be-users and the context in which they operate. According to the Priority Proposition, the model suggests that the initial focus primarily is on prioritization and experimentation techniques to focus and stabilize requirements. Also, the Interaction Proposition suggests that the project more lightly adopt complementary specification tactics to document requirements as they are validated.

Journal of the Association for Information Systems

In **profile 3,** the requirements development risks are assessed as HI-LO-HI. According to the Priority Proposition, the model therefore suggests a primary focus on requirements discovery. Applying the Interaction Proposition, complementary specification techniques are needed to help capture requirements as they are discovered, whereas experimentation techniques are not needed, as requirements are stable rather than volatile.

In **profile 4** (HI-HI-LO), the project faces relatively simple requirements, but there are serious identity and volatility risks. Hence, according to the Priority Proposition, the model suggests a primary focus on discovery techniques to identify or predict would-be-user needs and appreciate the context in which they operate. Also, the Interaction Proposition suggests giving experimentation techniques medium priority from the outset to help assess requirements as they are discovered; because requirements are relatively simple, specification techniques are only given low priority.

Medium-risk profiles cover profiles 5-7, which all have one high-risk item. The model suggests focusing development efforts primarily on resolving this risk first.

Profile 5 (LO-LO-HI) can be the result of successful application of risk resolution (cf. profiles 1, 2 and 3) or an independent starting point for a project. These projects face a complex set of requirements. However, the requirements reflect business and user needs, and they remain relatively stable over the project life-cycle. The Risk and Resolution Propositions suggest in these situations to mainly focus on prioritization and specification techniques.

Profile 6 (LO-HI-LO) can be the result of successful risk resolution (cf. profile 4) or an independent starting point for a project. These projects have access to relevant requirements, but the requirements are highly volatile. According to the Risk and Resolution Propositions the model suggests emphasizing prioritization and experimentation techniques to focus and stabilize requirements.

Profile 7 (HI-LO-LO), in turn, describes a project in which it is difficult to identify requirements, but the requirements are expected to be rather stable and simple. The Risk and Resolution Proposition suggests emphasizing discovery techniques to identify requirements.

Finally, **low-risk** profiles are reflected in **profile 8** (LO-LO-LO). This profile can result from successful application of our contingency model for software requirements development (cf. profiles 1-7) or an intriguing starting point for software developers where requirements are available, stable, and relatively simple. The development team understands requirements well and knows from previous experience how to design and develop software that meets the requirements. In these situations, the model suggests putting medium emphasis on prioritization techniques to help focus the project and make it quickly converge toward a solution; straightforward approaches can be adopted to develop the software.

Discussion

Having analyzed the literature and synthesized the findings into an integrative contingency model for requirements development, we discuss the contribution of this study and its implications for research and practice.

Research Contribution

Most literature on requirements development focuses on solving particular problems, and only a handful of articles discuss how to resolve risks (Byrd et al., 1992; Chatzoglou and Macaulay, 1996; Davis, 1982; Fazlollahi and Tanniru, 1991; Keil and Carmel, 1995). The major contribution of this study is proposing an up-to-date, integrative contingency model for requirements development that takes into account key risks and that can serve as the foundation for future research and for developing guidance for software practices.

First, we categorize requirements development risks into identity, volatility, and complexity risks as summarized in Table 3 and the Risk Proposition. Only two of the reviewed articles address requirements development risks directly (Davis, 1982; Fazlollahi and Tanniru, 1991), and the latter represents a revision of Davis' model. Our classification adopts volatility risks in line with uncertainty risks in these models; it adopts complexity risks (Mathiassen et al., 1995) in line with equivocality risks in Fazlollahi and Tanniru's model (1991); and it adopts identity risks as a reflection of the difficulties involved in developing software applications for internal and external users (Coughlan and Macredie, 2002; Damian and Zowghi, 2003; Lang and Duggan, 2001; Lausen and Vium, 2005; Mich et al., 2005).

Second, we categorize requirements development techniques into discovery, prioritization, experimentation, and specification techniques. We define each type of technique as summarized in Table 4 and the Resolution Proposition. Surveys of requirements development techniques have been conducted every ten years or so (Byrd et al., 1992, Davis, 1982). Appendix III provides a survey of requirements development techniques based on the proposed classification.

Third, the proposed classification of requirements techniques is mapped through heuristics to the proposed classification of risk sources. This mapping is summarized in the Priority and Interaction Propositions and in Figure 2 and Table 5. Former risk-strategy analysis models (e.g., Davis (1982) and Fazlollahi and Tanniru (1991)) do not offer similar mappings of techniques to risks. The contribution of our research is in this respect twofold: 1) the explication of the Priority and Interaction Propositions for how to apply requirement techniques to risks; and 2) the proposed risk-strategy model with easy-to-use knowledge on how to link patterns of risk resolution to archetypical risk profiles. The proposed heuristics express a dynamic view of the relation between requirements development techniques and risks (see Figure 1), and they rely on human action and decision making as determining factors for project outcomes (Barley, 1990, Lyytinen et al., 1998). The heuristics can be seen as a response to Hickey and Davis's (2004) call for research to help identify and apply appropriate requirements development techniques in specific contexts.

Implications for Research

The proposed model is hypothetical in nature and not fully elaborated. It is, at this stage, purely based on our literature review and not empirically founded. We therefore encourage researchers to: 1) further develop and validate the proposed contingency model to support its practical use; 2) assess available techniques with respect to their usefulness in different types of requirements development contexts; and, 3) develop specific measures for identity, volatility, and complexity risks in requirements development.

Thus, the first research challenge is to validate the proposed contingency model as a practical tool for managing requirements development risks (see Figure 1). This calls for empirical work on applying the proposed risk management model to real world contexts under different conditions and studying how patterns of risk resolution apply to archetypical risk profiles as expressed in Table 5 and Figure 2. These efforts could investigate requirements development practices based on a variety of research approaches: surveys of how practitioners apply requirements development techniques (Blackburn et al., 1996; Chatzoglou and Macaulay, 1996; Rai and Al-Hindi, 2000); case studies of the relationship between practices and techniques, of how and why techniques are adopted and combined, and of the effects that techniques have on resolving risks (Browne and Rogich, 2001; Darke and Shanks, 1997; Elboushi and Sherif, 1997; Haumer et al., 1998; Kujala, 2003; Liu et al., 1998); and finally, action research similar to Iversen et al. (2004) to develop, apply, modify, and validate the proposed model in real-world business contexts.

The second research challenge is to assess available techniques to provide guidance on how to most effectively apply them to specific contexts (Hickey and Davis, 2004). Such insights can guide practical requirements development as well as continued efforts to develop a better and more comprehensive portfolio of techniques. This research should critically contrast the espoused benefits and actual effects of using different techniques, and it should differentiate techniques based on their ability to resolve identity, volatility, and complexity risks as suggested in Appendix III.

The final research challenge could start out by projecting available measures (e.g., Barki et al., 1993; Lyytinen et al., 1998; Davis, 1982) into the requirements development space and updating them to reflect today's practices. The goal should be to develop valid and reliable measures to identify and assess identity, volatility, and complexity risks in requirements development projects. One approach would be to identify a generic set of measures across all types of projects and software. Another approach would be to categorize types of software (e.g., custom versus package) (Keil and Carmel, 1995) or types of projects (e.g., in-house or outsourced) to develop specialized risk measures.

Implications for Practice

While our review is limited to the academic literature on requirements development, the results suggest best practices for development of software applications that distinguish between different requirements development contexts. A contingency model, like the one proposed here, suggests how to do so. To manage risks, the model suggests that developers assess the risk profile with appropriate intervals over the project life-cycle. For each assessment they then apply appropriate resolution patterns (as summarized in Table 5) to address the risks they face and they translate these patterns into action by applying specific requirements techniques. To do so, they critically review the techniques they are currently using while exploring alternative and complementary techniques (e.g., using Appendix III). Finally, as suggested by Figure 2, developers continue reassessing risks and adjusting resolution strategies to help apply available resources to develop requirements that are complete, understood, and representative of user or market needs (see Figure 1).

Limitations

This research has its own shortcomings. Most importantly, we have limited ourselves to analyzing and synthesizing a specific selection of scientific articles published in journals. We have not searched broadly for articles in the considered journals using keywords such as 'information systems development' and 'end-user computing'; articles focusing on these issues were

only included if they explicitly emphasized requirements development issues. Also, we have not included analyses of the extensive practitioner-oriented literature on requirements development; such analyses could provide additional and valuable insights into available techniques and into espoused theories about the applicability of different techniques. Finally, following Webster and Watson (2002), we approached the literature with the ambition to extend our current knowledge about requirements development. To that end, we designed the literature analysis with the specific goal of proposing an up-to-date, integrative contingency model that links patterns of risk resolution to archetypes of risk profiles. While this approach has focused the analysis and helped develop a set of propositions and a simple, yet comprehensive, set of requirement techniques and risks, it has also given us a specific and limited perspective on the extensive knowledge that is available about requirements development.

Conclusion

This research has analyzed what we know about requirements development techniques and risks in the context of developing software applications. We developed a rigorous procedure that helped us identify 116 scholarly articles on the subject in leading journals. We adopted a research approach based on contingency theory to structure the analysis, and we synthesized the findings into a contingency model for requirements development, which links patterns of risk resolution to risk profiles. The proposed model can be used to continuously identify risk profiles and apply patterns of requirements techniques to those over the project life-cycle. The proposed model has implications for future research on requirements development risks and techniques and encourages continued efforts to link theory and practice to help practitioners effectively apply techniques to given situations.

Acknowledgements

We would like to thank the editor and the reviewers for providing very valuable and constructive feedback on earlier versions of the paper. Also many thanks to Johanna Bragge for her help with facilitate.com in developing the technique classification that is in Appendix III.

References

Alter, S. and M. Ginzberg (1978) "Managing uncertainty in MIS implementation," *Sloan Management Review* (20) 1, pp. 23-31.

Andrea, H. P. and R. W. Zmud (2002) "A Contingency Approach to Software Project Coordination," *Journal of Management Information Systems* (18) 3, pp. 41-70.

Andreou, A. S. (2003) "Promoting software quality through a human, social and organisational requirements elicitation process," *Requirements Engineering Journal* (8) 2, pp. 85-101.

Antón, A. I., R. A. Carter, A. Dagnino, J. H. Dempster et al. (2001) "Deriving Goals from a Use-Case Based Requirements Specification," *Requirements Engineering Journal* (6) 1, pp. 63-73.

Apte, U., C. S. Sankar, M. Thakur, and J. E. Turner (1990) "Reusability-Based Strategy for Development of Information Systems: Implementation Experience of a Bank," *MIS Quarterly* (14) 4, pp. 421-433.

Arthur, J. D. and M. K. Gröner (2005) "An operational model for structuring the requirements generation process," *Requirements Engineering Journal* (10) 1, pp. 45-62.

Atkinson, C. J. (2000) "Socio-Technical and Soft Approaches to Information Requirements Elicitation in the Post-Methodology Era," *Requirements Engineering Journal* (5) 2, pp. 67-73.

Barki, H., S. Rivard, and J. Talbot (1993) "Toward an Assessment of Software Development Risk," *Journal of Management Information Systems* (10) 2, pp. 203-225.

Barki, H., S. Rivard, and J. Talbot (2001) "An Integrative Contingency Model of Software Project Risk Management," *Journal of Management Information Systems* (17) 4, pp. 37-69.

Barley, S. R. (1990) "The Alignment of Technology and Structure through Roles and Networks," *Administrative Science Quarterly* (35) 1, pp. 61-103.

Basili, V. R. and H. D. Rombach (1988) "The TAME project: towards improvement-oriented software environments," *IEEE Transactions Software Engineering* (14) 6, pp. 758-773.

Basili, V. R. and A. J. Turner (1975) "Iterative Enhancement: A Practical Technique for Software Development," *IEEE Transactions Software Engineering* (1) 4, pp. 390-296.

Baskerville, R., L. Levine, J. Pries-Heje, B. Ramesh et al. (2001) "How Internet software companies negotiate quality," *Computer* (34) 5, pp. 51-57.

Bergman, M., J. L. King, and K. Lyytinen (2002) "Large Scale Requirements Analysis Revisited: The Need for Understanding the Political Ecology of Requirements Engineering," *Requirements Engineering* (7) 3, pp. 152-171.

Bersoff, E. H. and A. M. Davis (1991) "Impacts of Life-Cycle Models on Software Configuration Management," *Communications of the ACM* (34) 8, pp. 104-118.

Blackburn, J. D., G. D. Scudder, and L. N. Van Wassenhove (1996) "Improving speed and productivity of software development: A global survey of software developers," *IEEE Transactions on Software Engineering* (22) 12, pp. 875-885.
Boehm, B. (1988) "A Spiral model of software development and enhancement," *IEEE Computer* (21) 5, pp. 61-72.
Boehm, B. (1991) "Software Risk Management: Principles and Practices," *IEEE Software* (8) 1, pp. 32-41.
Boehm, B. and R. Ross (1989) "Theory-W software project management principles and examples," *IEEE Transactions Software Engineering* (15) 7, pp. 902-916.
Bostrom, R. P. (1977) "MIS Problems and Failures: A Socio-Technical Perspective PART 1:THE CAUSES," *MIS Quarterly* (1) 3, pp. 17-32.
Bostrom, R. P. (1989) "Successful application of communication techniques to improve the systems development process," *Information & Management* (16) 5, pp. 279-275.
Bowen, J. P. and M. G. Hinchey (1995) "10-Commandments of Formal Methods," *IEEE Computer* (28) 4, pp. 56-63.
Breitman, K. K., J. C. S. d. P. Leite, and D. M. Berry (2005) "Supporting scenario evolution," *Requirements Engineering Journal* (10) 2, pp. 112-131.
Brooks, F. P. (1987) "No Silver Bullet - Essence and Accidents of Software Engineering," *IEEE Computer* (20) 4, pp. 10-19.
Browne, G. J. and V. Ramesh (2002) "Improving information requirements determination: a cognitive perspective," *Information & Management* (39) 8, pp. 625-645.
Browne, G. J. and M. B. Rogich (2001) "An empirical investigation of user requirements elicitation: Comparing the effectiveness of prompting techniques," *Journal of Management Information Systems* (17) 4, pp. 223-249.
Bryant, J. (1997) "Requirements capture using SODA," *European Journal of Information Systems* (6) 3, pp. 155-163.
Burns, T. and G. M. Stalker (1994) *The management of innovation*, revised ed. edition. New York: Oxford University Press Inc.
Byrd, T. A., K. L. Cossick, and R. W. Zmud (1992) "A Synthesis of Research on Requirements Analysis and Knowledge Acquisition Techniques," *MIS Quarterly* (16) 1, pp. 117-138.
Chatzoglou, P. D. and L. A. Macaulay (1996) "Requirements capture and IS methodologies," *Information Systems Journal* (6) 3, pp. 209-225.
Chen, J. Y. J. and S. C. Chou (1999) "Consistency management in a process environment," *Journal of Systems and Software* (47) 2-3, pp. 105-110.
Chen, M. and J. F. Nunamaker Jr. (1991) "The Architecture and Design of a Collaborative Environment for Systems Definition," *DATA BASE* (22) 1-2, pp. 22-29.
Cohen, M. D., J. G. March, and J. Olsen (1972) "A garbage can theory of organizational choice," *Administrative Science Quarterly* (17) 1, pp. 1-25.
Coughlan, J. and R. D. Macredie (2002) "Effective Communication in Requirements Elicitation: A Comparison of Methodologies," *Requirements Engineering Journal* (7) 2, pp. 47-60.
Curtis, B., M. I. Kellner, and J. Over (1992) "Process modeling," *Communications of the ACM* (35) 9, pp. 75-90.
Curtis, B., H. Krasner, and N. Iscoe (1988) "A field study of the software design process for large systems," *Communications of the ACM* (31) 11, pp. 1268-1287.
Cysneiros, L. M. and J. C. S. D. Leite (2004) "Nonfunctional requirements: From elicitation to conceptual models," *IEEE Transactions on Software Engineering* (30) 5, pp. 328-350.
Daft, R.L. and R.H. Lengel (1981) "Organizational information requirements, media richness and structural design," *Management Science* (32) 5, pp. 554-571.
Damian, D. E. and D. Zowghi (2003) "RE Challenges in multi-site software development organizations," *Requirements Engineering Journal* (8) 3, pp. 149-160.
Darke, P. and G. Shanks (1997) "User viewpoint modelling: understanding and representing user viewpoints during requirements definition," *Information Systems Journal* (7) 3, pp. 213-239.
Davidson, E. J. (2002) "Technology frames and framing: A socio-cognitive investigation of requirements determination," *MIS Quarterly* (26) 4, pp. 329-358.
Davis, G. (1982) "Strategies for information requirements determination," *IBM Systems Journal* (21) 1, pp. 4-31.
Dennis, A. R., J. F. George, L. M. Jessup, J. F. Nunamaker Jr. et al. (1988) "Information Technology to Support Electronic Meetings," *MIS Quarterly* (12) 4, pp. 591-624.
Donaldson, S. E. and S. G. Siegel (2001) *Successful Software Development*. Upper Saddle River, NJ: Prentice Hall.
Drehmer, D. E. and S. M. Dekleva (2001) "A note on the evolution of software engineering practices," *Journal of Systems and Software* (57) 1, pp. 1-7.
Dubois, E. and K. Pohl (2003) "RE 02: A major step toward a mature requirements engineering community," *IEEE Software* (20) 1, pp. 14-15.
Duggan, E. W. (2003) "Generating systems requirements with facilitated group techniques," *Human-Computer Interaction* (18) 4, pp. 373-394.
Duggan, E. W. and C. S. Thachenkary (2004) "Integrating nominal group technique and joint application development for improved systems requirements determination," *Information & Management* (41) 4, pp. 399-411.

Elboushi, M. I. and J. S. Sherif (1997) "Object-oriented software design utilizing quality function deployment," *Journal of Systems and Software* (38) 2, pp. 133-143.
Fazlollahi, B. and M. R. Tanniru (1991) "Selecting a Requirement Determination Methodology-Contingency Approach Revisited," *Information & Management* (21) 5, pp. 291-303.
Fitzgerald, B. (1996) "Formalized systems development methodologies: A critical perspective," *Information Systems Journal* (6) 1, pp. 3-23.
Franch, X. and J. P. Carvallo (2003) "Using Quality Models in Software Package Selection," *IEEE Software* (20) 1, pp. 34-41.
Frolick, M. N. and B. P. Robichaux (1995) "EIS Information Requirements Determination - Using a Group Support System to Enhance the Strategic Business Objectives Method," *Decision Support Systems* (14) 2, pp. 157-170.
Galbraith, J. R. (1973) Designing Complex Organizations, 1st edition. Boston, MA: Addison-Wesley Longman Publishing Co., Inc.
García-Duque, J., M. López-Nores, J. J. Pazos-Arias, A. Fernández-Vilas et al. (2006) "Guidelines for the incremental identification of aspects in requirements specifications," *Requirements Engineering Journal* (11) 4, pp. 239-263.
Gillenson, M. and J. Stutz (1991) "Academic Issues in MIS: Journals and Books," *MIS Quarterly* (15) 4, pp. 147-452.
Glass, R. L., I. Vessey, and S. A. Conger (1992) "Software Tasks - Intellectual or Clerical," *Information & Management* (23) 4, pp. 183-191.
Halling, M., S. Biffl, and P. Grünbacher (2003) "An economic approach for improving requirements negotiation models with inspection," *Requirements Engineering Journal* (8) 4, pp. 236-247.
Hardgrave, B. and K. Walstrom (1997) "Forums for Management Information Systems Scholars," *Communications of the ACM* (38) 3, pp. 93-102.
Haumer, P., K. Pohl, and K. Weidenhaupt (1998) "Requirements elicitation and validation with real world scenes," *IEEE Transactions on Software Engineering* (24) 12, pp. 1036-1054.
Hausler, P. A., R. C. Linger, and C. J. Trammell (1994) "Adopting Cleanroom Software Engineering with a Phased Approach," *IBM Systems Journal* (33) 1, pp. 89-109.
Hevner, A. R. and H. D. Mills (1993) "Box-structured methods for systems development with objects," *IBM Systems Journal* (32) 2, pp. 232-251.
Hevner, A. R. and H. D. Mills (1995) "Box-Structured Requirements Determination Methods," *Decision Support Systems* (13) 3-4, pp. 223-239.
Hickey, A. M. and A. Davis (2004) "A Unified Model of Requirements Elicitation," *Journal of Management Information Systems* (20) 4, pp. 65–84.
Hirschheim, R. and M. Newman (1991) "Symbolism and Information Systems Development: Myth, Metaphors and Magic," *Information Systems Research* (2) 1, pp. 29-62.
Holsapple, C., L. Johnson, H. Manakyan, and J. Tanner (1994) "Business Computing Research Journals: A Normalized Citation Analysis," *Journal of Management Information Systems* (11) 1, pp. 131-140.
Holtzblatt, K. (1995) "Requirements Gathering: The Human Factor," *Communications of The ACM* (38) 5, pp. 31-33.
Houston, D. X., G. T. Mackulak, and J. S. Collofello (2001) "Stochastic simulation of risk factor potential effects for software development risk management," *Journal of Systems and Software* (59) 3, pp. 247-257.
Iivari, J. (1990) "Hierarchical Spiral Model for Information-System and Software- Development .1. Theoretical Background," *Information and Software Technology* (32) 6, pp. 386-399.
Iivari, J. (1992) "The organizational fit of information systems," *Journal of Information Systems* (2) 1, pp. 3-29.
Iivari, J., R. Hirschheim, and H. K. Klein (2000) "A Dynamic Framework for Classifying Information Systems Development Methodologies and Approaches," *Journal of Management Information Systems* (17) 3, pp. 179-218.
Iversen, J. H., L. Mathiassen, and P. A. Nielsen (2004) "Managing Risk in Software Process Improvement: An Action Research Approach," *MIS Quarterly* (28) 3, pp. 395-433.
Jarke, M., X. T. Bui, and J. M. Carroll (1998) "Scenario Management: An Interdisciplinary Approach," *Requirements Engineering Journal* (3) 3-4, pp. 155-173.
Jayaratna, N. (1994) *Understanding and Evaluating Methodologies*. London: McGraw Hill.
Jenkins, A. M., J. D. Naumann, and J. C. Wetherbe (1984) "Empirical investigation of systems development practices and results," *Information & Management* (7) 2, pp. 73-82.
Jirotka, M. and P. Luff (2006) "Supporting Requirements with Video-Based Analysis," *IEEE Software* (23) 3, pp. 42-44.
Jones, R. M., L. Candy, and E. A. Edmonds (1993) "Knowledge-Based System Requirements," *Knowledge-Based Systems* (6) 1, pp. 31-37.
Kaiya, H., D. Shinbara, J. Kawano, and S. Motoshi (2005) "Improving the detection of requirements discordances among stakeholders," *Requirements Engineering Journal* (10) 4, pp. 289-303.
Katerattanakul, P., B. Han, and S. Hong (2003) "Objective Quality Ranking of Computing Journals," *Communications of the ACM* (46) 10, pp. 111-114.
Keil, M. and E. Carmel (1995) "Customer-developer links in software development," *Communications of the ACM* (38) 5, pp. 33-44.

Keil, M., P. E. Cule, K. Lyytinen, and R. C. Schmidt (1998) "A Framework for Identifying Software Project Risks," *Communications of the ACM* (41) 11, pp. 76-83.
Kickert, W. J. M. (1983) "Research note: Research models underlying situational dependency," *Organization Studies* (4) 1, pp. 55-72.
Kraut, R. E. and L. A. Streeter (1995) "Coordination in software development," *Communications of the ACM* (38) 3, pp. 69-81.
Kujala, S. (2003) "User involvement: a review of the benefits and challenges," *Behaviour & Information Technology* (22) 1, pp. 1-16.
Lamb, R. and R. Kling (2003) "Reconceptualizing Users as Social Actors in Information Systems Research," *MIS Quarterly* (27) 2, pp. 197-235.
Lang, M. and M. Duggan (2001) "A Tool to Support Collaborative Software Requirements Management," *Requirements Engineering Journal* (6) 3, pp. 161-172.
Larman, C. and V. R. Basili (2003) "Iterative and Incremental Development: A Brief History," *IEEE Software* (36) 6, pp. 47-56.
Larsen, T. J. and J. D. Naumann (1992) "An Experimental Comparison of Abstract and Concrete Representations in Systems-Analysis," *Information & Management* (22) 1, pp. 29-40.
Lausen, S. and J. P. Vium (2005) "Communication gaps in a tender process," *Requirements Engineering Journal* (10) 4, pp. 247-261.
Lawrence, P. R. and J. W. Lorsch (1974) *Organization and Environment - Managing Differentiation and Integration*. Homewood, Illinois: Richard D. Irwin Inc.
Lee, W. J., S. D. Cha, and Y. R. Kwon (1998) "Integration and analysis of use cases using modular Petri nets in requirements engineering," *IEEE Transactions on Software Engineering* (24) 12, pp. 1115-1130.
Leifera, R., S. Leeb, and J. Durgeea (1994) "Deep structures: Real information requirements determination," *Information & Management* (27) 5, pp. 275-285.
Leite, J. and P. A. Freeman (1991) "Requirements Validation through Viewpoint Resolution," *IEEE Transactions on Software Engineering* (17) 12, pp. 1253-1269.
Liou, Y. I. and M. Chen (1993) "Using group support systems and joint application development for requirement specification," *Journal of Management Information Systems* (10) 3, pp. 25-41.
Liu, S., A. J. Offutt, C. Ho-Stuart, Y. Sun et al. (1998) "SOFL: A formal engineering methodology for industrial applications," *IEEE Transactions on Software Engineering* (24) 1, pp. 24-45.
Lyytinen, K. (1987) "Different Perspectives on Information-Systems - Problems and Solutions," *ACM Computing Surveys* (19) 1, pp. 5-46.
Lyytinen, K. (1988) "Expectation Failure Concept and Systems Analysts' View of Information System Failures: Results of an Exploratory Study," *Information & Management* (14) 1, pp. 45-56.
Lyytinen, K., L. Mathiassen, and J. Ropponen (1996) "A framework for software risk management," *Journal of Information Technology* (11) 4, pp. 275-285.
Lyytinen, K., L. Mathiassen, and J. Ropponen (1998) "Attention shaping and software risk - A categorical analysis of four classical risk management approaches," *Information Systems Research* (9) 3, pp. 233-255.
Maiden, N. A. M. and M. Hare (1998) "Problem domain categories in requirements engineering," *International Journal of Human-Computer Studies* (49) 3, pp. 281-304.
Marakas, G. M. and J. J. Elam (1998) "Semantic structuring in analyst acquisition and representation of facts in requirements analysis," *Information Systems Research* (9) 1, pp. 37-63.
Markus, L. M. and J.-Y. Mao (2004) "Participation in Development and Implementation —Updating An Old, Tired Concept for Today's IS Contexts," *Journal of AIS* (5) 11-12, pp. 514-544.
Mathiassen, L., T. Seewaldt, and J. Stage (1995) "Prototyping and Specifying: Principles and Practices of a Mixed Approach," *Scandinavian Journal of Information Systems* (7) 1, pp. 55-72.
McFarlan, F. W. (1981) "Portfolio Approach to Information Systems," *Harvard Business Review* (59) 5, pp. 142-150.
McFarlan, W. (1982) "Portfolio Approach to Information Systems," *Journal of Systems Management* (33) 1, pp. 12-19.
McKeen, J. D., T. Guimaraes, and J. C. Wetherbe (1994) "The Relationship between User Participation and User Satisfaction - an Investigation of 4 Contingency Factors," *MIS Quarterly* (18) 4, pp. 427-451.
Mich, L., C. Anesi, and D. M. Berry (2005) "Applying a pragmatics-based creativity-fostering technique to requirements elicitation," *Requirements Engineering Journal* (10) 4, pp. 262-275.
Mills, H. D. (1999) "The management of software engineering - Part I: Principles of software engineering (Reprinted from IBM Systems Journal, vol 19, 1980)," *IBM Systems Journal* (38) 2-3, pp. 289-295.
Minzberg, H. (1983) *Structure in fives: designing effective organizations*. New Jersey: Prentice-Hall International.
Montazemi, A. R. and D. W. Conrath (1986) "The Use of Cognitive Mapping for Information Requirements Analysis," *MIS Quarterly* (10) 1, pp. 44-56.
Mylonopoulos, N. and V. Theoharakis (2001) "On-Site: Global Perceptions of IS Journals," *Communications of the ACM* (44) 9, pp. 29-33.

Journal of the Association for Information Systems

Neill, C. J. and P. A. Laplante (2003) "Requirements Engineering: The State of the Practice," *IEEE Software* (20) 6, pp. 40-45.
Nguyen, L. and P. A. Swatman (2003) "Managing the requirements engineering process," *Requirements Engineering Journal* (8) 1, pp. 55-68.
Nidumolu, S. (1995) "The Effect of Coordination and Uncertainty on Software Project Performance: Residual Performance Risk as an Intervening Variable," *Information Systems Research* (6) 3, pp. 191-219.
Nunamaker, J. F., A. R. Dennis, J. S. Valacich, D. Vogel et al. (1991) "Electronic meeting systems to Support Group Work," *Communications of the ACM* (34) 7, pp. 40-61.
Nuseibeh, B., J. Kramer, and A. Finkelstein (1994) "A Framework for Expressing the Relationships Between Multiple Views in Requirements Specification," *IEEE Transactions Software Engineering* (20) 10, pp. 760-773.
Pai, W. C. (2002) "A quality-enhancing software function deployment model," *Information Systems Management* (19) 3, pp. 20-24.
Pedersen, T. B., C. S. Jensen, and C. E. Dyreson (2001) "A foundation for capturing and querying complex multidimensional data," *Information Systems* (26) 5, pp. 383-423.
Peffers, K., C. E. Gengler, and T. Tuunanen (2003) "Extending critical success factors methodology to facilitate broadly participative information systems planning," *Journal of Management Information Systems* (20) 1, pp. 51-85.
Peffers, K. and Y. Tang (2003) "Identifying and Evaluating the Universe of Outlets for Information Systems Research: Ranking the Journals," *Journal of Information Technology Theory and Application* (5) 1, pp. 63-84.
Pohl, K. (1994) "The Three Dimensions of Requirements Engineering - a Framework and Its Applications," *Information Systems* (19) 3, pp. 243-258.
Potts, C., K. Takahashi, and A. I. Antón (1994) "Inquiry-Based Requirements Analysis," *IEEE Software* (11) 2, pp. 21-32.
Pugh, D. S. (ed.) (1998) *The Aston Programme - The Aston Study and its developments, Volume I. Classic Research in Management*, Hants: Ashgate Publishing Company.
Quintas, P. (1994) "A Product-Process Model of Innovation in Software-Development," *Journal of Information Technology* (9) 1, pp. 3-17.
Rai, A. and H. Al-Hindi (2000) "The effects of development process modeling and task uncertainty on development quality performance," *Information & Management* (37) 6, pp. 335-346.
Ramamoorthy, C. V. and W. T. Tsai (1996) "Advances in software engineering," *Computer* (29) 10, pp. 47-58.
Ramesh, B. and K. Sengupta (1995) "Multimedia in a design rationale decision support system," *Decision Support Systems* (15) 3, pp. 181-196.
Ramesh, V. and G. J. Browne (1999) "Expressing casual relationships in conceptual database schemas," *Journal of Systems and Software* (45) 3, pp. 225-232.
Ravichandran, T. and A. Rai (1999) "Total quality management in information systems development: Key constructs and relationships," *Journal of Management Information Systems* (16) 3, pp. 119-155.
Ravichandran, T. and A. Rai (2000) "Quality management in systems development: An organizational system perspective," *MIS Quarterly* (24) 3, pp. 381-415.
Ravid, A. and D. M. Berry (2000) "A Method for Extracting and Stating Software Requirements that a User Interface Prototype Contains," *Requirements Engineering Journal* (5) 4, pp. 225-241.
Regnell, B., M. Höst, J. Natt och Dag, P. Beremark et al. (2001) "An Industrial Case Study on Distributed Prioritisation in Market-Driven Requirements Engineering for Packaged Software," *Requirements Engineering Journal* (6) 1, pp. 51-62.
Rolland, C., C. Souveyet, and M. B. Ayed (2003) "Guiding Lyee user requirements capture," *Knowledge-Based Systems* (16) 7-8, pp. 351-359.
Rolland, C., C. Souveyet, and C. Ben Achour (1998) "Guiding goal modeling using scenarios," *IEEE Transactions on Software Engineering* (24) 12, pp. 1055-1071.
Ropponen, J. and K. Lyytinen (1997) "Can software risk management improve system development: An exploratory study," *European Journal of Information Systems* (6) 1, pp. 41-50.
Ropponen, J. and K. Lyytinen (2000) "Components of software development risk: How to address them? A project manager survey," *IEEE Transactions on Software Engineering* (26) 2, pp. 98-112.
Saiedian, H., P. Kumarakulasingam, and M. Anan (2005) "Scenario-based requirements analysis techniques for real-time software systems: a comparative evaluation," *Requirements Engineering Journal* (10) 1, pp. 22-33.
Salaway, G. (1987) "An Organizational Learning Approach to Information Systems Development," *MIS Quarterly* (20) 1, pp. 244-264.
Sawyer, S. and P. J. Guinan (1998) "Software development: Processes and performance," *IBM Systems Journal* (37) 4, pp. 552-569.
Shin, J. E., A. G. Sutcliffe, and A. Gregoriades (2005) "Scenario advisor tool for requirements engineering," *Requirements Engineering Journal* (10) 2, pp. 132-145.
Simon, H. (1955) "A Behavioral Model of Rational Choice," *Quarterly Journal of Economics* (69) 1, pp. 99-118.
Simon, H. (1957) *Models of Man*. New York.

Stallinger, F. and P. Grunbacher (2001) "System dynamics modelling and simulation of collaborative requirements engineering," *Journal of Systems and Software* (59) 3, pp. 311-321.
Sutton, D. C. (2000) "Linguistic Problems with Requirements and Knowledge Elicitation," *Requirements Engineering Journal* (5) 2, pp. 114-124.
Telem, M. (1988) "Information Requirements Specification I: Brainstorming a Collective Decision- Making Approach," *Information Processing & Management* (24) 5, pp. 549-557.
Trammell, C. J., M. G. Pleszkoch, R. C. Linger, and A. R. Hevner (1996) "The incremental development process in Cleanroom software engineering," *Decision Support Systems* (17) 1, pp. 55-71.
Walz, D., J. Elam, and B. Curtis (1993) "Inside a software design team: Knowledge acquisistion, Sharing and Integration," *Communications of the ACM* (36) 10, pp. 62-77.
van Lamsweerde, A. and E. Letier (2000) "Handling obstacles in goal-oriented requirements engineering," *IEEE Transactions on Software Engineering* (26) 10, pp. 978-1005.
Watson, H. J. and M. N. Frolick (1993) "Determining information requirements for an EIS," *MIS Quarterly* (17) 3, pp. 255-269.
Webster, J. and R. T. Watson (2002) "Analyzing the Past to Prepare for the Future: Writing a Literature Review," *MIS Quarterly* (26) 2, pp. xiii-xx.
Weill, P. and M. H. Olson (1989) "An Assessment of the Contingency Theory of Management Information Systems," *Journal of Management Information Systems* (6) 1, pp. 59-85.
Vessey, I. and S. A. Conger (1994) "Requirements Specification: Learning Object, Process, and Data Methodologies," *Communications of the ACM* (37) 5, pp. 102-113.
Wetherbe, J. C. (1991) "Executive Information Requirements - Getting It Right," *MIS Quarterly* (15) 1, pp. 51-65.
Whitman, M., A. Hendrickson, and A. Townsend (1999) "Research Commentary. Academic Rewards for Teaching, Research and service: Data and Discourse," *Information Systems Research* (10) 2, pp. 99-109.
Willcocks, L. and H. Margetts (1994) "Risk assessment and information systems," *European Journal of Information Systems* (3) 2, pp. 127-138.
Woodward, J. (1994) *Industrial Organization*, 2nd edition. Oxford: Oxford University Press.
Zhang, P. and N. Li (2005) "The Intellectual Development of Human-Computer Interaction Research: A Critical Assessment of the MIS Literature (1990-2002)," *Journal of AIS* (6) 1, pp. 227-292.
Zmud, R. W. (1980) "Management of Large Software Development Efforts," *MIS Quarterly* (4) 2, pp. 45-55.
Zultner, R. E. (1993) "TQM for technical teams," *Communications of the ACM* (36) 10, pp. 79-91.

Appendix I

ACM Computing Surveys
ACM SIGecom Exchanges
ACM Trans. on Database Systems
ACM Trans. on Information Systems
AI Magazine
Artificial Intelligence
Australian J. of Information Systems
Behavior & Information Technology
Communications of the ACM
Communications of the AIS
Computer Journal
Computer Supported Cooperative Work
DATA BASE
Decision Support Systems
Electronic Commerce Research and Application
Electronic Markets
e-Service J.
European J. of Information Systems
Expert Systems w. Applications
Human-Computer Interaction
IBM Systems J.
IEEE Computer
IEEE Trans. on Software Engineering
Information & Management
Information and Organization
Information Processing & Management
Information Research
Information Resources Management J.
Information Systems
Information Systems Frontiers
Information Systems J.
Information Systems Management
Information Systems Research
Information Technology & People
Information Technology and Management
Informing Science
Int. J. of Human-Computer Studies
Int. J. of Electronic Commerce
Int. J. of Human Computer Study
Int. J. of Information Management
J. of Computer and System Sciences
J. of Computer Information Systems
J. of Computer IS
J. of Database Management
J. of End-User Computing
J. of Global Information Management
J. of Global Information Technology Management
J. of Information Systems Education
J. of Information Technology
J. of IT Cases & Applications
J. of Management Information Systems
J. of Strategic Information Systems
J. of Strategic IS
J. of Systems and Software
J. of the ACM
J. of the Association for Information Systems
J. of Information Technology Theory & Application
J. of Information Systems Management
J. of Information Technology
J. of Information Technology Education
J. of Management
J. of Organizational Computing and EC
Knowledge Based Systems
MIS Quarterly
MISQ Discovery
Scandinavian J. of Information Systems
The Information Society
Wirtschaftsinformatik

Appendix II

Journal		Article
Requirements Engineering Journal (20)	1.	Andreous A.S. (2003)
	2.	Antón, A.I., R.A. Carter, A. Dagnino, J.H. Dempster and D.F. Siege (2001)
	3.	Arthur J.D. and M.K. Gröner (2005)
	4.	Atkinson C.J. (2000
	5.	Breitman K.K., J.C.S. Leite and D.M. Berry (2005)
	6.	Coughlan J. and R.D. Macredie (2002)
	7.	Damian D.E. and D. Zowghi (2003)
	8.	García-Duque J., M. López-Nores, J.J. Pazos-Arias, A. Fernández-Vilas, R.P. Díaz-Redondo, A. Gil-Solla, M. Ramos-Cabrer and Y. Blanco-Fernández (2006)
	9.	Halling M., S. Biffl and P. Grünbacher (2003)
	10.	Jarke M., X.T. Bui and J.M. Carroll (1998)
	11.	Kaiya H., D. Shinbara, J. Kawano and M. Saeki (2005)
	12.	Lang M. And J. Duggan (2001)
	13.	Lausen S. and J.P. Vium (2005)
	14.	Mich L., C. Anesi and D.M. Berry (2005)
	15.	Nguyen L. and P.A. Swatman (2003)
	16.	Ravid A. And D.M. Berry (2000)
	17.	Regnell B., M. Höst, J. Natt och Dag, P. Beremark and T. Hjelm (2001)
	18.	Saiedian H., P. Kumarakulasingam and M. Anan (2005)
	19.	Shin J.E., A.G. Sutcliffe and A. Gregoriades (2005)
	20.	Sutton D.C. (2000)
IEEE Trans. On Software Engineering (12)	21.	Basili V.R. and H.D. Rombach (1988)
	22.	Blackburn J.D., G.D. Scudder and L.N. VanWassenhove (1996)
	23.	Boehm B. and R. Ross (1989)
	24.	Cysneiros, L.M. and Leite, J. (2004)
	25.	Haumer P. , K. Pohl and K. Weidenhaupt (1998)
	26.	Lee W.J., S.D. Cha and Y.R. Kwon (1998)
	27.	Leite J. and P.A. Freeman (1991)
	28.	Liu S. , A.J. Offutt C., Ho-Stuart Y. Sun and M. Ohba (1998)
	29.	Nuseibeh B. , J. Kramer and A. Finkelstein (1994)
	30.	Rolland C., C. Souveyet and C. Ben Achour (1998)
	31.	Ropponen J. and K. Lyytinen (2000)
	32.	van Lamsweerde A. and E. Letier (2000)
MIS Quarterly (11)	33.	Apte U. , C.S. Sankar M., Thakur and J.E. Turner (1990)
	34.	Bostrom R.P. (1977)
	35.	Byrd T.A., K.L. Cossick and R.W. Zmud (1992)
	36.	Davidson E.J. (2002)
	37.	Dennis A.R., J.F. George ,L.M. Jessup, J.F. Nunamaker Jr. and D.R. Vogel (1988)
	38.	Montazemi A.R. and D.W. Conrath (1986)
	39.	Ravichandran T. and A. Rai (2000)
	40.	Salaway G. (1987)
	41.	Watson H.J. and M.N. Frolich (1993)
	42.	Wetherbe J.C. (1991)
	43.	Zmud R.W. (1980)
Communications of the ACM (10)	44.	Bersoff E.H. and A.M. Davis (1991)
	45.	Curtis B., H. Krasner and N. Iscoe (1988)
	46.	Curtis B., M.I. Kellner and J. Over (1992)
	47.	Holtzblatt K. (1995)
	48.	Keil M. and E. Carmel (1995)
	49.	Kraut R.E. and L.A. Streeter (1995)
	50.	Nunamaker J.F. , A.R. Dennis, J.S. Valacich, D. Vogel and J.F. George

	(1991)
	51. Vessey I. and S.A. Conger (1994)
	52. Walz D. , J. Elam and B. Curtis (1993)
	53. Zultner R.E. (1993)
Information & Management (10)	54. Bostrom R.P. (1989)
	55. Browne G.J. and V. Ramesh (2002)
	56. Duggan E.W. and C.S. Thachenkary (2004)
	57. Fazlollahi B. and M.R. Tanniru (1991)
	58. Glass R.L., I. Vessey and S.A. Conger (1992)
	59. Jenkins A.M. , J.D. Naumann and J.C. Wetherbe (1984)
	60. Larsen T.J. and J.D. Naumann (1992)
	61. Leifera R. , S. Leeb and J. Durgeea (1994)
	62. Lyytinen K. (1988)
	63. Rai A. and H. Al-Hindi (2000)
J. of Systems and Software (6)	64. Chen J.Y.J.and S.C. Chou (1999)
	65. Drehmer D.E. and S.M. Dekleva (2001)
	66. Elboushi M.I. and J.S. Sherif (1997)
	67. Houston D.X., G.T. Mackulak and J.S. Collofello (2001)
	68. Ramesh V. and G.J. Browne (1999)
	69. Stallinger F. and P. Grunbacher (2001)
IBM Systems J. (5)	70. Davis G. (1982)
	71. Hausler P.A., R.C. Linger and C.J. Trammell (1994)
	72. Hevner A.R. and H.D. Mills (1993)
	73. Mills H.D. (1999)
	74. Sawyer S. and P.J. Guinan (1998)
IEEE Computer (5)	75. Baskerville R., L. Levine, J. Pries-Heje, B. Ramesh and S. Slaughter (2001)
	76. Boehm B. (1988)
	77. Bowen J.P. and M.G. Hinchey (1995)
	78. Brooks F.P. (1987)
	79. Ramamoorthy C.V. and W.T. Tsai (1996)
J. of Management Information Systems (5)	80. Barki H. S. Rivard and J. Talbot (1993)
	81. Browne G.J. and M.B. Rogich (2001)
	82. Liou Y.I. and M. Chen (1993)
	83. Hickey and Davis (2004)
	84. Ravichandran T. and A. Rai (1999)
Decision Support Systems (4)	85. Frolick M.N. and B.P. Robichaux (1995)
	86. Hevner A.R. and H.D. Mills (1995)
	87. Ramesh B. and K. Sengupta (1995)
	88. Trammell C.J., M.G. Pleszkoch, R.C. Linger and A.R. Hevner (1996)
European J. of Information Systems (4)	89. Bryant J. (1997)
	90. Chatzoglou P.D. and L.A. Macaulay (1996)
	91. Ropponen J. and K. Lyytinen (1997)
	92. Willcocks L. and H. Margetts (1994)
IEEE Software (4)	93. Jirotka M. and P. Luff (2006)
	94. Larman and Basili (2003)
	95. Neill and Laplante (2003)
	96. Potts C., K. Takahashi and A.I. Antón (1994)
Information Systems Research (4)	97. Hirschheim R. and M. Newman (1991)
	98. Lyytinen K., L. Mathiassen and J. Ropponen (1998)
	100. Marakas G.M. and J.J. Elam (1998)
	101. Nidumolu S. (1995)
Information Systems J. (3)	102. Fitzgerald B. (1996)
	103. Darke P. and G. Shanks (1997)
	104. Lyytinen K., L. Mathiassen and J. Ropponen (1996)
Knowledge-Based Systems (2)	105. Jones R.M., L. Candy and E.A. Edmonds (1993)
	106. Rolland C., C. Souveyet and M.B. Ayed (2003)

ACM Computing Surveys (1)
Behavior & Information Technology (1)
DATA BASE (1)
Human-Computer Interaction (1)
Information Systems
Information Systems Management (1)
Int. J. of Human-Computer Studies (1)
J. of Information Technology (1)
J. of Systems Management (1)
Scandinavian J. of Information Systems (1)

107. Lyytinen K. (1987)
108. Kujala S. (2003)
109. Chen M. and J.F. Nunamaker Jr. (1991)
110. Duggan E.W. (2003)
111. Pedersen, Jensen and Dyreson (2001)
112. Pai W.C. (2002)
113. Maiden N.A.M. and M. Hare (1998)
114. Quintas P. (1994)
115. McFarlan W. (1982)
116. Mathiassen L., T. Seewaldt and J. Stage (1995)

Appendix III

The technique overview was developed as follows. One author reviewed the selected literature and listed all requirements development approaches explicitly mentioned in the articles. We divided the identified approaches into methodologies and techniques according to the definitions in Iivari et al. (2000). Thus a methodology is "an organized collection of concepts, methods (or techniques), beliefs, values, and normative principles supported by material resources" and a technique "consists of a well-defined sequence of elementary operations that more or less guarantee the achievement of certain outcomes if executed correctly". We decided only to include techniques in our list because the proposed contingency model operates on the level of techniques rather than methodologies[3]. Therefore, we ended up not including full scale development methods such as Unified Model Language and Extreme Programming.

Next, all techniques were discussed among the authors. This led to omission of proprietary techniques and obvious duplicates. After cleaning the list, the authors classified the techniques based on the proposed categories. The authors split into two groups, one in Atlanta and the other in Helsinki. Each group classified the techniques independently using the facilitate.com group support software. Each technique was classified as having an emphasis on one or more of discovery, prioritization, experimentation, and specification using the definitions of these terms in the paper. This resulted in 69 techniques being unanimously classified as having an emphasis on one or more of the four categories; there were no techniques that were suggested to have emphasis on all four categories and only two comprehensive techniques that emphasized three; finally, there was disagreement about the categorization of 15 techniques. The classifications of these were resolved through a documented discussion amongst the authors. The table below shows the resulting classification of techniques.

Name	Reference	Specifi-cation	Experi-mentation	Discovery	Prioriti-zation
1. Affinity technique	(Duggan, 2003)			√	
2. Aspect mining in requirements specification	(García-Duque et al., 2006)			√	
3. Attributed goal-oriented analysis	(Kaiya et al., 2005)	√		√	
4. Behavior analysis	(Byrd et al., 1992)	√		√	
5. Box structure specification and design	(Hausler et al., 1994, Hevner and Mills, 1993, Hevner and Mills, 1995)	√			
6. Brainstorming	(Byrd et al., 1992)			√	
7. Business information analysis and integration technique	(Davis, 1982)	√		√	
8. Business process planning (BSP)	(Davis, 1982)	√		√	√
9. Card sorting	(Byrd et al., 1992, Maiden and Hare, 1998)			√	√
10. Cognitive mapping	(Byrd et al., 1992, Montazemi and Conrath, 1986)			√	
11. Contextual design	(Holtzblatt, 1995, Jones et al., 1993, Kujala, 2003)	√		√	√
12. Cooperative prototyping	(Leifera et al., 1994)		√		
13. CREV	(Hickey and Davis, 2004)	√		√	
14. CREWS	(Haumer et al., 1998)	√		√	
15. Critical success factors	(Byrd et al., 1992)			√	√
16. Data flow diagram	(Larsen and Naumann, 1992, Marakas and Elam, 1998, Ramesh and Browne, 1999)	√			
17. Decision analysis	(Watson and Frolick, 1993)			√	
18. Delphi method	(Davis, 1982)			√	√
19. Deriving requirements from existing system	(Davis, 1982)			√	
20. Domain specific modeling	(Franch and Carvallo, 2003)	√			

[3] We would like to thank the senior editor for this valuable comment

21. EasyWinWin	(Stallinger and Grunbacher, 2001)			√	√
22. Email/bulletin board	(Keil and Carmel, 1995)			√	
23. Ends/means analysis	(Wetherbe, 1991)			√	
24. Entity-relationship modeling	(Haumer et al., 1998, Pedersen et al., 2001	√			
25. Facilitated team	(Keil and Carmel, 1995)			√	
26. Focus group	(Keil and Carmel, 1995, Leifera et al., 1994, Telem, 1988)			√	
27. Future analysis	(Byrd et al., 1992)			√	
28. Goal modeling oriented requirements elicitation	(Darke and Shanks, 1997, Hevner and Mills, 1995, Leite and Freeman, 1991, Nuseibeh et al., 1994, van Lamsweerde and Letier, 2000)	√		√	
29. Goal oriented approach	(Byrd et al., 1992, Darke and Shanks, 1997)	√		√	
30. Group support systems and strategic business objectives	(Frolick and Robichaux, 1995)			√	√
31. Guided brainstorming	(Davis, 1982)			√	
32. Human, social and organizational requirements elicitation	(Andreou, 2003)			√	
33. Inquiry cycle model – structure and describe requirements discussions	(Potts et al., 1994)	√		√	
34. Joint application design	(Andrews, 1991, Kujala, 2003, Wetherbe, 1991)		√	√	
35. KAOS	(van Lamsweerde and Letier, 2000)	√			
36. Laddering	(Browne and Ramesh, 2002, Browne and Rogich, 2001, Byrd et al., 1992)			√	
37. Lyee	(Rolland et al., 2003)	√			
38. Machine rule induction	(Byrd et al., 1992)	√			
39. Marketing and sales	(Byrd et al., 1992)			√	
40. MIS intermediary	(Keil and Carmel, 1995)	√		√	
41. Multidimensional data models	(Pedersen et al., 2001)	√			
42. Multidimensional scaling	(Byrd et al., 1992)	√			
43. Nominal group technique	(Duggan, 2003)			√	√
44. Normative analysis	(Watson and Frolick, 1993)			√	
45. Object oriented Z	(Liu et al., 1998)	√			
46. Open interview	(Byrd et al., 1992)			√	
47. Open systems task analysis	(Jones et al., 1993)			√	
48. Participatory design	(Duggan, 2003, Kujala, 2003)		√	√	
49. Petri nets	(Lee et al., 1998)	√			
50. Petri nets combined with use cases	(Lee et al., 1998)	√		√	
51. Precision model	(Bostrom, 1989)			√	
52. Prime-CREWS	(Haumer et al., 1998)	√		√	
53. Process analysis	(Watson and Frolick, 1993)			√	
54. Protocol analysis	(Byrd et al., 1992)			√	
55. Prototyping	(Byrd et al., 1992, Davis, 1982, Watson and Frolick, 1993)		√		
56. Quality function deployment	(Duggan, 2003, Elboushi and Sherif, 1997, Pai, 2002,	√			√

Journal of the Association for Information Systems

	Ravichandran and Rai, 1999, Ravichandran and Rai, 2000, Zultner, 1993)				
57. Repertoire grids	(Byrd et al., 1992)			√	
58. Requirements generation model	(Arthur and Gröner, 2005)			√	
59. Requirements prototyping	(Keil and Carmel, 1995)		√		
60. Requirements workshops	(Hickey and Davis, 2004)			√	
61. Rich pictures	(Darke and Shanks, 1997)	√		√	
62. Scenario-based requirements elicitation	(Breitman et al., 2005, Haumer et al., 1998, Jarke et al., 1998, Rolland et al., 1998, Saiedian et al., 2005, Shin et al., 2005)	√		√	
63. Semantic maps	(Marakas and Elam, 1998)			√	
64. Socio-technical analysis	(Davis, 1982)			√	
65. State charts	(Haumer et al., 1998)	√			
66. Strategic business objectives	(Frolick and Robichaux, 1995)			√	√
67. Strategy set analysis	(Watson and Frolick, 1993)			√	
68. Structured group elicitation method	(Bryant, 1997)			√	
69. Structured interview	(Byrd et al., 1992, Wetherbe, 1991)			√	
70. Structured walkthroughs	(Salaway, 1987, Sawyer and Guinan, 1998)			√	
71. Support line	(Keil and Carmel, 1995)			√	
72. Surveys	(Keil and Carmel, 1995)			√	
73. Teach-back interview	(Byrd et al., 1992)			√	
74. Testing	(Keil and Carmel, 1995)		√	√	
75. Text analysis	(Byrd et al., 1992)			√	
76. Trade show	(Keil and Carmel, 1995)		√	√	
77. Usability lab	(Keil and Carmel, 1995)		√		
78. Use cases	(Lee et al., 1998)			√	
79. Use of video in requirements elicitation	(Jirotka and Luff, 2006)			√	
80. User group	(Keil and Carmel, 1995)			√	
81. User-interface prototyping	(Keil and Carmel, 1995, Ravid and Berry, 2000)		√	√	
82. Variance analysis	(Byrd et al., 1992)			√	
83. VDM ++,VDM-SL	(Liu et al., 1998)	√			
84. Warnier-Orr diagrams	(Fazlollahi and Tanniru, 1991)	√			
85. Z	(Liu et al., 1998)	√			

About the Authors

Lars Mathiassen received his master's degree in computer science from Aarhus University, Denmark, in 1975, his PhD in informatics from Oslo University, Norway, in 1981, and his Dr. Techn. degree in software engineering from Aalborg University, 1998. He is currently GRA Eminent Scholar and professor in Department of Computer Information Systems and co-founder of Center for Process Innovation at Georgia State University. His research interests are within information systems and software engineering with a particular emphasis on process innovation. He is a member of IEEE, ACM and AIS and coauthor of *Computers in Context* (Blackwell 1993), *Object Oriented Analysis & Design* (Marko Publishing, 2000), and *Improving Software Organizations* (Addison-Wesley, 2002). He currently serves as senior editor for *MIS Quarterly* and his research is published in journals like *Information Systems Research, MIS Quarterly, IEEE Transactions on Engineering Management, Communications of the ACM, Information, Technology & People, Journal of Strategic Information Systems, Information Systems Journal, Scandinavian Journal of Information Systems, Journal of Information Technology, and IEEE Software. lmathiassen@gsu.edu, Georgia State University.*

Timo Saarinen is a professor of information systems and electronic commerce at the department of Business Technology and a Vice-Rector for research at the Helsinki School of Economics. He holds a Ph.D. in information systems form the Helsinki School of Economics. Timo Saarinen has published in major journals as *MIS Quarterly, Journal of Management Information Systems, Information & Management and Journal of Strategic Information Systems*. His research interests include economics of information systems and the development of efficient market-driven services, with the focus on the multi-channel environment of electronic commerce.

Tuure Tuunanen is a Senior Lecturer in the Department of Information Systems and Operations Management at the University of Auckland. He holds a D.Sc. (Econ) from the Helsinki School of Economics. His current research interests lie in the areas of IS development methods and processes, requirements engineering, risk management, and convergence of IS and marketing disciplines, specifically in design of interactive consumer services and products. His research has been published in *Information & Management, Journal of Database Management, Journal of Information Technology Theory and Application, Journal of Management Information Systems, and Technology Analysis and Strategic Management*. Dr. Tuunanen is a member of Association of Computing Machinery, Association of Information Systems, and Institute of Electrical and Electronics Engineers.

Matti Rossi is a professor of information systems at Helsinki School of Economics. He is currently on sabbatical as a visiting researcher at Claremont Graduate College. He has worked as research fellow at Erasmus University Rotterdam and as a visiting assistant professor at Georgia State University, Atlanta and is currently a visiting professor at Claremont Graduate University. He received his Ph.D. degree in Business Administration from the University of Jyväskylä in 1998. He has been the principal investigator in several major research projects funded by the technological development center of Finland and Academy of Finland. He is the coordinating editor of *Scandinavian Journal of Information Systems*. His research papers have appeared in journals such as *CACM*, *Journal of AIS*, *Information and Management* and *Information Systems*, and over thirty of them have appeared in conferences such as ICIS, HICSS and CAiSE.

Journal of the Association for Information Systems
ISSN: 1536-9323

Editor
Kalle Lyytinen
Case Western Reserve University, USA