Risk Knowledge Capture in the Riskit Method

Jyrki Kontio and Victor R. Basili
jyrki.kontio@ntc.nokia.com / basili@cs.umd.edu
University of Maryland
Department of Computer Science
A.V.Williams Building
College Park, MD 20742, U.S.A.
http://www.cs.umd.edu/users/{jkontio, basili}/

Abstract

This paper describes how measurement data and experience can be captured for risk management purposes. The approach presented is a synthesis of the Riskit risk management method and the Experience Factory. In this paper we describe the main goals for risk knowledge capture and derive a classification of information based on those goals. We will describe the Riskit method and its integration with the Experience Factory. We will also outline the initial experiences we have gained from applying the proposed approach in practice.

1. Introduction

Unanticipated problems frequently cause major problems to projects, such as cost overruns, schedule delays, quality problems, and missing functionality. To some degree these problems can be seen as signs of immaturity of our field and we should expect some improvements in our discipline as our methods and knowledge improve. However, as each software development project involves at least some degree of uniqueness and our technology changes continuously, uncertainty about the end results will always accompany software development. While we cannot remove risks from software development, we should learn to manage them better.

Ability to capture, analyze and package experience is a prerequisite for systematic, planned improvements in software engineering [2], as in any field. The framework proposed in this paper builds upon the Riskit method and the Experience Factory, both developed at the University of Maryland. The proposed risk knowledge capture framework contains templates for capturing data about risk elements, templates for capturing relevant information about the risk management process, definition of where in the risk management process risk management knowledge is captured and utilized, and a proposed model for improvement goals for risk management.

2. Background

Risks in software development were not addressed in detail until late 1980's when Boehm [6] proposed and synthesized an approaches for software risk management. His work was complemented by Charette [9], and on these foundations recent advances in software risk management have produced well-documented approaches for risk management [14,18,24,26], several categories of risks have been identified [6,8,23], quantitative approaches for risk management have been proposed and used [5,7,11], and there are several software tools available for risk management. Furthermore, most commonly used software engineering standards [15,16] or assessment frameworks [17,27] require at least some form of risk management to take place.

Despite these efforts and the obvious industry interest in risk management, it seems that few organizations apply specific risks management methods actively [28]. The limited survey data from a recent workshop by Basili and Koji Tori supports this observation: only 20% of respondents claimed to use risk management techniques "extensively" while 40% stated that they are not using "any risk management techniques or approaches" [19]. Clearly, the industrial practice of risks management methods has not yet reached its full potential.

There is little reported work on utilizing data and experience from past project in software engineering risk management literature. Some aspects of Boehm's work implicitly assumed that data from past projects is available if simulation and cost models are used for estimating risks [6]. He also mentioned factors of cost models as possible risk monitoring metrics. Charette has presented an outline of items that should be defined for a project to initiate risk management [10]. He has also given examples of what should be measured and how this data can be graphed for risk management purposes. However, neither one of these approaches can be considered a systematic way to capture or utilize risk management experience.

The Software Engineering Institute (SEI) has collected data from risk assessments they have carried out during the last few years. Their goal seems to be to support analysis risks and their relationships using lexical analysis on the qualitative descriptions in the database [25]. It also seems that frequencies of risks in the database have been used to indicate what are the most common risks. To our knowledge, this database focuses on the results of risk assessments and contains little or no data of what actually happened in projects. Also, it is not clear how much context information is captured about risks and projects so that information in the database can be utilized more effectively.

Hall has defined and implemented a risk database while working at Harris corporation [12]. Risks from three projects were collected [13] and used for analysis in evaluating Hall's risk management maturity model. Hall has also collected survey data on the levels of risks management practices in various organizations [12].

There have been several other, less formal approaches in documenting information about software risks. The ACM SIGSOFT Software Engineering Notes has run a long series of reports on computer related problems or disasters. However, such a list is not very useful for analyzing risks of an individual projects as most of the reported risks do not contain enough context information and details to be useful.

In summary, it seems that while several some advances have been made in the area of software risk knowledge capture, none of the reported approaches provide a comprehensive framework for capturing risk knowledge. Furthermore, software risk management data and knowledge is rarely systematically collected and utilized in the industry. We hope that the framework proposed in this paper can act as a step towards more systematic risk knowledge capture so that our understanding of risks and risk management methods can improve.

3. Risk Knowledge Capture

We have identified three generic types of goals for risk knowledge capture: monitoring risks, understanding risks, and risk management process improvement. First, the risk situation in a project needs to be monitored so that appropriate risk controlling action can be taken. Second, we need to collect information about risks so that frequencies of occurrence and losses of risks can be estimated better. Finally, information needs to be collected so that the risk management process itself can be improved.

Each of the three goals described above focus on different kinds of information and, as always in measurement, the individual metrics and data collection procedures may vary between situations. However, we have identified some generic classes of information based on these three goals. This risk information classification will be introduced in the following paragraphs.

Project context information refers to such information that determines the circumstances and setting where the project is carried out. Project context information is relevant for all software engineering measurement data, but it is particularly important for risk management. The probability of a risk event is often influenced by many factors. By capturing as much as possible of the risk management context information we make it easier to interpret risk management data in the future.

The *risk management infrastructure information* defines what risk management methods, techniques, tools, processes and approaches are used for in risk management. The risk management infrastructure can also be extended to include several other organizational issues that marginally influence risk management, as proposed by Hall [12]. In fact Hall's framework can be used as a model to document the state of risk management infrastructure in an organization.

The *project information* defines the project itself and it includes the definition of the goals, customers, schedule, and constraints of the project. It also includes the definition of the risk management mandate for the project: the risk management mandate is a project-specific statement of the scope of risk management in a project.

	Risk monitoring	Understanding risks	Risk management process improvement
Project context information	X	X	X
Risk management infrastructure information			X
Project information		X	X
Enactment data	X	X	X
Risk management process information			X
Risk element information	X	X	X

Table 1: The relationships between risk knowledge capture goals and risk information types

While the project information provides a static view to the project, *enactment data* provides the dynamic perspective to the project: how much effort is spent, what artifacts are produced and when, how much time has passed, and which individuals worked on the project. Enactment data is usually collected for project control and experience capture purposes as a part of software engineering measurement program.

The *risk management process information* describes the activities and events related to risk management in the project. The risk management process information is, in fact, a special case of project information, but as it represents our special focus, it is meaningful to separate it from the general enactment data of the project.

Finally, *risk element information* refers to information about risks in a project. This type of information can include descriptions of factors that influence risks, such as methods, tools, resources; events that may influence the project; or impacts that risks might have. As we will discuss later, the Riskit method contains conceptual tools to structure such information more formally than is usually done.

The relationships between risk knowledge capture goals and risk information types is presented in Table 1. Each row in Table 1 represents a risk information type and each column a risk knowledge capture goal. An "X" in a cell indicates that the goal in that row normally needs to utilize the type of information listed in that row. However, it is important to point out that information from other categories may often be needed as well, Table 1 merely represents what we believe to be typical relationships between goals and information types.

4. Towards a Risk Knowledge Capture Framework

4.1 The Riskit Method

The Riskit method has been developed to support systematic risk analysis. The Riskit method uses a graphical formalism to support qualitative analysis of risk scenarios before quantification is attempted, its risk ranking approach can be selected based on the availability of history data or accuracy of estimates, it supports multiple goals and stakeholders, and its risk ranking approach is based on the utility theory [20]. We have presented an overview of the activities in the Riskit process in Figure 1. More information about the method is available in separate reports [20-22].

A central part of the Riskit method is the graphical formalism used to document risks, the *Riskit analysis graph.* The Riskit analysis graph is used to define the different aspects of risk explicitly and more formally than is done in casual conversation. The Riskit analysis graph is used during the Riskit process to decompose risks into clearly defined components, *risk elements*. Its components are presented in Figure 2. Each rectangle in the graph represents a risk element and each arrow describes the possible relationship between risk elements. We will define the components of the graph in the following paragraphs.

Instead of informal, general descriptions of risks, we can document the different aspects of risks more precisely, as is shown in Figure 2. The Riskit analysis graph allows explicit and more formal documentation of risks and risk scenarios.

The Riskit method has several potentially useful characteristics that can support risk knowledge capture. First, the Riskit Analysis Graph enforces more formal definition of risks so that more information is collected about each risk. Second, the graphical formalism used as well as the tool that is used to draw these diagrams lay the foundations for automating some of the risk knowledge capture: information about risks can be captured as Riskit graphs are drawn. Third, the Riskit process itself is a defined process that increases repeatability of the risk management process and supports the collection of relevant risk management experience through the templates and guidelines included in the method.

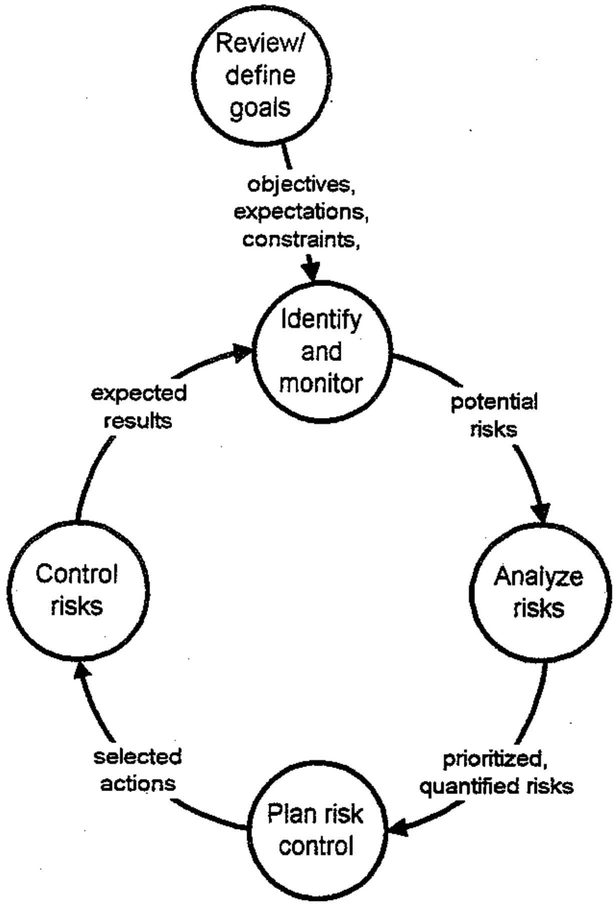

Figure 1: The Riskit risk management cycle[1]

4.2 Risk Knowledge Capture in the Experience Factory Framework

In this section we present how the Riskit method can be integrated into Basili's Experience Factory (EF) and Quality Improvement Paradigm (QIP) [3,4]. The Quality Improvement Paradigm (QIP) is a systematic process for continuous improvement. It is similar to the scientific principle of learning in its emphasis of learning through empirical experience. The QIP process can be seen as consisting of three main

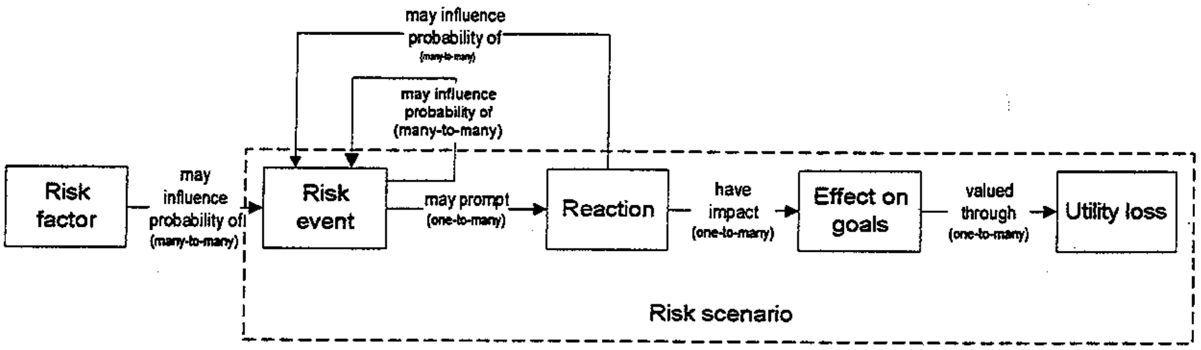
may influence probability of (many-to-many)
may influence probability of (many-to-many)
Risk factor
may influence probability of (many-to-many)
Risk event
may prompt (one-to-many)
Reaction
have impact (one-to-many)
Effect on goals
valued through (one-to-many)
Utility loss
Risk scenario

Figure 2: A conceptual view of the elements in the Riskit analysis graph

[1] Note that Figure 1 presents a simplified view of the activities in the Riskit process. More comprehensive description of the Riskit process is available through other publications [20].

activities that include the six steps normally described for QIP: *planning*, consisting of the steps characterize, set goals, and choose process; *execute*; and *learning*, consisting of steps analyze and package [4].

The Experience Factory Organization is an organizational model for implementing the QIP process. The main idea of this approach is the recognition the distinct roles belonging to the project organization and a learning organization, the Experience Factory. The Project Organization focuses on delivering the software product and the Experience Factory focuses on learning from experience and improving software development practice in the organization. A central aspect of the Experience Factory is the Experience Base, a repository of data and knowledge about the software development process and products. The knowledge in the Experience Base can be in various forms, it can include raw and summarized data, mathematical models about the data (e.g., prediction models), experiment reports, and qualitative lessons learned reports [1-4].

From risk management perspective the Experience Factory concept serves to fulfill the following goals:

- separation of responsibilities between risk management within projects and improving the risk management process itself and improving the understanding of risks;
- systematic capture and accumulation of risk management knowledge into the Experience Base;
- continuous learning from risk management experience through measurement, data collection, analysis and synthesis; and
- systematic reuse of accumulated risk management knowledge through packaging and dissemination of this knowledge.

When the Riskit process is viewed from the perspective of the Experience Factory and the QIP cycle, it is possible to identify steps where risk management process needs to be initiated to support the QIP process, as shown in Figure 3. The initial planning cycle represents the first cycle of the Riskit process, whereas the risk management cycle supporting the execute step support mainly project monitoring, i.e., risk monitoring and control. The learning step analyzes and packages the risk management experience gained through the process.

All of the QIP and Riskit activities represented in Figure 3 produce data about risk management that can be captured and stored in an experience base. We have defined a database definition for such information for the Riskit process. Furthermore, the project planning step in QIP also includes goal definition for risk understanding and risk management process improvement. These goals can introduce new data and experience capture needs that can be implemented as required. The learning step of QIP, and the two risk related activities associated with it, utilize the data and experience collected about risks and produce packaged, reusable pieces of risk knowledge to be stored in the Experience Base and utilized in future projects.

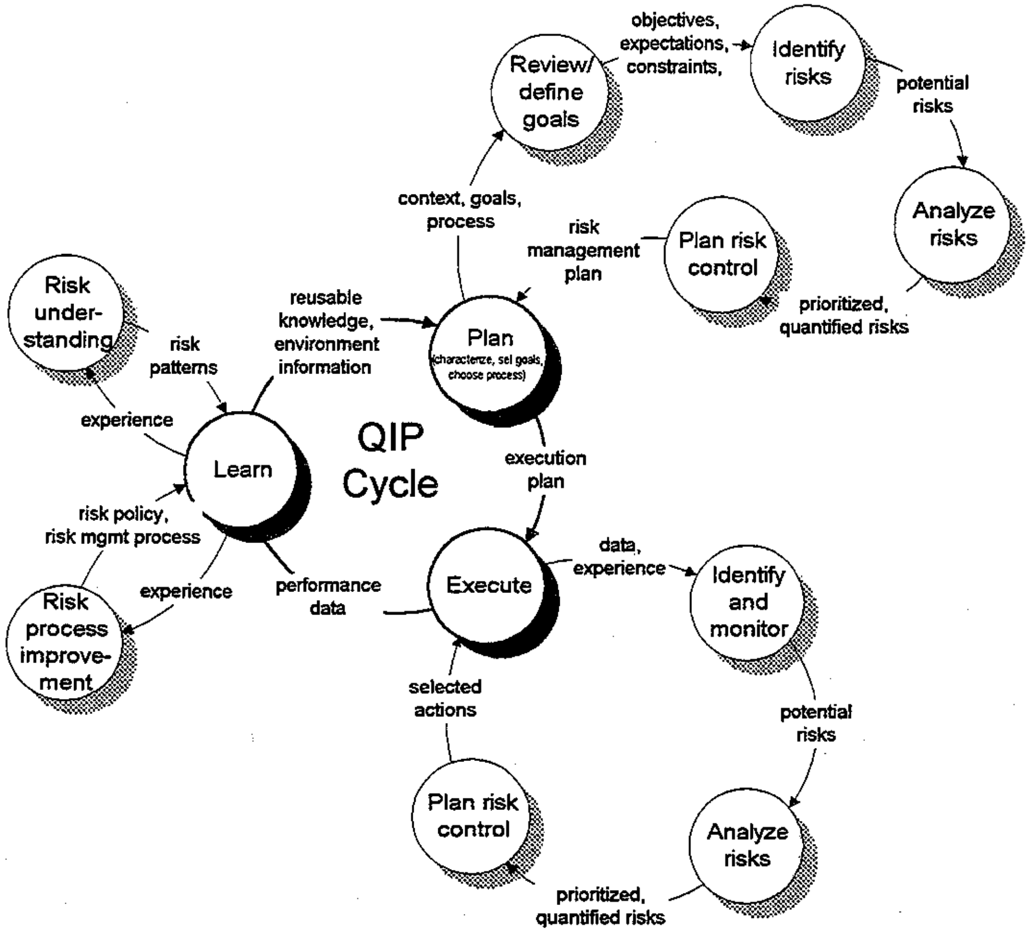

Figure 3: The mapping between QIP cycle and the Riskit process

4.3 Applying the Riskit Knowledge Capture Framework

The Riskit method and its knowledge capture framework have been applied in several trial projects. So far the case studies have focused on the last one of the goals we introduced earlier: improving the method itself.

The goals of the first case study [22] were to characterize the method, investigate its feasibility, and to collect empirical feedback on its use to be able to improve it. This first case study resulted in several changes in the method itself and it produced approximately 15 risk scenarios (corresponding to about 50 risk elements). Project and context information was documented informally in a separate report [22]. Other, on-going empirical studies with the method focus similarly on obtaining feedback on the methods feasibility and effectiveness.

These case studies have produced large amounts of risk management data and experience and we are in the process of formalizing this data into a risk management database, or a risk management experience base. Our goal is to evaluate the feasibility and potential benefits of such a database given the empirical data we have obtained.

5. Conclusions

This paper presented background and motivation for risk knowledge capture and proposed a classification of goals and information types for such capture. We also outlined how the Riskit method supports this type of experience capture. We reported some initial experiences from the use of the Riskit method and the proposed risk knowledge capture framework.

The potential benefits from risk knowledge capture are significant. Frequency and severity of typical risks can be estimated more accurately, changes in potential risks observed more concretely, risk management methods and tools can be improved based on empirical feedback, and projects have more up-to-date information about risks and risk management actions in a project. Furthermore, it may be possible to identify and package some risk management patterns: reusable pieces of risk management knowledge that can be utilized by project managers. Examples of such risk patterns could be lists of risks that are associated with certain project characteristics and descriptions of risk controlling actions that have been found effective in controlling certain types of risks. The Riskit method itself, through its more formal definition of risk and its graphical representation formalism, provides a good basis to capture and reuse such knowledge in practice.

While it is too early to make any conclusions about the feasibility and benefits of the proposed risk knowledge capture approach, the combination of Riskit and the Experience Factory contain the necessary foundations for more systematic and detailed experience capture. The initial empirical studies indicate that the approach is feasible in industrial context.

However, it is yet to be determined whether such experience capture is cost effective. Although the Riskit method may potentially allow automation of some of the experience capture processes, it is currently a manually driven process and therefore potentially too costly in large scale use. Furthermore, given the subjective nature of the definition of risk, one could also question how reliable is experience that, to a large degree, is based on subjective opinions and judgment calls about future events.

While there may be some valid concerns about the cost-effectiveness of a risk management database and its utilization, it is nevertheless likely that risk management experience needs to be captured and formulated into knowledge to be reused in future projects. The Riskit method provides a more concrete basis even for qualitative knowledge formulation process, even when the risk management experience and data are not captured into a formal database but stored in less formal parts of the Experience Base.

6. References

[1] V. R. Basili, Quantitative Evaluation of Software Engineering Methodology, 1985. Proceedings of the First Pan Pacific Computer Conference. Also available as computer science technical report TR-1519, University of Maryland.

[2] V. R. Basili, Software Development: A Paradigm for the Future, 1989. Proceedings of the 13th Annual Computer Software and Applications Conference (COMPSAC).

[3] V. R. Basili, G. Caldiera, F. McGarry, R. Pajerski, G. Page, and S. Waligora, The Software Engineering Laboratory - an Operational Software Experience Factory, pp. 370-381, 1992. Proceedings of the International Conference on Software Engineering, May 1992.

[4] V. R. Basili, G. Caldiera, and H. D. Rombach. The Experience Factory. In: *Encyclopedia of Software Engineering*, Anonymous New York: John Wiley & Sons, 1994.pp. 470-476.

[5] J. Berny and P. R. F. Townsend, Macrosimulation of project risks -- a practical way forward, *International Journal of Project Management*, vol. 11, pp. 201-208, 1993.

[6] B. W. Boehm. *Tutorial: Software Risk Management*, IEEE Computer Society Press, 1989. pp. 1-469.

[7] J. A. Bowers, Data for project risk analyses, *International Journal of Project Management*, vol. 12, pp. 9-16, 1994.

[8] M. J. Carr, S. L. Konda, I. A. Monarch, F. C. Ulrich, and C. F. Walker. *Taxonomy-Based Risk Identification, SEI Technical Report SEI-93-TR-006*, Pittsburgh, PA: Software Engineering Institute, 1993.

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

[10] R. N. Charette. *Applications Strategies for Risk Analysis*, New York: McGraw-Hill, 1990.

[11] R. Fairley, Risk Management for Software Projects, *IEEE Software*, vol. 11, pp. 57-67, 1994.

[12] E. M. Hall, Proactive Risk Management Methods for Software Engineering Excellence 1995. Florida Institute of Technology. Also available from UMI Dissertation Services.

[13] E. M. Hall, Email correspondence ed. J. Kontio. 1996. email correspondence.

[14] R. Hefner, Experience with Applying SEI's Risk Taxonomy, 1994. Proceedings of the Third SEI Conference on Software Risk Management. SEI. Pittsburgh, PA.

[15] IEEE. *IEEE Standard for Developing Software Life Cycle Processes*, New York: IEEE Computer Society, 1992.

[16] ISO. *ISO 9000-3, Guidelines for the application of ISO 9001 to the development, supply and maintenance of software, ISO 9000-3:1991(E)*, International Standards Organization, 1991.

[17] ISO. *SPICE: Baseline Practices Guide, an unfinished draft of a standard being developed for ISO, version 1.00*, 1994. (UnPub)

[18] D. W. Karolak. *Software Engineering Risk Management*, Washington, DC: IEEE, 1996.

[19] J. Kontio, IWSED-95 Web pages Anonymous. Anonymous. <*None Specified*>, vol. 1995. University of Maryland. World Wide Web. http://www.cs.umd.edu/projects/SoftEng/ESEG/iwsed/iwsed95/.

[20] J. Kontio, The Riskit Method for Software Risk Management, version 1.00 Anonymous1996. Computer Science Technical Reports. University of Maryland. College park, MD.

[21] J. Kontio and V. R. Basili, Empirical Evaluation of a Risk Management Method, 1997. Proceedings of the SEI Conference on Risk Management. Software Engineering Institute. Pittsburgh, PA.

[22] J. Kontio, H. Englund, and V. R. Basili, Experiences from an Exploratory Case Study with a Software Risk Management Method Anonymous CS-TR-3705, 1996. Computer Science Technical Reports. University of Maryland. College Park, Maryland.

[23] L. Laitinen, S. Kalliomäki, and K. Känsälä. *Ohjelmistoprojektien Riskitekijät, Tutkimusselostus N:o L-4*, Helsinki: VTT, Tietojenkäsittelytekniikan Laboratorio, 1993.

[24] J. V. Michaels. *Technical Risk Management*, Upper Saddle River, NJ: Prentice Hall, 1996.

[25] I. A. Monarch, S. L. Konda, and M. J. Carr, Software Engineering Risk Repository, 1996. Proceedings of the 1996 SEPG Conference. Software Engineering Institute. PIttsburgh, PA.

[26] G. Pandelios, T. P. Rumsey, and A. J. Dorofee, Using Risk Management for Software Process Improvement, 1996. Proceedings of the 1996 SEPG Conference. SEI. Pittsburgh.

[27] M. C. Paulk, B. Curtis, M. B. Chrissis, and C. V. Weber. *Capability Maturity Model for Software, Version 1.1, Technical Report SEI-93-TR-024*, Pittsburgh: Software Engineering Institute, Carnegie Mellon University, 1993.

[28] J. Ropponen, Risk Management in Information System Development Anonymous TR-3, 1993. Computer Science Reports. University of Jyväskylä, Department of Computer Science and Information Systems. Jyväskylä.