Attention Shaping and Software Risk—A Categorical Analysis of Four Classical Risk Management Approaches

Kalle Lyytinen • Lars Mathiassen • Janne Ropponen
Department of Computer Science and Information Systems, University of Jyväskylä, Seminaarinkatu 15, PL 35, 40350 Jyväskylä, Finland
kalle@cs.jyu.fi
Department of Computer Science, Aalborg University, Frederik Bajers Vej 7, DK-9220 Aalborg East, Denmark
lmathiassen@cit.dk
Nokia Telecommunications, Network Management Systems, P.O. Box 759, 33101 Tampere, Finland
janne.ropponen@mission.fi

This paper examines software risk management in a novel way, emphasizing the ways in which managers address software risks through sequential attention shaping and intervention. Software risks are interpreted as incongruent states within a socio-technical model of organizational change that includes task, structure, technology, and actors. Such incongruence can lead to failures in developing or implementing the system and thus to major losses. Based on this model we synthesize a set of software risk factors and risk resolution techniques, which cover the socio-technical components and their interactions. We use the model to analyze how four classical risk management approaches—McFarlan's portfolio approach, Davis' contingency approach, Boehm's software risk approach, and Alter's and Ginzberg's implementation approach—shape managerial attention. This analysis shows that the four approaches differ significantly in their view of the manager's role and possible actions. We advise managers to be aware of the limitations of each approach and to combine them to orchestrate comprehensive risk management practices in a context. Overall, the paper provides a new interpretation of software risk management which goes beyond a narrow system rationalism by suggesting a contingent, contextual, and multivariate view of software development.
(*Software Development; Information System Failure; Risk Management; Risk Management Technique; Socio-Technical Analysis; Content Analysis*)

1. Introduction

In 1992 a large hardware company cancelled its strategic sales support system initiative CONFIG after ten years of continued struggle. The system had swallowed a huge amount of development effort, management attention and cost over 100 million US dollars (Keil 1995). The major reason for failure was early on signalled to management, but it ignored this information systematically (Markus and Keil 1994) due to their wrong conception of the role of the technology and organizational politics. The cause—inattention to the nature of sales personnel's work practices and inadequate fit between their tasks and the developed system (Markus and Keil 1994)—was never addressed. Instead, management proposed numerous alternative solutions to save the system: technological fixes (through improved user interfaces and better reliability), structural changes (by installing participative design teams), and actor related changes (by starting thorough user training and support), but with meager results. In the end the system failed because of inattention to a fundamental flaw in the design.

1047-7047/98/0903/0233$05.00
Copyright © 1998, Institute for Operations Research and the Management Sciences

Unfortunately this example is not unique. The information systems field has been plagued by various system failures (Lyytinen and Hirschheim 1987) such as failures to deliver a system, budget overruns, massive delays, or organizational rejection. Usually they are outcomes of cognitive limitations, management inattention, or mediocre skills to address observed problems. To combat these a wide variety of approaches have been developed but with relatively weak results (Lyytinen 1987, Lyytinen and Hirschheim 1987). Information system development has remained a high risk proposal.

One promising approach to deal with system failures is software risk management and a large number of approaches to manage software development[1] risks have been developed. In general, they help identify, analyze, and tackle *software risks* (Charette 1989, Boehm and Ross 1989, Boehm 1991, Fairley 1994), *implementation risks* (Lucas 1981, Alter and Ginzberg 1978, Keen and Scott-Morton 1978, Lyytinen 1987, Kwon and Zmud 1987), *project portfolio risks* (McFarlan 1982, Earl 1987), or *requirements risks* (Davis 1982, Burns and Dennis 1985) by formalizing risk oriented correlates of success and deriving from them a set of managerial principles (Boehm 1991). These approaches help managers to question critical assumptions underlying software development and to identify and handle incidents, which threaten successful software operation or cause software rework, implementation difficulty, delay, or uncertainty. The failure above could, for example, have been avoided if a critical analysis of the assumptions concerning the system's task and the actors involved had been carried out (Lyytinen et al. 1996).

It is surprising to notice how diverse risk management approaches are. They often address similar situations with different tactics and different situations with similar tactics. For example, user's lack of experience can be attacked in these approaches with technologies like prototyping, organizational process changes like participation, or changing the requirements (task) by avoiding the change. In a similar vein, information hiding can be used to manage requirements changes, to overcome the problem of nonexistent users, overcome designer's lack of experience, and to manage the complexity of the design. This raises questions such as: How do these approaches relate to one another? Why do they differ so much? Which concepts underlie these approaches? Do the approaches embody any theoretically reasoned theory of risk and risk oriented behavior? Overall, we lack theories, which help relate risk management approaches and explain to what extent, how, and why they vary. We also lack systematic frameworks to organize risk assessments and to generate risk resolution tactics. The majority of studies present prescriptions to combat specific sets of risks and they only provide weak theoretical discussions of the concept of software risk, risk management, and risk management approaches (Boehm 1989, 1991, Charette 1989, McFarlan 1982). Most students of software risk management have not been attentive to a growing body of theories of risky behavior, uncertainty absorption, and organizational change (March 1988, Galbraith 1977, Perrow 1984, Roberts 1993). In summary, the area is dominated by limited and ad hoc analyses and by specific models to manage particular aspects of software risks.

The goal of this paper is to nurture a more systematic account of risk management by drawing on behavioral theories of risky behavior (March and Shapira 1987), by considering risk management as an organizational attention shaping routine (Cyert and March 1963), and by using socio-technical models of organizational change (Leavitt 1964). The key insights are that software risk managers, rather than following a rational calculus, attempt to master their environments in order to avoid major losses, and that risk management approaches fundamentally embody organizational routines of attention shaping which represent causal dependencies between managerial action and observed events. We draw upon Leavitt's (1964) socio-technical model of organizational change to highlight how each

[1]We understand software development quite broadly to cover requirements analysis, design, implementation, and organizational adoption of software systems. This definition is broader than usually assumed in software engineering and associated discussions of risks (Boehm 1991, Charette 1989), but it coincides quite well with the notion of information systems development as expressed in standard textbooks (Davis and Olson 1985), or literature surveys on IS development (Lyytinen 1987, Iivari 1991).

risk management approach shapes managers' attention in specific ways and how they invoke a limit set of heuristics to guide intervention. More specifically, we apply categorical analysis to discover and contrast the attention foci, heuristics, and intervention targets of four authoritative approaches that have dominated our discussions of software risk management. These are Boehm's (1989, 1991), Davis's (1982), McFarlan's (1982), and Alter's and Ginzberg's (1978) models.

The argument is organized as follows. After a short overview of the risk concept in managerial theory and in software development we develop a socio-technical model and justify its use as a means to analyze the content and logic of different risk management approaches. Based on a literature survey, we then summarize a categorization of risks and risk resolution techniques using the socio-technical model. After this, we conduct a categorical analysis of the four classical risk management approaches. We conclude by raising questions about these four approaches, by suggesting how they can be combined for improved risk management, and by proposing new avenues for theoretical and empirical research in software risk management.

2. Risk Management

2.1. Risk in Managerial Theory

In rational decision theory the concept of risk reflects the variation in the distribution of possible outcomes, their likelihoods, and their subjective values (Arrow 1965). A risky alternative is one for which the variance is large, and risk is one of the attributes which, along with the expected value of the alternative, is used in evaluating alternative gambles. Rational choice theory expects that decision-makers deal with risk by first calculating alternatives and then choosing one alternative among the available risk-return combinations yielding the highest outcome thus behaving in a rational (risk-aversive) manner (Yates 1992).

The decision theoretic view is not, however, consistent with empirical studies of how managers deal with risks and how they define success (March and Shapira 1987, Bromley and Curley 1992). In these studies, managers follow a less precise calculus. First, uncertainty of positive outcomes is not behaviorally an important aspect of risk. For managers a risky choice is one that contains a threat of a poor performance (March and Shapira 1987, Fischoff et al. 1984) and success is defined through the avoidance of such alternatives. Second, human decision makers do not treat risk with a probability concept. Instead, they associate it with the magnitude of a bad outcome (Bell 1985). Accordingly, managers act in a loss-aversive manner instead of a rational one as predicted by the traditional theory (Kahnemann and Tversky 1982, Arrow 1965) and they define their measures of success accordingly. Moreover, though quantities may be involved in assessing the level of risk, there is little desire to reduce risk to a single construct of outcomes.

This behavioral view of risk has several implications. First, when risks involve great losses, managers seek to avoid risks rather than just accept them (Cyert and March 1963). They make fast decisions to avoid risks, negotiate uncertainty absorbing contracts, or just delay decisions if possible (MacCrimmon and Wehrung 1986). Second, managing risks is not seen as gambling, but as mastering the environment so as to bring risks under control (MacCrimmon and Wehrung 1986; c.f. Adler's (1980) concept of "risk makers"). Third, managers neither understand, nor care to use precise probability estimates: crude characterizations are used to exclude certain possibilities from the decision (Fischoff et al. 1981) and thus make the managerial process a sequential pruning exercise instead of an overall optimization decision.

2.2. Risk in Software Development

In the literature dealing with software development risks (Boehm and Ross 1989, Charette 1989, Boehm 1991), risk is primarily defined following the rational decision theoretic view. Software risks are seen to form independent speculative gambles which have both a downside loss and an upside profit associated with them. Yet, most risk management approaches deal solely with negative outcomes and with how to avoid them. The central insight of the rational decision theoretic view (the importance of considering the whole distribution of possible outcomes) thus becomes obscured.

The behavioral perspective (March and Shapira 1987) more accurately characterizes the assumptions that underline most risk management approaches:

they help focus on ambiguous losses (Boehm 1991), rely on multidimensional, qualitative models that make the management task simple and practicable, and they seek to avoid or master risks through sequential pruning exercises. Accordingly, risk management forms a continuous exercise where project managers engage in multiple maneuvers in order to master their environment. In most situations, these maneuvers exclude bad alternatives (Boehm 1991, Charette 1989). To achieve this, risk management approaches suggest a plethora of means for controlling the environment. Examples of such means abound in the literature: investments in methods and standards, ignoring user resistance through "sign-off" practices, or preferring standards for interfaces (Boehm and Ross 1989) are but a few examples.

The behavioral interpretation of risk management invites us to examine how software managers enact cognitive images of the complex environments they seek to master (Ciborra and Lanzara 1987, Seely Brown and Duguid 1991). These strategies help managers make sense of their situations, exclude bad choices, and launch maneuvers to reduce, or avoid risks. They provide managers with concrete examples of development situations, and their dynamics, as well as how interventions might affect the situation. These strategies are limited, selective, and relatively stable. They are primarily outcomes of past experience, or have been learned from studying analogical situations. Sometimes they are derived from abstract scientific theories using deductive reasoning (March et al. 1991). Over time, such images tend to become more elaborated and involve relatively complicated cause-effect chains between situations and the resulting management action.

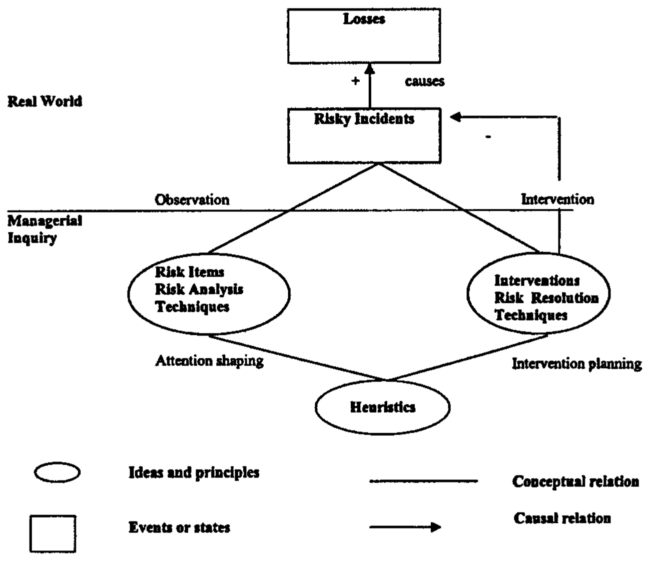
To cope with a wider range of software development situations and to improve their ease of use, many such managerial strategies have been formalized into risk management approaches. These approaches—despite their great variation and drastic differences—provide recipes in a relatively standardized format of how to inquire and observe, how to organize and interpret observations, and how to subsequently launch managerial action. They form institutionalized organizational routines that are sequentially carried out to master the environment (March and Olsen 1989).

In Figure 1 we represent schematically the general format of risk management approaches. We distinguish between the realm of management inquiry which consists of both attention shaping patterns and plans of managerial intervention and the realm of the real world where the actual software development takes place. Risky incidents are events or states in the real world which have a potential to cause a loss and thus make the development project fail. They remain unknown until observed and handled by the software developer, i.e. until they have been spotted in the search light of managerial attention and treated in some manner (which was not the case with the CONFIG system). Their origin can vary from internal personal deficits (poor project management skills) to external "acts of God" (business merger) (Schmidt et al. 1997).

Risk management approaches make use of implicit causal (local) theories—or causal dependencies[2]—of the development environment. Such dependencies

[2]We use in this paper the term causal dependency to differentiate these forms of knowledge from validated scientific theories. Such dependencies are incomplete, ambiguous, poorly validated, and even contradictory. Such dependencies, however, must be assumed to make any managerial action possible, i.e. if "I set out to do A I can achieve B" assumes a causal dependency of the form A → B.

Figure 1 Risk Management Approaches



suggest what to observe and how to intervene. They enable and constrain managerial cognition and action by making events and actions intelligible, and by determining what can be seen (objectified) and acted upon. They also reflect an archeology of learning in the sense that they are based on experiences from interactions between risky incidents and losses or between interventions and risky incidents in the past. The implicit causal theories (or risk management approaches) include two types of statements. First, they conjecture positive causal dependencies between risky incidents and losses (as depicted by the plus sign). For example, a theory could state that the lack of adequate requirements specification increases the likelihood of organizational rejection. Second, they conjecture negative causal dependencies between management interventions and the existence or the severity of a risky incident (as depicted by the negative sign). For example, a theory could state that the use of a prototyping technique decreases the likelihood of not having adequate functional requirements. Both these types of causal dependencies are needed to formulate normative guidelines to address software risk.

Risk management approaches are normally represented using the concepts of risk items, risk resolution techniques, and heuristics. Risk items (or risk factors) combined with heuristics constitute the attention shaping component of the risk management approach. Risk items are derived from postulated positive causal dependencies between risky incidents and losses. They provide a vocabulary to recognize and classify risky events and states. An example of a risk item is "Unclear or misunderstood scope and objectives" (see Schmidt et al. 1997). Risk items form cognitive mechanisms that direct managerial attention, for example in the form of checklists of risk items that managers use to scan their environment (Boehm 1991). Risk management approaches further operationalize risk items through risk analysis techniques, which prescribe how to organize observations for further analysis (see e.g. McFarlan 1982, Davis 1982, Boehm 1991). They may also suggest how to organize risks into separate classes in terms of urgency or type, to aggregate them into new concepts (such as general level of risk), or to provide measures for the reliability of observations.

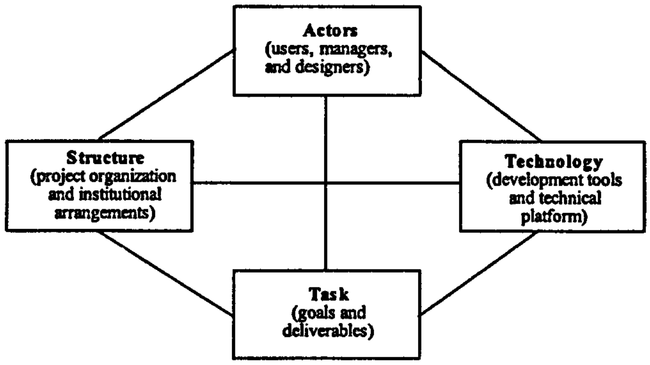
Heuristics combined with risk resolution techniques constitute the intervention planning part of the approach. Risk resolution techniques are based on espoused causal dependencies of how interventions influence risky incidents, and how this will change the consequent development trajectory. Each resolution technique suggests a schematic plan for an intervention that will decrease the impact of at least one risky incident, or help avoid it altogether. Examples of risk resolution techniques are "Apply user participation" or "Use scenarios".

Heuristics represent formalized decision making routines to master the environment. To be more exact they embody knowledge of cause-effect chains within the environment. They link recognized risky incidents (called in the risk literature risk profiles or risk lists, see Lyytinen et al. 1996) to would-be-effective managerial interventions. As embodiments of decision-making routines they follow a rule-format. Heuristics consists of statements of the form "If *X* set out for *Y* unless *W*", where *X* denotes a set of risky incidents organized into specific risk items, *Y* denotes a set of risk resolution techniques, and *W* states a set of exceptions where the rule does not apply.

For example the set *X* might include the item "Unclear or misunderstood scope and objectives" and the set *Y* items like "Use scenarios" and "Apply user participation", while *W* would be unless "The project is smaller than 2 man months". Heuristics may thus consist of one or more risk resolution techniques for each risk item included in the risk profile. Moreover, heuristics may need to be augmented with second order rules which rank heuristics that apply to the same risk item, or which may help exclude heuristics because of their contradictory nature.

This interpretation of risk management approaches highlights a number of crucial questions: 1) Which aspects of the situation are inductively generalized into risk items, i.e. which types of causal dependencies underlie different risk management approaches? 2) Which risk resolution techniques are proposed, i.e. which causal dependencies are postulated between interventions and risky incidents? 3) How do heuristics combine risk items and risk resolutions techniques? and 4) What is the level of detail, the format, and the degree of formality of such rules?

Figure 2 A Socio-Technical Model of System Development


Actors
(users, managers, and designers)
Structure
(project organization and institutional arrangements)
Technology
(development tools and technical platform)
Task
(goals and deliverables)

3. A Categorization of Software Risks

3.1. A Socio-Technical Change Model

The idea of risk management approaches as formalized routines (rule-sets) invoked to master the environment raises the issue of the content and structure of the causal dependencies underlying them. In this paper we use Leavitt's open system model of organizational change (Leavitt 1964) to analyze this issue. Accordingly, we interpret a software risk as a variation in a socio-technical system. This interpretation helps overcome the idea of gambles and rational decision calculus, which are currently fashionable. Instead, we characterize software risk management by using ideas of sequential environmental adaptation and of interventions that establish, or maintain the system equilibrium.

The socio-technical model is well-suited for our purposes.[3] It has been widely used to classify schools of organizational change and to understand management of organizational change. It has also been used extensively in the IS literature (Mumford 1983, Keen 1981, Kwon and Zmud 1987). Moreover, it was originally developed as an attempt to achieve a synthesis of major dimensions of organizational change, which is similar to what we are trying to do with software risk management approaches. Finally, the model displays the virtues of a good classification model: it is simple, extensive, and it is sufficiently well defined to be applicable. As earlier research using the model shows, it can also be extended.[4]

The socio-technical model (Leavitt 1964) views organizations as multivariate systems that consist of four interacting components—task, structure, actor, and technology. These components can easily be translated into well-known elements of software development that form the basic building blocks of embedded causal theories: actors cover all stakeholders including users, managers, and designers; structure denotes project organization and other prevailing institutional arrangements; technology means development tools and methods, as well as hardware and software platforms; and task signifies expected outcomes in terms of goals and deliverables (see Figure 2).

A key conjecture of the model is that these four model components are strongly related: change in one component will have effects, planned or unplanned, on the others. These interrelations form causal dependencies that inform risk management heuristics. This strong mutual dependency is depicted in Figure 2 by the edges that connect every component with the others thus creating the well-known diamond shape. The model also postulates—based on the open system equilibrium notion—that if one component's state is incongruent with others, this will create considerable dysfunctional effects in others and on the whole system. This effect is referred to as a variation by socio-technical theory (Mumford 1983). The theory also postulates that because these components are in a continuous change and interaction due to influences from the environment, the variations are constant and unavoidable. Accordingly, the goal of the management system is to control such variations in order to maintain the system in balance.

[3]The socio-technical model of organizations has been widely criticized in the literature. These include its focus on the static structure, ignorance of the environment, and the nature of the distinctions it suggests. Many of these are valid concerns if one wants to develop a comprehensive theory of organizations in uncertain and complex environments. But this is not our goal: we want to use the model only as an analytical framework to examine software risk management approaches and to interpret the concept of software-risk.

[4]Such extensions have been made. For example, Kwon and Zmud (1987) augment the model with the concept of environment and environmental factors. Davis and Olson (1985) add to the model the concept of organizational culture.

Disturbances created by the variations, which endure over longer periods of time, or which are unexpectedly strong, will induce oscillations, which will decrease the systems' capability to accomplish its task, and can eventually undermine the existence of the whole system.

The connection between the model and the behavioral concept of software risk can be stated as follows: *a change in any socio-technical component or relation in a systems development process can create variations which, in the extreme, can lead to a failure of the system development (system), otherwise known as a loss.* Accordingly, risk analysis becomes similar to analyzing and managing variances in socio-technical systems (Mumford 1983). Major variations form risky incidents which increase the difficulty in estimating what the performance of the development project is likely to be in terms of product or process (Nidumolu 1995). These incidents can be categorized into one or several model components—such as technology and its reliability—or their connections—such as an actor's capability to deal with the technology. In this way, the model provides a basis to analyze the content of causal dependencies informing risk management approaches and suggests that these dependencies can relate to any or all of the components.

3.2. Software Risk Items and Risk Resolution Techniques

We have used the socio-technical model to synthesize the literature in software risk management, and to specify in more detail the risk items and risk resolution techniques (see Appendix 1). In particular, we have classified problems related to development trajectories and examined suggested coping strategies using the diamond model. Our analysis demonstrates that all components—the task, structure, technology, actors, and their combinations—constitute key risk items as well as targets for risk resolution techniques. Below we describe in more detail the contents of each model component.

The model component *task* describes an organization's raison d'etre (Leavitt 1964). In system development a task is normally defined through project deliverables and process features, i.e. a development task dictates what developers should accomplish and how (Blokdijk and Blokdijk 1987). Several task related features increasing the exposure to risk have been identified: task size or complexity (Lyytinen 1987, Curtis et al. 1988, Oz 1994, Waters 1993, Zmud 1980, Beath 1987), task uncertainty (Burns and Dennis 1985, Lucas 1982, Turner 1992, Waters 1993, Nidumolu 1994), specificity of design (Saarinen and Vepsäläinen 1993b), the stability of the task (Walters 1993, Saarinen and Vepsäläinen 1993a), the ambiguity or existence of the task description (Beynon-Davis 1995, De Salabert and Newman 1995), or the limits to what is known (Jones and Walsham 1992). Process related features include unrealistic targets (Sabherwal and Elam 1996) or pressures to get the system working (Sabherwal and Elam 1996). In general, these can be classified along two task related properties (Mathiassen and Stage 1992): task complexity which represents the amount of relevant information available to carry out the task, and task uncertainty which is defined by the availability, validity, and reliability of the task related information. The higher the amount of available information about the task, or the lower its validity and reliability, the higher is the development risk.

The *structure* component covers systems of communication, systems of authority, and systems of work flow (Leavitt 1964). It includes both the normative dimension, i.e. values, norms, and role expectations, and the behavioral dimension, i.e. the actual patterns of behaviors as actors communicate, exercise authority, or work. Development risks associated with structure have been discussed in relation to: task (Beath 1987), technology (Curtis et al. 1988), or actors (Markus and Keil 1994). Neglect of the structural dimension is likely to generate considerable difficulties in developing the system within time and cost (Curtis et al. 1988, Thambain and Wilemon 1986, Nidumolu 1995, van Swede and van Vliet 1994, van Genuchten 1991), in getting the requirements right (Nidumolu 1995, Beath 1987), or getting the system accepted (Davis et al. 1992, Markus and Keil 1994).

Actors represent individuals or groups of stakeholders who can set forward claims or benefit from software development. Actors include customers, managers, maintainers, developers, and users (Boehm and Ross 1989). Examples of actor related software risks abound: personal shortcomings (Keil 1995, Boland

1992), lack of commitment and skill (Beynon-Davies 1995), differences among stakeholders (Wilkocks and Margetts 1994), wrong expectations (Ginzberg 1981), false beliefs (Hirschheim and Newman 1991), nonexistent or unwilling users (Grover et al. 1988), unethical professional conduct (Oz 1994), personnel turnover, personal politics, and opportunism (Borum and Christiansen 1993, Markus and Keil 1994, Saarinen and Vepsäläinen 1993b, Keen 1981, Grover et al. 1988), or dysfunctional effect of actors' learning on project performance (Brooks 1974). Empirical studies also stress the importance of individual talent and experience in reducing task or technology related risks (Curtis et al. 1988, Henderson and Lee 1992).

Technology denotes "tools—problem solving inventions like work measurement, computers, and drill presses" (Leavitt 1964).[5] In line with the concept of "problem solving inventions" we include within technology the methods, tools, and infrastructure used to develop and implement the software system (Cooprider and Henderson 1991). These technologies can create considerable risks, especially if they are unreliable, inefficient, nonstandardized, noncompliant, or have functional limitations (van Genuchten 1991, Willkocks and Margetts 1994, Oz 1994, Sabherwal and Elam 1996).

Considering each of the four components individually provides a valuable, first order understanding of sources of risks and their resolution. Equally important are, however, *relations* which deal with interdependencies between task, structure, technology, and actors.

Task-Actor interdependencies focus on the actors' ability and shortcomings in relation to achieving the task, the ability to specify and analyze the task and its problems, and the inclination to make shortcuts. Typically such risks relate to the actor's experience in carrying out or specifying the development task, their disagreement about the task (see e.g. Curtis et al. 1988, Benyon-Davis 1985, Lyytinen 1988), or opportunism (Sabherwal and Elam 1996).

Task-Technology interdependencies clarify how technologies fit with the task, and how misfits can create considerable risks. Several contingency models have formulated rules to select an appropriate software development technology for a given task (Mathiassen and Stage 1992, Saarinen and Vepsäläinen 1993a). Studies in software engineering (Curtis et al. 1988, Jarke and Pohl 1992, van Genuchten 1991, Ciborra and Lanzara 1987) stress the importance of deploying appropriate technology components, such as CASE-tools, e-mail, or configuration tools to cater for task variation. Some empirical studies have shown that the level of technological sophistication tends to increase the task complexity (Sørensen 1993) and thereby the development risk.

Task-Structure interdependencies deal with how the project organization is instrumental in carrying out the development task, and how a misfit between the structure and the task can bring about risks. These concerns have lead to contingency models of the interactions between the development task and the institutional arrangements (Boehm 1988, Ciborra and Bracchi 1983, Lehtinen and Lyytinen 1986, Parnas and Clements 1986, Beath 1987). The conclusion from these works is that inappropriate arrangements can lead to bad or unsatisfactory outcomes (Nidumolu 1995, Saarinen and Vepsäläinen 1993b, Beath 1987).

Actor-Technology interdependencies address risks which are created by improper matching of people with technology, or by introducing untried technologies. Problems of matching inexperienced users with unknown and complex technologies are well documented in the empirical literature (Barki et al. 1993, Keil 1995, Smolander et al. 1990, Kendall et al. 1992). Technologies may also lower professional esteem and change career paths (Orlikowski 1993). In some empirical studies we have discovered that technological choices like machine architecture correlate with some actor related risks such as gold plating i.e. adding nonnecessary and costly features into the system (Ropponen and Lyytinen 1997).

Actor-Structure interdependencies focus on interactions between the structure and the actors. Typical concerns are: incentive schemes and sanctions, values and beliefs, and how actors' behaviors are in concordance with the prevailing organizational structure. This area has been a traditional focus in socio-technical design and project management (Mumford 1983, Lundeberg et al. 1981, Andersen et al. 1990, DeMarco and Lister

[5]He also goes on to point out that there is "some uncertainty about the line between structure and technology" (see also Gervin 1981).

1987, Henderson and Lee 1992). Recent studies have focused on the interactions between stakeholder participation and process structure (Sabberwal and Robey 1995) and on the problems related to actors being located in different places (Sabherwal and Elam 1996).

Technology-Structure interdependencies deal with interactions between technology and the organizational structure. The notion is that an inappropriate structure, given the technology, or inappropriate technology, given the structure, will create considerable disturbances (Caufield 1989, Schmidt 1992). Typical concerns are: how work-flows (Curtis et al. 1992) and systems of authority and communication lines are affected by technological choices and what risks these choices create. Technologies can be in conflict with the systems of authority (Markus 1983, Markus and Robey 1983), or they can prevent creation of effective lines of communication (Mumford 1983). The use of a highly formalized specification technique assumes, for example, formal sign-on procedures and this can undermine effective communication with users (Bjørn-Andersen and Markus 1993).

4. A Categorical Analysis of Four Approaches

We can use the socio-technical model to analyze how risk management approaches can shape managers' attention towards specific risk items and risk resolution techniques. The available literature is not particularly detailed about the structure of the heuristics that relate risk items to specific interventions. We therefore decided to analyze in detail four authoritative approaches that provide specific normative guidelines for risk managers (Alter and Ginzberg 1978, Boehm 1989, 1991, Davis 1982, McFarlan 1982).[6] The motivation for choosing these four approaches was their widespread use and typical coverage of risk management issues. They are seen by many as classics in the field (see Lucas 1981, Saarinen and Vepsäläinen 1993a).

[6]These four approaches were not included in the literature analysis that derived the table in Appendix 1 in order to avoid bias in the analysis. The resulting classifications of these four models were compared with the list for omissions and weaknesses, but none were found.

4.1. Research Method

We used categorical analysis[7] in analyzing these approaches. First, we classified their lists of risk items into the different model components. In a similar manner we characterized their risk resolution techniques. Finally, by using the format of heuristics as illustrated in Figure 1 we investigated how the approaches translate risky incidents into managerial action. We chose categorical analysis to make valid inferences from the studied texts to their underlying meaning in terms of pre-specified set of categories (Weber 1985). In our situation, this analysis technique helps clarify the theories of development situations that underlie software risk management approaches. In particular, we used categorical analysis to reveal the following aspects in each approach (for a similar approach see Beath and Orlikowski 1994):

(1) The *definition of risk* indicates the risk concept as explicitly defined in the approach.

(2) The *risk management rationale* indicates which justifications are provided for the use of the approach and defines specific goals that managers should pursue.

(3) The *underlying risk metric* indicates whether risk is defined by using single or multiple constructs together with the applied measurement scale. This sheds light on whether the approach uses higher level concepts to aggregate observed risky incidents into summarized information of the observed variations.

(4) The *risk behavior type* indicates whether the approach assumes a rational or a loss-aversive stance, i.e. whether the approach follows an economic or a behavioral concept of decision-making.

(5) The *number of and coverage of risk items* indicates the number of risk items recognized in the approach. This can be interpreted as a coarse indicator of how fine grained the approach is in distinguishing alternative sources as risky incidents. The coverage (in terms of proportions) highlights the major focus of the approach and also how wide its scope is in terms of the socio-technical components.

(6) The *focus of risk items* indicates how strongly different model components are covered. By counting the

[7]Normally this method is called content analysis. But we use the term categorical analysis as suggested by one of the reviewers as it better conveys the idea underlying the research technique.

number of items per component and their relative proportions we can determine how focused or balanced the risk identification patterns are.

(7) The *number and coverage of risk resolution techniques* is a coarse indicator of how fine grained and rich the approach is in suggesting interventions. We take the number of items as indicating the level of detail, while the coverage ratio indicates the scope of the intervention patterns. Multiple usage, as opposed to single usage, indicates that the same risk resolution technique can be used to resolve problems associated with several risk items.

(8) The *focus of risk resolution techniques* indicates how different model components are covered by risk resolution techniques. By counting the number of risk resolution techniques per component and analyzing their relative proportion we can judge how focused or balanced the risk resolution patterns are.

(9) The *heuristics* indicates how risk items and risk resolution techniques are related.

(10) The *application scope* indicates when and under what circumstances the approach should be applied. Continuous approaches are used throughout the development process, while discrete approaches are applicable only in specific phases or contexts. This provides information about the conditions that must prevail in the management environment to effectively use the risk management approach.

Each approach and their mappings into model categories are explained in Appendices 2 through 5. The coding and analysis of the data is explained in Appendix 6, which also discusses the reliability and validity of the coding.

4.2. How Do the Four Approaches Differ?

The results of the analysis are summarized in Table 1. The table reveals important differences in the four risk management approaches.

The Concept of Risk. We can observe prominent differences between the four approaches in their concept of risk. Alter and Ginzberg discuss risk of failure, but only the failure to implement the system organizationally. Boehm's concept of risk is more restricted and focuses on avoiding losses for some or all stakeholders during the project. Davis' view (due to its focus on requirements specification) deals exclusively with the difficulty of obtaining a correct and complete understanding of the task. McFarlan defines risk as failure to obtain some or all the goals that are relevant in setting up the project. These differences in the notion of software risk explain diversity in the way risks are treated in these approaches and how they engage managers to react to software development problems. There is a clear difference between, on the one hand, Boehm's and Alter and Ginzberg's approaches, and, on the other, the approaches of Davis and McFarlan. The former focus entirely on possible losses, but provide no means to measure the magnitude of loss. As a consequence, these approaches cannot be used to compare risk levels of different projects and help managers evaluate the magnitude of risk exposure and the scope of losses. The latter set up a contingency model in which they match situational risk items to a repertoire of risk resolution techniques. In doing so they abstract situational risk items into a simple rank order metric (instead of the nominal scale metric used with Boehm and Alter and Ginzberg) which allows for comparison between different projects. Moreover, they (though somewhat implicitly) look at the combination of costs and benefits, i.e. they are more in line with the rational decision ideal.

Risk Item Focus. We notice major differences in how the four approaches mould the attention of software risk managers. Whilst Boehm is balanced in that all four socio-technical components perceive equal weight, the other three approaches are more concentrated in their risk item focus. Davis ignores technology and structure related risks; McFarlan sees risk items to be primarily task related; and Alter and Ginzberg focus on actor related risk items. It is also worth noticing that the two contingency approaches, i.e. Davis and McFarlan (for operational reasons) identify very few, high level risk items, whereas the two other approaches identify more items on varying levels of detail and thus offer a more fine grained vocabulary and classification scheme for observing software development situations.

Risk Resolution Focus. The four approaches propose different parts of the environments that can be submitted to management control. Accordingly, they

Table 1 Categorical Analysis of Four Risk Management Models

Type of Analysis	Alter and Ginzberg (1978)	Boehm (1991)	Davis (1982)	McFarlan (1982)
Risk Definition	"Uncertainty as to whether the project or parts of it can be completed at all" (p. 23)	"Risk exposure is the probability of an unsatisfactory outcome times the loss to the parties if the outcome is satisfactory . . . Unsatisfactory outcome is multidimensional: . . . budget overruns, wrong functionality, user interface shortfalls, . . . poor quality software." (p. 33)	"Difficulty in arriving at correct and complete requirements" (p. 5)	"Exposure to . . . Failure to obtain all, or any of the benefits . . . Costs of implementation that vastly exceed planned levels . . . Time for implementation that is much greater than expected . . . Technical performance significantly bellow estimate . . . Incompatibility of system with hardware and software" (p. 13)
Risk Metric	Multidimensional construct, nominal scale	Multidimensional construct, nominal scale	Single construct, rank order scale	Single construct, rank order scale
Risk Behavior Type	Loss-aversive	Loss-aversive	Rational (risk aversive)	Rational (risk aversive)
Risk Management Rationale	Avoid failure in organizational implementation	Avoid cost overrun, delays, rework, overkill, poor quality of software	Increase likelihood of obtaining useful information requirements	Maximize fit between project management approach and level of risk
Number of Risk Items	#8	#10	#3	#3
Coverage of Risk Items[a]				
Actor	6.5 (1)[a]	2 (2)	2	0.5 (1)
Task	0	3.5 (1)	1	2
Technology	1	2.5 (1)	0	0.5 (1)
Structure	0.5 (1)	2	0	0
Risk Item Focus	Actor	Balanced	Actor and Task	Task
Number of Risk Resolution Techniques	#16 with multiple usage	#36 with multiple usage	#4 with single usage	#29 with single usage
Coverage of Risk Resolution Techniques				
Actor	5/15	4/4.5 (2)	0	2 (2)
Task	2/3	2/4 (0)	0	0
Technology	3/8	21/30 (0)	2 (4)	7
Structure	6/17	9/10.5 (2)	2 (4)	20 (2)
Risk Resolution Focus	Balanced	Technology and structure	Technology and structure	Structure and technology
Heuristics	A set of resolution techniques proposed for each risk item No rules for the composition of resolution techniques The same resolution technique applies to different risk items	A set of resolution techniques proposed for each risk item No rules for the composition of resolution techniques The same resolution technique applies to different risk items	Four strategies for requirements determination Selection of strategy based on the risk level Each strategy composed of risk resolution techniques	Eight strategies for project design Selection of strategy based on risk items Each strategy involves a number of risk resolution techniques
Scope	Continuous	Continuous	Discrete	Discrete

= number of distinct risk items and resolution techniques identified in each approach.

x / y = number of distinct entries / number of all entries.

(z) = number of relation entries, i.e. entries related to two Leavitt variables.

[a]Use of the value 0.5 is due to "relational" risk items and risk resolution techniques, which look at the interactions and dependencies between components. See Appendix 6 for a more detailed explanation.

embody distinct causal dependencies as to how variations in the development system unfold. Alter and Ginzberg are primarily concerned with implementation risks. These are perceived as products of actor's attitudes and beliefs and they are combined with a detailed and balanced set of risk resolution techniques that cover all model components. Boehm's focus is on traditional software engineering solutions and he exhibits a clear attachment to technological fixes. His approach tells software managers that software risks are better managed by disciplining the process and by applying new or complementary technologies. Davis' focus and bias are different. He assumes that the task and actor elements are fixed and that the primary sources of risk are technology and structure during requirement specification. Therefore, risk resolution tactics must seek to configure the best possible combination of technology and structure to match with the observed risk level. Davis assumes that these elements are under the control of the project manager, and that he or she can shape them as suggested in the approach. Finally, McFarlan emphasizes risk items associated with inappropriate task specification. At the same time his risk resolution techniques aim at submitting the development organization to better managerial control. McFarlan thus advocates a managerial fix. He argues that any organizational structure in software development is malleable and can be reshaped by swift management action to reduce variations in the socio-technical system.

It is worth noting that the number of resolution techniques offered varies greatly in these approaches. Alter and Ginzberg's and Boehm's risk resolution techniques are more elaborate, but the level of detail and content of their techniques varies considerably and lacks systematic organization. For example, they use the same risk resolution technique to resolve many different risk items in different model components without any clarification of the rationale behind such a decision. This reflects their inherent assumption that some techniques are generic and can help manage a wider set of variations. Both Davis and McFarlan propose specific techniques which are applicable only to one risk item at a time. Interestingly neither of these contingency approaches offer resolution techniques to modify the task. This can be a symptom of a lack of systematic analysis, but it can also be seen as a different perception of the role of the software manager in which the task is given and the challenge is to adapt the rest of the socio-technical system to this task.

Heuristics. The heuristics proposed by each approach are simple and not thoroughly articulated. This simplicity reflects that the underlying theoretical models of the socio-technical system are fragmented and ambiguous. But the simplicity is also a reflection of practical needs. To be applicable, heuristics must be easy-to-use, simple, and allow for improvisation to meet the demands of varying situations. Two different structural formats are suggested for heuristics. Boehm and Alter and Ginzberg compile a list of risk resolution techniques for each observed risk item (like a menu list). No rationale is provided as to why these techniques are appropriate for a given risk item, or how one should choose between them. In this way, they embrace simple rules of thumb accepted on their face value, and adapted to the situation using experience, local knowledge, or intuition. In addition, these approaches provide no rules for how to compose a total set of risk resolution techniques given the observed risk item list. In contrast, the contingency models of Davis and McFarlan offer a relatively systematic procedure to compose a risk resolution strategy. Their approach is to abstract observed risky incidents into a specific risk configuration and then to provide an unambiguous heuristic for each observed configuration. These approaches too fail to provide theoretical reasoning behind the rules, although they exclusively use managerial uncertainty about the task to form a risk management strategy.

Scope. This feature deals with the applicability of the risk management approach across the development trajectory. Discrete approaches deal with situations at specific points of the software development trajectory. Continuous approaches cover larger segments of the trajectory. Again, we can observe differences between the approaches. Davis' and McFarlan's approaches are discrete. McFarlan's approach is used by functional and IT managers during the project initiation phase. This specific context explains the managerial focus on structural aspects before setting up the new development system. Davis addresses the task of avoiding incomplete or incorrect requirements and sees the technology and structure as controllable targets during the early phases of a project. In contrast, Boehm's and Alter and Ginzberg's approaches are continuous. Boehm's approach spans most parts of software development starting from requirement specification and ending up with configuring and implementing the software system. The intention is to provide new tools for project managers to complement their traditional method and tool kits. Alter and Ginzberg cover the whole development trajectory including organizational implementation of the system. Their scope is narrow, but the approach is meant to be used by all actors who have a stake in organizational acceptance of the system. In this way, it complements Boehm's approach.

5. Summary and Conclusions

We have addressed a number of concerns in software risk management. We point out that the concept of software risk is congruent with the behavioral concept of risk: managers engage in maneuvers in a complex and uncertain environment to avoid a major loss. From this point of view we interpret risk management approaches as articulations of specific causal dependencies that are believed to hold in development situations. These, in turn, are used to formulate heuristics, which managers can routinely deploy to master their environment. This they achieve by paying attention to specific features in the development situations that have a potential of creating a major loss. By doing so risk management approaches can expand organizational intelligence and make the organization more prepared for a wider spectrum of threats. We show that this interpretation of software risk has affinity with the concept of variation used in socio-technical models and risk management can thus be seen as analogical with managing change in a complex socio-technical system. Overall, the paper provides a new interpretation of software risk management which helps strip it off from a narrow system rationalism by suggesting a contingent, contextual, and multivariate view of software development and its risk.

Using a socio-technical model of organizational change we can analyze the content of risk management approaches in terms of: software risk items (what should one observe and monitor in the socio-technical system?), risk resolution techniques (what can one do to manage observed variations in the socio-technical system?), and heuristics (what is the format of rules which map software risk items into risk resolution techniques, i.e. what should one do when a specific incident is observed?). By doing so we can analyze the content of managerial attention shaping and the format of consequent managerial decisions and actions. As a result, by using the dimensions of the socio-technical model we synthesize a generic list of risk items and risk resolution techniques that cover an extensive set of aspects that can be heeded to by software managers. We deepen this analysis by investigating the contents and structure of four classical risk management approaches in terms of how they shape managerial attention. The analysis shows that their heuristics vary considerably in terms of their scope, content and format. Moreover, some approaches exclude specific aspects from managerial inquiry. This is an ideological choice in the sense that it is accepted without justification or empirical validation (Berger and Luckmann 1967, Hirschheim and Newman 1991, Trice and Beyer 1984). Because the four approaches differ considerably in their attention shaping protocold we conjecture that no single approach can address the pitfalls of not observing risky incidents during software development in a satisfactory manner. To use a system theoretical vocabulary: none of them generates enough requisite variety (Ashby 1956) as to control the environment and to distil information from it.

Therefore one way to expand the models and thereby expand management intelligence is to use a deductive approach. We can develop a more encompassing framework as suggested by the socio-technical model to systematically generate risk items and risk resolution techniques during risk identification. Our analysis also reveals that the four approaches adopt radically alternative formats and tactics to formulate heuristics. Some of them—like Boehm's or Alter and Ginzberg's model—offer fairly detailed and fine grained risk items and risk resolution techniques for management action—whereas Davis' and McFarlan's model offer few risk items and relatively few risk resolution techniques, which, however, consist of several items. It is likely that these formats engage management in different ways to risk management exercises with different outcomes. Future research is, however needed to develop a more systematic account of the benefits and weaknesses of different formats for risk management heuristics.

The four approaches we have analyzed can be combined to increase the variety generated by a risk management control system and thus to reshape the managerial attention. For example, Alter and Ginzberg's approach emphasizes a generic risk concern- actor related risks- which is largely missing in Boehm's approach. Therefore, these could be combined to yield more generic risk identification and resolution tactics using the socio-technical model as an organizing framework. In the same manner, Davis' approach and McFarlan's approach can be combined with Boehm's approach to cater for specific contexts in the risk driven spiral model (Boehm 1988). McFarlan's approach can be used before embarking on the spiral while thinking of the project set-up. Clearly, in such an environment task and structural issues dominate. Davis's approach can be used to plan and monitor the risks associated with the first cycle in the spiral for which Boehm does not offer any risk driven plan.

Overall, it is feasible to organize contingent risk management plans in which the development phases are organized as rows and the socio-technical model components as columns. By using values such as low, moderate, and high for each risk component we can develop risk profiles for each development phase. At the same time we can develop heuristics to select the risk resolution technique for the situation. Here we can use Appendix 1 as a basis. This framework can be used to direct managerial attention, to observe largely ignored risk areas and to orchestrate action. In this way the risk of management inattention as discussed in the introduction can be decreased. Another way of using the framework is to examine how the use of risk management techniques helps reshape management attention and thereby to change managers' local causal theories over time. To do so we would need also diagnostic instruments to measure managers' current ways to organize their experiences and to develop causal models of them. Generally a combination of first hand experience and the supply of *new* cognitive frames (risk management approaches) are likely to yield the best solutions for risk management considerations (Ropponen and Lyytinen 1997).

The proposed interpretation of risk management has its limitations. First, the socio-technical model adds an additional layer of theoretical complexity to an area, which does not necessarily need one. We felt however, that it is necessary to establish this theoretical structure in order to promote further progress in the field. In practice, the complexity can be overcome by crafting vocabularies that are closer to the every day experience of software managers.[9] The second limitation concerns the scope of our analysis. Though we achieved very high levels of reliability in our coding, and did not find the classification scheme to be too limited, the analysis technique must be tested with other risk management approaches to achieve higher levels of confidence of its coverage and applicability.

The paper has several implications for research and practice. The analysis suggests that risk management can and should be examined as an instance of organizational uncertainty absorption and management and that how managers read and interpret their environment has drastic impacts on development outcomes. This aspect has been touched upon here through concepts of attentions shaping, variation in socio-technical systems, and heuristics. Ultimately, we are dealing with problems of managing complex socio-technical systems where we can only achieve satisficing behavior (Simon 1979, 1983, Galbraith 1977) and where we have open-ended solutions due to the fact that the systems are organic and capable of learning. Therefore, one fruitful avenue is to explore in more depth risk management as satisficing behavior (Lyytinen et al. 1996). Another question is how organizations learn from risky incidents and how these can be inductively generalized into causal dependencies. Again, the socio-technical model can serve as a valuable asset in organizing, structuring, and validating such knowledge. Our conclusion from probing the four classical approaches is that their coverage is not complete and

[9]We have also used the socio-technical model to teach risk management and risk management approaches and to organize discussions of their project experiences. In these situations, however, project managers have had no difficulty to understand the model and they have appreciated the fact that it gives them a generic framework to organize their thinking about risk management which is broader than just some specific normative approach like Boehm's model (1991).

their items are not systematically inferred. Therefore, one research task ahead is to develop and organize a more encompassing risk item and risk resolution inventory that can be used to direct management thinking and attention. One step towards this end is presented in Appendix 1, but it still lacks heuristics which map risk items into a more systematic evaluations of development risk, and rules which state how such information can be used to choose risk resolution strategies. To this end one author has been engaged in developing a systematic risk management approach using the principles outlined in this paper as part of the EuroMethod—a standard developed by and for the European Commission for contracted software development (Euromethod 1996).

Our analysis also invites new types of empirical studies on risk management. First, the open system model can be used to conduct an empirical meta-analysis of how software initiatives have succeeded or failed with varying constellations of task, actor, structure, and technology risks. Second, we can use the framework to carry out rich ethnographic investigations on software risk management. In these studies we would describe and analyze contextual features such as personal anxiety, organizational protocols, and incentives and how they affect the scope and direction of risk management and attention shaping. Third, we can examine the structure of suggested intervention plans in order to understand how software managers seek to address observed risks at various stages of software development.

Finally, the paper conveys a useful message to practice. It argues that practitioners should be cautious with regard to the expected miracles of any risk management approach. Risk management approaches are neither complete nor risk-proof. Their value in the context of managing complex socio-technical change is that they put high premium on shaping management attention through learning new organizing schemes that help make sense of the development situations in new ways; and on the sane principles of management: be open, fear the worst, and honor the complexity of change.[10]

[10]This paper has largely benefited from Roy Schmidt's constructive criticisms, and literature suggestions. Thanks go also to John King, the Associate Editor, and three reviewers for their detailed and insightful suggestions of how to improve the paper.

Appendix 1: A Socio-Technical Model of Risk Management

Socio-Technical Component	Risks Item	Risks Resolution Technique
Task	*Task complexity* • project size • number of parties	*Reduce complexity* • divide tasks • requirements scrubbing
	Task uncertainty • ambiguity • task specificity • wrong functions • continuous change • existence of requirements	*Reduce uncertainty* • keep system simple • reduce the scope • use scenarios • use pilots to demonstrate system value • test the system *Manage process* • carefully plan and manage milestones and new releases
Structure	*Systems of communication* • inefficient • poor • lack of channels *Systems of authority* • inappropriate structure • poorly defined responsibilities • inappropriate rewards • inefficient governance structure *Systems of work flow* • unrealistic schedules • inappropriate work flow and coordination • poor physical arrangements	*Improve communications* • user participation • user surveys • team meetings • user lead teams • publicize participation results • monitor progress and promote open discussion • focus on critical task related topics *Reorganize* • project organization • external contracts and outsourcing • user committees and good relationships • formal procedures • user managed decisions and development • cost allocation structures *Change work flow* • pre-scheduling • cost and schedule estimation • incremental approach • path-analysis • risk-driven project planning • physical arrangements

(continued)

Appendix 1: (Continued)

Socio-Technical Component	Risks Item	Risks Resolution Technique
Actor	*Actor pitfalls* • lacking/variation • turnover • non-willing/ethical problems • poor or inappropriate beliefs, skills, and experience • political conflicts and power plays	*Improve actors* • staff with top talent • seek champions • cross training • morale building • user commitment • manage expectations • implementation games • training • role playing • study and screen potential actors
Technology	*Pitfalls in technology* • complexity • components unreliable • performance shortfalls • technical interfaces, • defects in quality • new and untried *Technological uncertainty* • high cost and non-adaptability • maintainability • extendibility	*Improve technologies* • specification standards and methods • task and organizational analysis techniques • information hiding/abstraction and modeling • bench marking • simulation/scenarios • prototyping
Task-Actor	*Inappropriate actors for a given task* • inability to specify or implement • goldplating	*Improve fit* • flexible governance structures • task matching • training
Task-Technology	*Inappropriate technology for a given task* • impossibility to implement or specify • poor performance • technology too expensive	*Improve fit* • contingency models for software development • manage technology options
Task-Structure	*Inappropriate structure for the task* • wrong project strategy • wrong control structure	*Change the task to fit the structure* • requirements scrubbing *Change the structure to fit the task* • adapt authority and decision structure • modify process model
Actor-Technology	*Incompetent/too competent actors for the given technology* • actor's experience • available computer science capabilities • gold plating • actors not willing to work with outdated/standard technology	*Improve fit* • prototyping, • technical analysis • scenario techniques • service assessment • technical training • hire top talent
Actor-Structure	*Lack of commitment* • wrong incentives • poor responsibilities • false beliefs and values • poor goals	*Gain management support* • apply appropriate leadership tactics • hire with good cooperation and management skills • install team building programs
Technology-Structure	*Inappropriate fit* • technology not aligned with authority and workflow, • structure not appropriate for technology	*Improve fit* • change authority or work flow • adopt/configure new organizational technologies

Appendix 2: Boehm's Software Risk Management Model
Boehm's model (1991) suggests a comprehensive set of steps and guidelines to manage software production risks. The basic model consists of six steps. The most important steps for our analysis are those where one identifies risks and identified risks are planned for (steps 1 and 4). For step 1 Boehm suggests a risk-identification checklist which gives the top 10 primary sources of risks. Other techniques for this step include assumption analysis, decision driver analysis, and decomposition, but these are not analyzed here as they do not increase decision-maker's understanding of the development domain. Proposed check lists are the main means to identify the most likely problems. Boehm's top ten list is exhibited in Table A2-1. Boehm also suggests more detailed checklists to analyze further each risk item and its probability (see Boehm 1991, p. 35–36). These lists are mostly based on software development handbooks from the U.S. Department of Defense. We shall not, however, explore these lists here further (though such an analysis could be done easily) as Boehm does not use them to identify major sources of risk.

Table A2-1 Boehm's Top-Ten Risk Item List

Risk Item	Coding
1. Personnel shortfalls	A
2. Unrealistic schedules and budgets	S
3. Developing the wrong functions and properties	Ta
4. Developing the wrong user interface	Ta
5. Gold-plating	A-Ta
6. Continuing stream of requirements changes	Ta
7. Shortfalls in externally furnished components	T
8. Shortfalls in externally performed tasks	S
9. Real-time performance shortfalls	T
10. Straining computer-science capabilities	A-T

For each risk item Boehm attaches a set of risk-management techniques that "have been most successful to date in avoiding and resolving the source of risk". The idea is that after detecting the most important risk items risk-managers can compile the associated set of risk management measures and plans. Boehm's list of risk management techniques for each risk item is illustrated in Table A2-2.

Table A2-2 Boehm's Risk Resolution Techniques

Risk Item	Risk Resolution Technique	Coding
1. Personnel shortfalls	1. Staffing with top talent	A
	2. Job-matching	A-S
	3. Team-building	A-S
	4. Morale building	A
	5. Cross-training	A
	6. Pre-scheduling	S

2. Unrealistic schedules and budgets	1. Detailed, multisource cost and schedule estimation	S Ta
	2. Design to cost	S
	3. Incremental development	S
	4. Software Re-use	Ta
	5. Requirements scrubbing	Ta
3. Developing the wrong functions and properties	1. Organizational analysis	T
	2. Mission analysis	T
	3. OPS-concept formulation	T
	4. User Surveys	T
	5. Prototyping	T
	6. Early user's manuals	T
4. Developing the wrong user interface	1. Task Analysis	T
	2. Prototyping	T
	3. Scenarios	T
	4. User characterization	T
5. Gold-plating	1. Requirements Scrubbing	Ta
	2. Prototyping	T
	3. Cost-benefit analysis	T
	4. Design to Cost	Ta
6. Continuing stream of requirements changes	1. High change threshold	S
	2. Information hiding	T
	3. Incremental development	S
7. Shortfalls in externally furnished components	1. Bench marking	T
	2. Inspections	T
	3. Reference checking	T
	4. Compatibility analysis	T
8. Shortfalls in externally performed tasks	1. Reference checking	T
	2. Pre-award audits	T
	3. Award-fee contracts	S
	4. Contracts	S
	5. Competitive design	S
	6. Prototyping	T
	7. Team building	A-S
9. Real-time performance shortfalls	1. Simulation	T
	2. Bench marking	T
	3. Modeling	T
	4. Prototyping	T
	5. Instrumentation	T
	6. Tuning	T
10. Straining computer-science capabilities	1. Technical analysis	T
	2. Cost-benefit analysis	T
	3. Prototyping	T
	4. Reference checking	T

Appendix 3: Davis' Model to Manage Requirements Specification Risks
Davis's model is concerned with selecting procedures that lead to complete and correct information requirements. He argues that one of the reasons for poor performance (and high risks) in systems development is that methods for obtaining and documenting user requirements are presented as general solutions rather than alternative methods for implementing a chosen strategy of requirements determination. Therefore he suggests "packaging" different methods into alternative strategies that are then carefully explained. These strategies are connected to a contingency framework (risk management model in our terminology) which suggests the most successful (least risky) requirements determination strategy for a given situation. The three risk items affecting the requirements specification are depicted in Table A3-1. Davis uses these to define the overall requirements process uncertainty (risk level). Davis's model has four strategies which are regarded as most effective on different levels of requirements process uncertainty. Each strategy basically embodies a different technology/structure combination which the risk manager should configure so as to affect the actors to obtain a good actor-task fit. Hence we get a coding of Davis' requirements specification strategies as shown in Table A3-2.

Table A3-1 Davis' Risk Items

Risk Items	Coding
1. Existence and stability of a set of usable requirements	Ta
2. Ability of users to specify requirements	A
3. Ability of analysts to elicit and evaluate requirements	A

Table A3-2 Davis' Risk Resolution Strategies

Requirements Determination Strategy	Coding
1. Asking from users	T-S
2. Deriving from existing systems	T-S
3. Synthesis from characteristics of the utilizing system	T-S
4. Discovering from experimentation	T-S

Appendix 4: Alter and Ginzberg's Implementation Risk Model
Alter and Ginzberg's implementation risk model focuses on problems associated with the organizational acceptance and implementation of the information system. They argue that implementing any system which leads to organizational acceptance will involve uncertainty from the managerial point of view. Therefore, these uncertainties should be detected and appropriate measures should be taken to minimize their impact. Overall they recognize several sources (factors) for organizational implementation uncertainty. These are depicted in Table A4-1.

Table A4-1 Alter and Ginzberg's List of Implementation Risk Factors

Risk Items	Coding
1. Designer lacking experience	A
2. Nonexistent or unwilling users	A
3. Multiple users or designers	A
4. Disappearing users, designers or maintainers	A
5. Lack or loss of support	A-S
6. Inability to specify the purpose or usage pattern in advance	A
7. Unpredictable impact	A
8. Technical or cost-effectiveness problems	T

In suggesting risk resolution strategies Alter and Ginzberg follow an approach similar to Boehm. For each risk factor they list a set of risk reduction strategies which are classified into inhibiting (I), or compensating (C) strategies. Inhibiting strategies are *ex ante* means to avoid a particular problem, i.e. to manage the environment to reduce the risk, while compensating strategies are *ex post* means to make up for a previous error or problem, i.e. to reduce the impact of the observed risk. Overall, Alter and Ginzberg's strategies lead to the classification shown in Table A4-2.

Table A4-2 Alter and Ginzberg's List of Risk Resolution Strategies

Risk item	Risk Reduction Strategy	Coding
1. Designer lacking experience	1. Use prototypes (C)	T
	2. Use evolutionary approach (C)	S
	3. Use modular approach (C)	T
	4. Keep the system simple (C)	Ta
2. Nonexistent or unwilling users	1. Hide complexity (C)	T
	2. Avoid change (C)	S
	3. Obtain user participation (I)	S
	4. Obtain user commitment (I)	A
	5. Obtain management support	A
	(C)	A
	6. Sell the system (I)	S
	7. Insist on mandatory use,	S
	8. Permit voluntary use (C)	S
	9. Rely on diffusion and exposure	
	(C)	
3. Multiple users or designers	1. Obtain user participation (C)	S
	2.Obtain user commitment (C)	A
	3. Obtain management support	A
	(C)	A
	4. Provide training programs (C)	S
	5. Permit voluntary user	S
	6.Rely on diffusion and experience (C)	Ta
	7. Tailor system to people's capabilities (C)	
4. Disappearing users, designers or maintainers	1. Obtain management support	A
	(C)	A
	2. Provide training programs (C)	A
	3. Provide ongoing assistance (C)	
5. Lack or loss of support	1. Obtain user participation (I)	S
	2. Obtain user commitment (I)	A
	3. Obtain management support	A
	(I)	A
	4. Sell the system (I)	S
	5. Permit voluntary use (C)	S
	6. Rely on diffusion and exposure	
	(C)	
6. Inability to specify the purpose or usage pattern in advance	1. Use prototypes (C)	T
	2. Use evolutionary approach (C)	S
	3. Use modular approach (C)	T
	4. Obtain user participation (I)	S
	5. Provide training programs (C)	A
7. Unpredictable impact	1. Use prototypes (I)	T
	2. Use evolutionary approach (I)	S
	3. Obtain user participation (I)	S
	4. Obtain management support	A
	(C)	A
	5. Sell the system (c)	
8. Technical or cost-effectiveness problems	1. Use prototypes (I)	T
	2. Use evolutionary approach (I)	S
	3. Use modular approach (C)	T
	5. Keep the system simple (I)	Ta

Appendix 5: McFarlan's Model of Managing a Portfolio of Projects

McFarlan proposes three major risk items (dimensions influencing the inherent risk) related to project organization and management as shown in Table A5-1.

Table A5-1 MacFarlan's Risk Items

Name of the Risk Items	Content of the Risk Item	Coding
1. Project size	Size in cost, time, staffing level, or number of affected parties	Ta
2. Experience with technology	Familiarity of the project team and the IS organization with the target technologies	A-T
3. Project Structure	How well structured is the project task	Ta

McFarlan classifies general methods (risk resolution techniques) for managing projects into four principal types:

External integration tools include organizational and other communication devices that link the project team's work to the users at both the managerial and the lower levels. See Table A5-2a.

Table A5-2a McFarlan's Risk Resolution Techniques for External Integration Tools

External Integration Tools	Coding
1. Selection of user as project manager	S
2. Creation of user steering committee	S
3. Frequency and depth of meetings of this committee	S
4. User-managed change control process	S
5. Frequency and detail of distribution of project team minutes to key users	S
6. Selection of users as team members	S
7. Formal user specification approval process	S
8. Progress reports prepared for corporate steering committee	S
9. Users responsible for education and installation of system	S
10. Users manage decisions on key action dates	S

Internal integration devices ensure that the team operates as an integrated unit. See Table A5-2b.

Table A5-2b McFarlan's Risk Resolution Techniques for Internal Integration Tools

Internal Integration Tools	Coding
1. Selection of experienced DP professional leadership team	A-S
2. Selection of manager to lead team	S
3. Frequent team meetings	S
4. Regular preparation and distribution of minutes on key design decisions	
5. Regular technical status reviews	S
6. Managed low turnover of team members	A
7. High percentage of team members with significant previous work	A-S
8. Participation of team members in goal setting and deadline establishment	S
9. Outside technical assistance	S

Formal planning tools help to structure the sequence of tasks in advance and estimate the time, money, and technical resources the team will need to execute them. See Table A5-2c.

Table A5-2c McFarlan's Risk Resolution Techniques for Formal Planning Tools

Formal Planning Tasks	Coding
1. PERT, critical path, etc., networking	T
2. Milestone phases selection	S
3. Systems specification standards	T
4. Feasibility study specifications	T
5. Project approval process	S
6. Project post audit procedures	T

Formal control mechanisms help managers evaluate progress and spot potential discrepancies so that corrective action can be taken. See Table A5-2d.

Table A5-2d McFarlan's Risk Resolution Techniques for Formal Control Tasks

Formal Control Tasks	Coding
1. Periodic formal status reports versus plan	T
2. Change control disciplines	T
3. Regular milestone presentation meetings	S
4. Deviations from plan	T

For each group a number tools (risk resolution techniques) are suggested.

Appendix 6: Description of the Coding and Analysis

General Description and Motivation

Categorical analysis is the process of identifying, coding and categorizing primary patterns in the data. In conventional qualitative research this means analyzing the content of observations, interviews, or some publicly available data (such as annual reports) to identify trends and solicit major patterns of meaning. In our case, the data set consisted of four articles describing risk management approaches. The goal for using categorical analysis was to develop a systematic representation of the four approaches by using socio-technical categories and thereby to reveal their varying foci and rationale. Our use of research (or consulting) articles as data sets is original though not unique (see e.g. Beath et al. 1994).

Coding Rules

The coding was conducted by classifying every risk item and risk resolution technique using socio-technical components. Before actual coding we agreed on a number of coding rules. Each risk item and risk resolution technique was assigned to one socio-technical coding scheme. The coding was based on the title of each item plus a careful reading of its content description. Sometimes this lead to a further reading of the item description in the main body of the text (if this was available) to further clarify its meaning. This was motivated by the observation that applying the name of the risk item or the resolution technique as a sole basis for coding information was insufficient. For example, McFarlan's risk item "Project structure" is denoted in our coding as Ta (and not S) because he defines this item as to deal with how well structured the project task is (and not the project structure). We also agreed initially to use relational codings such as T-A (read Technology-Actor relationship) in addition to pure component codings to characterize those situations where the source of risk was caused by a misfit between the components. This was motivated by Leavitt's (1964) observation that any organizational problem can cover one or several of the components thus adding the need to consider mutual relationships. An example of such a coding is McFarlan's second risk item "Experience with the technology" which is defined as familiarity with the target technologies. This amendment was necessary as few relational codings for both risk items and risk resolution techniques were detected during the coding. Overall, however, the number of relational codings was surprisingly low. These were counted as 0.5 while counting the frequencies of risk items in each Leavittian component.

The concern for the validity of distinctions (like between technology and structure) was another important concern in our effort. Because we used Leavitt's components as analytical devices rather than ontological categories we did not regard distinctions to be absolute (in terms of the focus of perception). This idea is also noted by Leavitt when he writes "Although I differentiate structural from technical from human approaches to organizational tasks, the differentiation is in points of relative weightings and underlying conceptions and values, not in the exclusion of all other variables" (Leavitt 1964, p. 56). For our classification task the major concern is therefore the discriminant validity of Leavitt's constructs.

We started with the above coding rules, but were open to add or modify the coding scheme should any such need arise. In particular, we were concerned with how environmental and cultural issues would be covered in our coding scheme, because the omission of these components has been one major criticism against his model. This caution was premature as we found only two risk items (7. "shortfalls in externally furnished components" and 8. "shortfalls in externally furnished tasks" in Boehm's risk item list) out of 24 which we could classify as environmental risks. Both of them could be easily interpreted either as technological risks (7) or structural risks (8), and we therefore did not add a new category "environment" to our coding. We could observe two risk resolution techniques out of 85 that dealt with cultural phenomena. These were Boehm's suggestions to use "team building" and "morale building" in resolving personnel shortfalls. (Team building is also suggested to reduce shortfalls in externally furnished tasks which suggests a structural reading of this technique). Because the number of such items was so low we decided to code both of them as A-S to simplify the coding scheme. In our understanding this does not violate a "correct" reading of them.

During the coding we had to deal also with possible multiple codings of the risk items or risk resolution techniques, i.e. the possibility that the item or the technique could cover two independent Leavittian categories thus signaling an ambiguous definition. We could find only one such instance. It dealt with Boehm's risk resolution technique "Competitive design or prototyping". In further analysis this was broken into two independent resolution techniques where the former dealt with the S component and the latter with the T component.

One concern in the coding is also the interrater reliability of the codings. In our case four risk items out of 24 (16%) were originally coded differently,[11] but with risk resolution techniques only one out of 82 (1%). In all the codings the level of consensus achieved was ca. 95% during the first round which is a very high coding reliability. In cases where differences in coding were observed the coders discussed their reasons for coding and tried to find a common understanding how the item should be coded. This was done by consulting once again the original text and interpreting the context of the term to ascribe a correct interpretation for the item. This resulted in codings on which all agreed and which are presented in Appendices 2 through 6.

Analysis
The two above steps were conducted independently for all articles by the authors. Obtained results were then compared for commonalities and differences to yield Table 1. We counted the frequencies of different types of Leavittian categories for both risk items and risk resolution techniques. These frequencies were then applied to determine the relative risk item and risk resolution focus.

References

Adler, S. "Risk-Making Management," *Business Horizons* 23, 2 (1980), 11–14.

Alter, S. and M. Ginzberg, "Managing Uncertainty in MIS Implementation," *Sloan Management Rev.*, Fall (1978), 23–31.

Andersen, N. E., F. Kensing, M. Lassen, J. Lundin, L. Mathiassen, A. Munk-Madsen and P. Sørgaard, *Professional Systems Development: Experience, Ideas, and Action*, Prentice-Hall, Englewood Cliffs, NJ, 1990.

Arrow, K. J., *Aspects of the Theory of Risk Bearing*, Yrjö Janssonin Säätiä, Helsinki, Finland, 1965.

Ashby, W. R., *An Introduction to Cybernetics*, Penguin Books, London, 1956.

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

Beath, C. M., "Strategies for Managing MIS Projects: A Transaction Cost Approach," *Proc. 4th International Conf. on Information Systems*, Houston, TX, 1983, 133–147.

——, "Managing the User Relationship in Information Systems Development Projects: A Transaction Governance Approach," *Proc. 8th International Conf. on Information Systems*, Pittsburgh, PA, 1987, 415–427.

—— and W. Orlikowski, "The Contradictory Structure of Systems Development Methodologies: Deconstructing the IS-User Relationship in Information Engineering," *Information Systems Res.*, 5, 4 (1994), 350–377.

[11]These were in Boehm's list: shortfalls in externally performed tasks, short-falls in externally furnished components, and straining computer-science capabilities; and Alter's and Ginzberg's list: lack or loss of support.

Bell, D. "Disappointment in Decision Making," *Oper. Res.*, 33 (1985), 1–27.

Berger, P. and T. Luckmann, *The Social Construction of Reality—A Treatise in the Sociology of Knowledge*, Penguin Books, London, 1967.

Beynon-Davis, P., "Information Systems Failure: The Case of LASCAD Project," *European J. of Information Systems*, 4, 3 (1995), 171–184.

Bjørn-Andersen, N. and L. Markus, Power Over Users: Its Exercise by Systems Professionals, *Comm. ACM*, 30, 6 (1993).

Blokdijk, A. and P. Blokdijk, *Planning and Design of Information Systems*, Academic Press, New York, 1987.

Boehm, B. W., "A Spiral Model of Software Development and Enhancement," *Computer*, May (1988), 61–71.

——, *Software Risk Management*, Tutorial, IEEE Computer Society Press, Los Calamitos, CA., 1989.

——, "Software Risk Management: Principles and Practices," *IEEE Software*, January (1991), 32–41.

—— and R. Ross, "Theory-W Software Project Management: Principles and Examples," *IEEE Trans. on Software Engineering*, 15, 7 (1989), 902–916.

Boland, R., *In Search for Management Information Systems: Explorations of Self and Organization*, Unpublished Working Paper, Weatherhead School of Management, Case Western Reserve University, Pittsburgh, PA, 1992.

Borum, F. and J. Christiansen, "Actors and Structure in IS Projects: What Makes Implementation Happen," *Scandinavian J. Management Studies*, (1993), 7(2), 102–130.

Bromiley, P. and S. Curley, "Individual Differences in Risk Taking," in J. F. Yates (Ed.), *Risk Taking Behavior*, Wiley, Chichester; 1992, 87–132.

Brooks, F., *The Mythical Man-Month*, Prentice-Hall, Englewood-Cliffs, NJ, 1974.

Burns, R. and A. Dennis, "Selecting an Appropriate Application Development Methodology," *Database*, Fall (1985), 19–23.

Caufield, C., *An Integrative Research Review of the Relationship Between the Technology and the Structure—A Meta-Analytic Synthesis*, Ph.D. Thesis, University of Iowa, 1989.

Charette, R. N., *Software Engineering Risk Analysis and Management*, McGraw-Hill, New York, 1989.

Ciborra, C. and G. Bracchi, "Systems Development and Auditing in Turbulent Contexts: Towards New Participative Approach," in E. M. Wysong et al. (Eds.), *Information Systems Auditing*, North-Holland, Amsterdam, 1983, 41–52.

—— and G. F. Lanzara, "Formative Contexts of Systems Design," in H. Klein et al. (Eds.), *Information Systems Development for Human Progress*, North-Holland, Amsterdam, 1987, 27–52.

Cooprider, J. G. and J. C. Henderson, "Technology—Process Fit: Perspectives on Achieving Prototyping Effectiveness," *J. Management Information Systems*, 7, 3 (1991), 67–87.

Curtis, B., H. Krasner, and N. Iscoe, "A Field Study of the Software Design Process for Large Systems," *Comm. ACM*, 31, 11 (1988), 1268–1287.

——, M. Kellner and J. Over, "Process Modeling," *Comm. ACM*, 35, 9 (1992), 75–90.

Cyert, R. and J. March, *A Behavioral Theory of the Firm*, Prentice-Hall, Englewood Cliffs, NJ, 1963.

Davis, G. B. "Strategies for Information Requirements Determination," *IBM Systems Journal*, 21, 1 (1982), 4–30.

—— and M. H. Olson, *Management Information Systems—Conceptual Foundations, Structure, and Development*, McGraw-Hill, New York, 1985.

——, A. S. Lee, K. R. Nickles, S. Chatterjee, R. Hartung, and Y. Wu, "Diagnosis of an Information System Failure: A Framework and Interpretive Process," *Information & Management*, 23, 2 (1992), 293–318.

DeMarco, T. and T. Lister, *Peopleware—Principles of Project Management*, Prentice-Hall, Englewood-Cliffs, NJ, 1987.

DeSalabert, A. and M. Newman, "Regaining Control: The Case of Spanish Air Traffic Control System-SACTA," in G. Doukidis et al. (Eds.), *Proc. 3rd European Conf. on Information Systems*, 1995, 1171–1180.

Earl, M., *Information Management Strategy*, Prentice-Hall, Englewood-Cliffs, NJ, 1987.

EuroMethod, *Euromethod Version 1*, European Commission, 1996. Information available at http://www.fast.de/Euromethod/.

Fairley, R., "Risk Management for Software Projects," *IEEE Software*, May (1994), 57–67.

Fischoff, B., S. Lichtenstein, P. Slovic, S. Derby, and R. Keeney, *Acceptable Risk*, Cambridge University Press, New York, 1981.

——, S. Watson, and C. Hope, "Defining Risk," *Policy Sciences*, 17 (1984), 123–139.

Galbraith, J., *Organization Design*, Addison-Wesley, Reading, MA, 1977.

Gersick, C., "Revolutionary Change Theories: A Multilevel Exploration of the Punctuated Equilibrium Paradigm," *Acad. Management Rev.*, 16, 1 (1991), 10–36.

Gervin, D., "Relationships Between Structure and Technology," in P. Nystrom et al. (Eds.), *Handbook of Organizational Design*, Oxford University Press, London, 1981, 3–38.

Ginzberg, M., "Early Diagnosis of MIS Implementation Failure: Promising Results and Unanswered Questions," *Management Sci.*, 27, 4 (1981), 459–478.

Grover, V., A. L. Lederer, and R. Sabherwal, "Recognizing the Politics of MIS," *Information & Management*, 14 (1988), 145–156.

Henderson, J. and S. Lee, "Managing I/S Design Teams: A Control Theories Perspective," *Management Sci.*, 38, 6 (1992), 757–777.

Hirschheim, R. and M. Newman, "Symbolism and Information Systems Development: Myth, Metaphor, Magic," *Information Systems Res.*, 2, 1 (1991), 29–62.

Humphrey, W., *Managing the Software Process*, Addison-Wesley, Reading, MA, 1989.

——, T. Snyder, and R. Willis, "Software Process Improvement at Hughes Aircraft," *IEEE Software*, 11, 7 (1991), 11–23.

Iivari, J., "A Paradigmatic Analysis of Contemporary Schools of IS Development," *European J. Information Systems*, 1, 4 (1991), 249–272.

Jarke, M. and K. Pohl, "Vision Driven System Engineering," in N. Prakash et al. (Eds.), *Information Systems Development Process*, North-Holland, Amsterdam, 1992, 3–20.

Jones, M. and G. Walsham, "The Limits of Knowable: Organizational and Design Knowledge in Systems Development," in K. Kendall et al. (Eds.), *The Impact of Computer Supported Technologies on Information Systems Development*, North-Holland, Amsterdam, 1992, 195–212.

Kahnemann, D. and A. Tversky, "Variants of Uncertainty," *Cognition*, 11 (1982), 143–157.

Keen, P. G. W., "Information Systems and Organizational Change," *Comm. ACM*, 24, 1 (1981), 24–33.

—— and S. Scott-Morton, *Decision-Support Systems: An Organizational Perspective*, Addison-Wesley, Reading, MA, 1978.

Keil, M., "Pulling the Plug: Software Project Management and the Problem of Project Escalation," *MIS Quarterly*, 19, 4 (1995), 421–448.

—— and R. Mixon, "Understanding Runaway IT Projects: Preliminary Results from a Program of Research Based on Escalation Theory," in *Proc. 27th HICSS*, IEEE Computer Society Press, Los Calamitos, CA., 1994.

Kendall, K., K. Lyytinen, and J. DeGross (Eds.), *The Impact of Computer Supported Technologies on Information Systems Development*, North-Holland, Amsterdam, 1992.

Kwon, T. H. and R. Zmud, "Unifying the Fragmented Models of Information Systems Implementation," (227–251) in R. Boland et al. (Eds.), *Critical Issues in Information Systems Research*, Wiley, Chichester, 1987.

Leavitt, H. J., "Applied Organization Change in Industry: Structural, Technical, and Human Approaches," in *New Perspectives in Organizational Research*, Wiley, Chichester, 1964, 55–71.

Lehtinen, E. and K. Lyytinen, "Seven Mortal Sins of Systems Work," (63–79) in P. Docherty et al. (Eds.), *Information Systems for Organizational Productivity—Participation and Beyond*, North-Holland, Amsterdam, 1986, 63–79.

Lucas, H., *Implementation—The Key to Successful Information Systems*, Columbia University Press, New York, 1981.

——, "Alternative Structures for the Management of Information Processing," in Goldberg et al. (Eds.), *On the Economics of Information Processing*, Vol. 2, Wiley, New York, 1982.

Lundeberg, M., G. Goldkuhl, and A. Nilsson, *Information Systems Development: A Systematic Approach*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

Lyytinen, K., "Different Perspectives on Information Systems: Problems and Solutions," *ACM Computing Surveys*, 19, 1 (1987), 5–44.

——, "Expectation Failure Concept and System's Analysts' View of Information System Failures: Results of an Exploratory Study," *Information & Management*, 14 (1988), 45–56.

—— and R. Hirschheim, "Information Systems Failures—A Survey and Classification of the Empirical Literature," *Oxford Surveys in Information Technology*, Oxford University Press, England, Vol. 4, 1987, 257–309.

——, L. Mathiassen, and J. Ropponen, "A Framework for Software Risk Management," *J. Information Technology*, 11, 4 (1996), 275–285.

MacCrimmon, K. and D. Wehrung, *Taking Risks: The Management of Uncertainty*, Free Press, New York, 1986.

March, J., "Variable Risk Preferences and Adaptive Aspirations," *J. Economic Behavior and Organizations*, (1988).

—— and Z. Shapira, "Managerial Perspectives on Risk and Risk-Taking," *Management Sci.*, 33 (1987).

——, L. Sproull L., and M. Tamuz, "Learning from Samples of One or Fewer," *Organization Sci.*, 2, 1 (1991), 1–13.

Markus L. "Power, Politics and MIS Implementation," *Comm. ACM*, 26 (1983), 430–444.

—— and D. Robey, "The Organizational Validity of Management Information Systems," *Human Relations*, 36, 3 (1983), 203–226.

—— and M. Keil, "If We Build It, They Will Come: Designing Information Systems that Users Want to Use," *Sloan Management Rev.* (1994), 11–25.

Mathiassen, L. and J. Stage, "The Principle of Limited Reduction in Software Design," *Information, Technology, and People*, 6, 2–3 (1992), 171–185.

McFarlan, W., "Portfolio Approach to Information Systems," *J. Systems Management*, January (1982), 12–19.

Mohr, L. B., *Explaining Organizational Behavior*, Jossey-Bass, San Francisco, 1982.

Mumford, E., *Designing Human Systems*, Manchester Business School, Manchester, U.K., 1983.

Neo, B. S. and K. Leong, "Managing Risks in Information Technology Projects: A Case Study of Tradenet," *J. Information Technology Management*, 15, 3 (1994) 20–45.

Newman, M. and F. Noble, "User Interaction as an Interaction Process: A Case Study," *Information Systems Res.*, 1, 1 (1990), 89–113.

—— and D. Robey, "A Social Process Model of User-Analyst Relationships," *MIS Quarterly*, 16, 2 (1992), 249–265.

Nidumolu, S., "The Effect of Coordination and Uncertainty on Software Project Performance: Residual Performance Risk as an Intervening Variable," *Information Systems Res.*, 6, 3 (1995), 191–219.

Orlikowski, W., "CASE Tools as Organizational Change: Investigating Incremental and Radical Changes in Systems Development," *MIS Quarterly*, 17, 3 (1993), 309–340.

Oz, E., "When Professional Standards are Lax—The Confirm Failure and its Lessons," *Comm. ACM*, 37, 10 (1994), 29–36.

Parnas, D. L. and P. Clements, "A Rational Design Process: How and Why to Fake It," *IEEE Trans. on Software Engineering*, 12, 2 (1986), 251–257.

Perrow, C., *Normal Accidents: Living with High-Risk Technologies*, Basic Books, New York, 1984.

Roberts, K. (Ed.), *New Challenges to Understanding Organizations*, MacMillan, New York, 1993.

Ropponen, J. and K. Lyytinen, "How Software Risk Management Can Improve Systems Development—An Explorative Survey," *European J. Information Systems*, 6, 1 (1997), 41–50.

Saarinen, T. and A. Vepsäläinen, "Managing Risk of System Implementation," in T. Saarinen: *Success of Information Systems—Evaluation of Development Projects and the Choice of Procurement and Implementation Strategies*, Ph. D. thesis, Helsinki School of Economics and Business Administration, Helsinki, Finland, 1993a, 91–117.

—— and ——, "Procurement Strategies for Information Systems," in T. Saarinen: *Success of Information Systems—Evaluation of Development Projects and the Choice of Procurement and Implementation Strategies*, Ph. D. thesis, Helsinki School of Economics and Business Administration, Helsinki, Finland, 1993b, 118–141.

Sabberwal, R. and D. Robey, "Reconciling Variance and Process Strategies for Studying Information Systems Development," *Information Systems Res.*, 6, 4 (1995), 303–323.

—— and J. Elam, "Overcoming Problems in Information Systems Development by Building and Sustaining Commitment," *Accounting, Management, and Information Technologies*, 5, 3/4 (1996), 283–309.

Schmidt, R., *Leadership Patterns and Information Technology Usage Patterns in Top Management Teams*, Ph. D. thesis, Indiana University, Bloomington, IN., 1992.

Schmidt, R., K. Lyytinen, M. Keil, P. Cule, "Identifying Software Project Risks: An International Delphi Study," submitted for publication, 1997.

Seely Brown, J. and P. Duguid, "Organizational Learning and Communities of Practice: Toward a Unified View of Working, Learning, and Innovation," *Organization Sci.*, 2, 1 (1991), 40–57.

Simon, H., "Rational Decision Making in Business Organizations," *American Econ. Rev.*, 69, 4 (1979), 493–513.

——, "Theories of Bounded Rationality," *Behavioral Economics and Business Organization*, MIT Press, Cambridge, MA (1983), 160–176.

Smolander, K., V.-P. Tahvanainen, and K. Lyytinen, "How to Combine Tools and Methods in Practice—A Field Study," in B. Steinholtz et al. (Eds.), *Proceedings of CAiSE'90*, Springer-Verlag, Berlin, 1990, 195–214.

Sørensen, C., *Adoption of CASE Tools—An Empirical Investigation*, Ph. D. thesis, Department of Mathematics and Computer Science, University of Aalborg, Aalborg, Denmark, 1993.

Swanson, B., "Organizational Designs for Software Maintenance," *Proc. 5th International Conf. on Information Systems*, Tucson, AZ, 1984, 73–81.

—— and C. M. Beath, *Software Maintenance*, Wiley, Chichester, 1990.

Starbuck, W., "Organizations as Action Generators," *American Sociological Rev.* 48, February (1983), 91–102.

Staw, B. and F. Fox, "Escalation: The Determinants of Commitment to a Chosen Course of Action," *Human Relations*, 30, 5 (1977), 431–450.

—— and J. Ross, "Behavior in Escalation Situations: Antecedents Prototypes and Solutions," *Research in Organizational Behavior*, 9 (1987), 39–78.

Thambain, H. J. and D. Wilemon, "Criteria for Controlling Projects According to Plan," *Project Management J.*, 6(2) (1986), 75–86.

Trice, H. and J. Beyer, "Studying Organizational Cultures through Rites and Ceremonials," *Acad. Management Rev.*, 9, 4 (1984), 653–669.

Turner, J., "A Comparison of the Process of Knowledge Elicitation with that of Requirements Determination," in W. W. Cotterman et al. (Eds.): *Challenges and Strategies for Research in Systems Development*, Wiley, New York, 1992, 415–430.

van Genuchten, M., "Why is Software Late? An Empirical Study of Reasons for Delay in Software Development," *IEEE Trans. on Software Engineering*, 17, 6 (1991), 582–590.

van Swede, D. and H. van Vliet, "Consistent Development: Results of a First Empirical Study in the Relation Between Project Scenario and Success," in G. Wijers et al. (Eds.): *Advanced Information Systems Engineering*, Lecture Notes in Computer Science Springer-Verlag, Berlin, 1994, 80–93.

Waters, R., "The Plan that Fell Earth," *Financial Times*, March 1993, 12.

Richard Boland, Associate Editor.

Weber, R., *Basic Content Analysis*, Sage Publications, Newbury Park, CA, 1985.

Willcocks, L. and H. Margetts, "Risk and Information Systems: Developing the Analysis," in L. Willcocks (Ed.): *Information Management: The Evaluation of Information Systems Investments*, Chapman-Hall, London, U.K., 1994, 208–230.

Williamson, O., *Markets and Hierarchies: Analysis and Antitrust Implications*, Free Press, New York, 1975.

Yates, J. F. (Ed.), *Risk Taking Behavior*, Wiley, Chichester, 1992.

Zmud, R., "Management of Large Software Development Efforts," *MIS Quarterly*, 4, 2 (1980), 45–55.