DRAFT

full paper published in:

proc. of 8th International Conference on Advanced Computer Systems

October 17-19, 2001 Mielno, Poland

Paper published in the proceedings and presented at the conference

Software support for collaborative risk management

Jakub Miler and Janusz Górski
Department of Applied Informatics, Technical University of Gdańsk, Gdańsk, Poland

Abstract: The paper recognises the increasing role of risk management in present software projects and aims at providing more support in this area. First we overview the objectives and processes of risk management with the particular stress on the need for effective and continuous communication. Then we develop a generic collaboration model that shows the roles and communication paths among the project members to support risk management activities as well as a data model showing the abstract data objects supporting the collaboration. We then propose a software architecture that supports the collaborative risk management in a project course. And finally we present a pilot implementation of a tool that facilitates collaborative risk management over Internet and the first results of its use.

Key words: software development, project risk management, collaboration, Internet

1. INTRODUCTION

The increasing competition in the market and the challenging expectations of the clients force software developers to reduce the overall development time and cost and to strive for increased product quality. At the same time the systems are getting larger and more complex and consequently the development processes tend to involve more people, often of different organisations, representing different competencies and working from distance locations. Effective management of such diverse groups of people requires great skill of project managers as well as the application of various supporting techniques and tools.

Despite of the progress in technology, the software engineering project management still faces similar problems as before [1]. The requirements are not defined precisely enough, change management is poor, the project scope and objectives are drifting and the turn over of the workforce is high. The result is that many (too many) projects are still overrunning their budgets, are delivering poor quality products and are delayed or, in extreme cases, even cancelled.

The real challenge in the present software development is effective risk management that would improve the success-to-failure ratio of software projects. This would require a deep change in the attitude of the project participants towards the way the project is conducted. The scope has to be broadened to account for alternative ways of the project development and to analyse various factors

that could affect the future events. The project members would have to develop a deeper understanding of project objectives to be able to recognise risks before they start to adversely influence the project course. All this calls for more intensive and more thorough communication within the project and outside the project boundaries. The need for more efficient communication is even higher if we recognise that present projects often involve multiple and diverse groups that have to co-operate from geographically separated locations using Internet as the primary communication medium.

The aim of this paper is to present a Web-based environment that supports risk management within a software development project. The tool supports effective information exchange about risks and covers risk identification, risk analysis and risk mitigation activities.

In the next section we give an overview of the risk management domain. Then we present simple communication and data models of risk management. Next two sections present the design and implementation of those models in the form of a Web-based co-operative workspace. The last section presents preliminary results of applying this tool to support a number of student projects.

2. OVERVIEW OF RISK MANAGEMENT

In general terms risk is defined as the possibility of loss, injury, damage or disadvantage. This definition describes two important features of risk [2, 11, 13]:

1. risk brings about negative consequences,
2. there is a degree of probability that these negative consequences come true.

Risk is formulated in the context of an undertaking, activity or opportunity (e.g. an investment or a project), because risks threaten the success of the undertaking meant by reaching the specified goals [2, 3, 13]. A project takes an opportunity to achieve success and create new value for the client, so it is automatically exposed to the risk of failure. No project is free of risk. A project without risk management recognises that there was a risk after the risk materialises as a real problem (e.g. it becomes obvious that the project overruns the budget or misses the deadline). Then the project team usually reacts and strives to minimise the negative consequences of the problem. As the rule, it is expensive and time-consuming. This price could have been much less if the risk were coped with before it converted to the problem. The lack of open communication, forward-looking attitude, team involvement in the management and the knowledge of typical problems, expose the project to a great risk of failure [4].

The essential factors of the project success are the quality, the time and the budget [13]. In the essential project aspects, the lack of risk management results in:

– schedule slippage,
– budget overrun,
– unsatisfactory quality of the product,
– failure to accomplish business goals of the project,
– disappointment of the employees and breakdown of their careers.

All of these failures refer to the primary project objectives, so they are unacceptable under ordinary circumstances. In some conditions, they may even be critical.

The only way to avoid serious consequences of a risk when it materialises is to catch the risk as early as possible and minimise its impact on the project. Though this advice sounds simple, it may be only realised through a defined set of activities focused on risk resolution. Altogether, it calls for a definition and elaboration of a systematic and explicit approach to risk management [9].

Software Engineering Institute (SEI) has elaborated a risk management approach that can be tailored to the specific environment of a software engineering project [4, 5, 11, 12, 13]. The generic

risk management methodology is enhanced and transformed into five recurring phases. The subsequent phases cover the risk management aspects:

– Risk assessment: identification, analysis.
– Risk mitigation: planning, tracking, controlling.
– Communication

The process starts with the identification of existing risks. Once the risks are identified, they are evaluated and prioritised and then appropriate corrective actions are planed and executed. To reach the acceptable level of success guarantee, the introduced actions must be controlled and the risks in mitigation must be continuously tracked for their status. This process continues all through the whole project schedule and across all the phases of development. Although the process is recurring, it may implement pipeline processing, when different phases of the risk management are executed in particular project areas. The SEI process model is shown in Fig. 1.

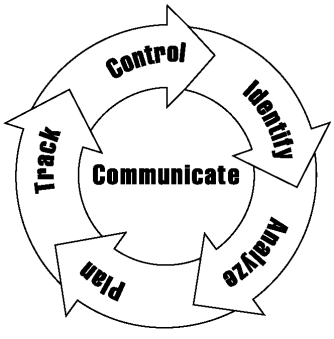

Figure 1. The SEI risk management paradigm

The risk management paradigm defines a set of continually performed activities that are based on communication among the participants. The cost and effort of these activities must be smaller than the estimated profit from the successful risk mitigation. To increase the 'rate of return', the risk management efficiency may be increased by means of computer-based tools.

3. COMMUNICATION MODEL OF RISK MANAGEMENT

The risk management process is a team activity involving all the project members, the customer representatives and the upper management of both the developer and the customer's organisation. To be both effective and profitable, this collective effort must be well organised, the roles must be defined and delegated, the means of communication established and the reporting standards set up. From the process point of view, the following roles can be distinguished:

a) Risk manager: he/she is responsible for facilitation of the risk management process. Analyses risk information, prioritises risks, prepares plans, receives tracking information and makes control decisions.
b) Information supplier: project members as well as the people involved from outside are acting as the information suppliers when they report the identified risks and the tracking information on the risks and the mitigation progress.
c) Information recipient: project members are acting as the information recipients when they receive the management plans and decisions.

d) Process sponsor: this is the project management role. It is responsible for assigning the other roles, establishing the communication and documentation infrastructure and encouraging the effort.

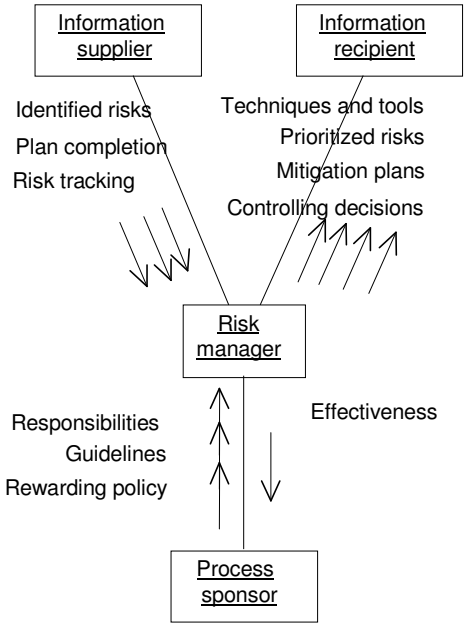

Figure 2. Communication model of risk management

Normally, a project member plays both, the supplier and recipient roles. Often, e.g. in case of technical-level risks, a person who identifies a risk can effectively mitigate it as well. In the general case, however, the information about risk has to be *communicated* if the risk is to be dealt with properly and in due time.

The resulting communication model is presented on Fig. 2.

4. DATA MODEL OF RISK MANAGEMENT

The data model of risk management underpins the activities of the risk management paradigm of Fig.1. Because of the space limitations we present here only a part of the model, that relates to the risk identification phase. It is shown in Fig. 3.

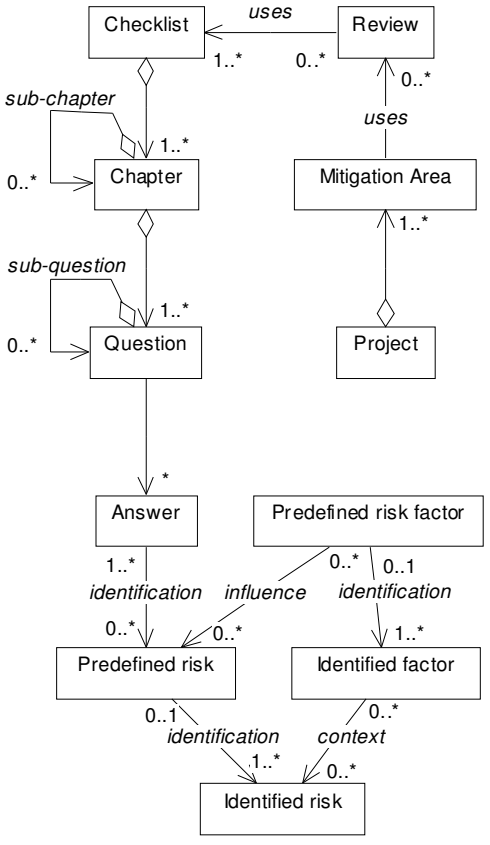

Figure 3. Risk identification data model

The model comprises the following elements:

- Project – General project description (process, methodology, organization, size, initiation date).
- Mitigation area – Area of a project that is exposed to a common type of risks (e.g. requirement specification, personnel management etc.)
- Review – This is the root object of the identification phase. Opening a new review starts risk identification activities whereas closing the review ends the risk information acquisition
- Checklist – Checklists are used to collect information that helps to identify risks. A checklist includes its name, description, authors identification and its components.
- Chapter – is a component of a checklist. It can be hierarchically decomposed into more fine structuring elements as shown in Fig. 3.
- Question – this is the lowest structuring level of a checklist (nevertheless it may include some sub-questions).
- Answer – represents the answer to a checklist question (it may be of different type: yes/no, range etc).
- Predefined risk – Risk that is stored in the risk knowledge base. It may be *selected* by one or more answers to the questions.
- Predefined risk factor – Risk factor providing the context for a risk stored in the risk knowledge base.

– Identified risk – Detailed risk description (from the risk knowledge base) in the context of a particular project. It is extracted from the knowledge base using the selection of predefined risks resulting from the answers to the checklist questions.
– Identified factor – Context of the identified risk extracted from the risk knowledge base.

Similar data structures were developed for the other phases of the risk management paradigm. The details can be found in [8]

5. SYSTEM ARCHITECTURE

The system architecture employs the client-server model with one central server and the user stations connected by a network. The hardware architecture is presented in Fig. 4.

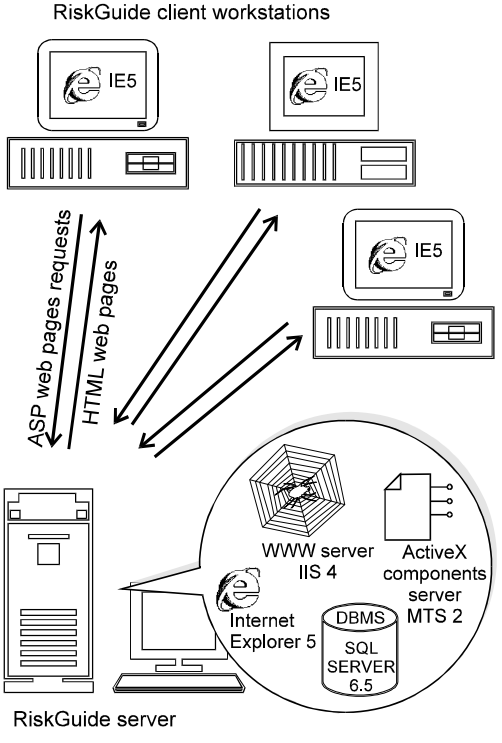

Figure 4. System architecture

The software architecture is formed upon the three-tier distributed application concept. The application is built in three levels: database, server components and interface. This concept is very useful in effective implementation of the applications accessed via Internet. The resulting application is well scalable, robust and portable. The general concept of the three-tier application is presented in Fig. 5.

The three tiers are highly independent. The system code modules are written individually for each tier. They communicate through program interfaces – function calls. As all the server applications for the tiers (SQL-Server, MTS, IIS, MSIE) are developed by the same vendor, no compatibility problems arise. The code was written using different programming languages, but nearly all data types were compatible and the values were automatically converted by the servers.

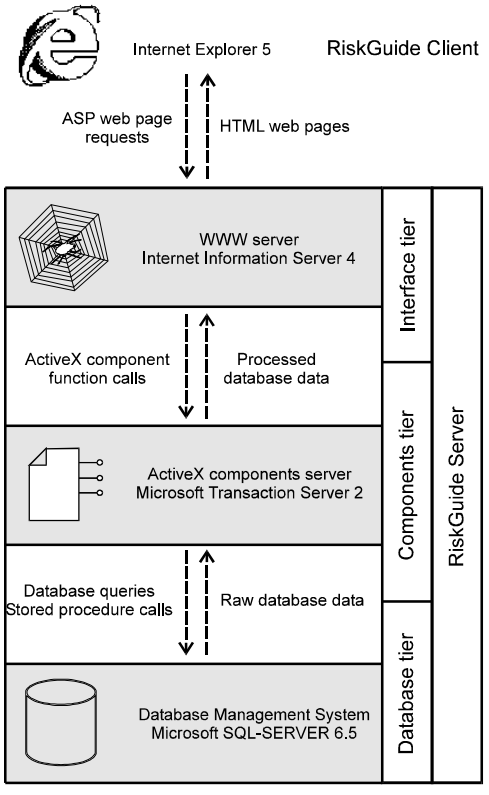

Figure 5. Software architecture

The advanced Internet technology used in the construction of our collaborative risk management tool led to the following features of the system:
- encapsulation of data processing,
- minimal software requirements for client workstations,
- multi-access,
- effective communication in distributed project teams,
- easy system maintenance,
- open user interface,
- short time of system development.

6. THE PILOT IMPLEMENTATION

A pilot implementation of the collaborative risk management tool called Risk-Guide has been accomplished to provide for some experiments. Risk-Guide supports the risk identification and analysis phases using checklists together with qualitative risk evaluation. The system offers a knowledge base of structured checklists and lists of common risks. It is designed to support multiple projects at a time with independent risk identification and analysis. Nevertheless, the knowledge base is shared by all those projects.

The checklists are accessed on-line and can be interactively answered by project members. To build the project risk repository the Risk Manager opens a review of the project and instructs the project members to browse the checklists and to provide answers to the questions. Risk-Guide

automatically searches the knowledge base and selects the risks that are implied by the answers. When the Risk Manager closes the current review, the system provides him/her with a list of risks that were identified during the review. The risks can then be individually assessed in three dimensions: possibility, severity and timeframe. For each dimension Risk-Guide offers a qualitative evaluation scale. After the individual assessment of risks is done, Risk-Guide calculates a list of risks ordered by priorities. The priorities are calculated from individual assessments using the risk evaluation matrix. Such a list can then be used to select the most dangerous risks for further mitigation.

As for now, the Risk-Guide knowledge base includes as the checklist the SEI Taxonomy-Based Questionnaire [12] and the Steve McConnell's Complete List of Schedule Risks [6, 7] as the list of risks. Nevertheless, due to the flexible data structures, it is easy to modify/extend the checklists and lists of risks as well as to modify their relationships. The present version of Risk-Guide is accessible on-line at [10].

7. THE EXPERIMENT

In the academic year 2000/2001, Risk-Guide was used by some 38 student groups as a supporting tool during the Software Engineering Project Management course at the Technical University of Gdansk. Each group comprised 3-4 members. The task of a group was to practice selected project management activities such as effort estimation and project planning. The size of the projects taken under consideration ranged from 3 man-months to 40 man-years. Risk-Guide was used by a group to support risk identification in relation to the preliminary project planning phase. The risks were then taken under consideration while building the detailed project plan and developing risk mitigation plans.

The details of this experiment are summarised in Table 1.

Table 1. Experiment constraints

No of projects	38
Experiment duration	10 weeks
Team size	3÷4 students
Project size	3 mm (3 persons / 1 month) ÷ 40.5 my (27 persons / 1.5 years) (22 projects – 10 ÷ 40 mm, 10 projects – 100 ÷ 250 mm, 6 other projects)
Phases	Planning, Design
Tasks	Prepare project strategy plan, identify project risks (answer TBQ questions, evaluate risks, select top 5 risks, describe top 5 risks, prepare mitigation and contingency plans for top 5 risks), elaborate detailed project plan
Products	Preliminary project plan, Risk management plan (list of identified risks (generated by Risk-Guide), risk evaluation, detailed description of top 5 risks, mitigation and contingency plans for top 5 risks), Detailed project plan

The experiment was then evaluated using a questionnaire that was distributed among the students to gather their opinions. The questions aimed at evaluation of the ease of use of Risk-Guide and assessing the effectiveness of its support. According to the collected answers (25 groups out of 38) and the opinions collected directly by the teaching staff, the experiment resulted in:

– better understanding of vulnerable project areas,
– bringing to the surface many potential problems that otherwise would have been omitted,
– increasing students' interest in risk management (increased awareness),
– elaboration of more realistic detailed plans,

– effective collaboration and parallel working,
– "common memory" of potential problems that can affect the future of the project.

In addition to the above many users stressed that the user interface of Risk-Guide was intuitive and user-friendly.

8. CONCLUSIONS

In the paper, we have argued that risk management is an important activity area in software development projects and that, because it is a team-oriented activity, it can benefit from tools that support communication and collaboration. As Internet is providing global interconnectivity and its use in businesses is rapidly increasing, such a tool should be Internet based and offered to a software development team disregarding the actual geographic dislocation of team members.

The paper introduced the idea of collaborative risk management and presented a tool that supports this concept. The tool has its pilot implementation that includes software project risk questionnaires and checklists that are publicly available. It has been applied during a number of student projects at our university. The preliminary results of those experiments are encouraging.

The further plans include:

– research in new ways of risk identification and representation including cause-effect chains (risk scenarios),
– research in new ways of overall project risk assessment from the assessments of individual risks,
– carrying out more case studies on real projects,
– further development of Risk-Guide towards a matured tool for collaborative risk management.

9. REFERENCES

[1] Brooks F. P., Jr., The Mythical Man-Month, Essays on Software Engineering, Anniversary Edition, Addison Wesley Longman Inc., 1995.
[2] Galagher B. P., Software Acquisition Risk Management Key Process Area (KPA) – A Guidebook Version 1.02, SEI report CMU/SEI-99-HB-001, Carnegie Mellon University, Pittsburgh PA, October 1999.
[3] Górski J., Risk Management, In: Project oriented software engineering, edited by J. Górski, Mikom, 2nd edition, 2000 (in Polish).
[4] Higuera R. P., Gluch D. P., Dorofee A. J., Murphy R. L., Walker J. A., Williams R. C., An Introduction to Team Risk Management, SEI report CMU/SEI--94-SR-01, Carnegie Mellon University, Pittsburgh PA, May 1994.
[5] Higuera R. P., Haimes Y. Y., Software Risk Management, SEI report CMU/SEI--96-TR-012, Carnegie Mellon University, Pittsburgh PA, June 1996.
[6] McConnell S., Code Complete, Microsoft Press, 1993.
[7] McConnell S., Rapid Development, Microsoft Press, 1996.
[8] Miler J., Computer system for supporting risk management in a software engineering project, Master's dissertation, Technical University of Gdańsk, Poland, 2000.
[9] Ould M. A., Strategies for Software Engineering, The management of risk and quality, John Wiley & Sons, Chichester, England, 1990.
[10] http://knot241.eti.pg.gda.pl/riskguide
[11] Rosenberg, L. H., Dr., Hammer T., Gallo A., Continuous Risk Management at NASA, presented at the Applied Software Measurement / Software Management Conference, San Jose, CA, February 1999.
[12] Sisti F. J., Joseph S., Software Risk Evaluation Method, SEI report CMU/SEI-94-TR-19, Carnegie Mellon University, Pittsburgh PA, December 1994.
[13] Van Scoy R. L., Software Development Risk: Opportunity, Not Problem, SEI report CMU/SEI-92-TR-30, Carnegie Mellon University, Pittsburgh PA, September 1992.