Intelligent Risk Management Tools for Software Development

John Dhlamini                Isaac Nhamu                             Admire Kachepa
School of Information Technology              School of Information Technology               School of Information Technology
Polytechnic of Namibia                      Polytechnic of Namibia                      Polytechnic of Namibia
P. Bag 13388                P. Bag 13388                P. Bag 13388
13 Storch Street              13 Storch Street              13 Storch Street
Windhoek, Namibia                        Windhoek, Namibia                         Windhoek, Namibia
Tel.:  +264 61 207 2725                     Tel.:  +264 61 207 2725                      Tel.:  +264 61 207 2725
jdhlamini@polytechnic.edu.na
[155x714]inhamu@polytechnic.edu.na
akachepa@polytechnic.edu.na

ABSTRACT

Software tools have been used in software development for a long time now. They are used for, among other things, performance analysis, testing and verification, debugging and building applications. Software tools can be very simple and lightweight, e.g. linkers, or very large and complex, e.g. computer-assisted software engineering (CASE) tools and integrated development environments (IDEs). Some tools support particular phases of the project cycle while others can be used with a specific software development model or technology. Some aspects of software development, like risk management, are done throughout the whole project from inception to commissioning. The aim of this paper is to demonstrate the need for an intelligent risk assessment and management tool for both agile or traditional (or their combination) methods in software development. The authors propose a model, whose development is subject of further research, which can be investigated for use in developing intelligent risk management tools

Categories and Subject Descriptors

Software Engineering : Project Management – risk assessment and management

General Terms

Risk Management

Keywords

Software development, risk assessment, software engineering tools, intelligent agents

1. INTRODUCTION

In the article by Deborah Hartmann [5], according to the CHAOS 2004 report, only 29% of projects were finished successfully; 18% of the projects failed without giving any delivery, and the other part, 53% of the projects were finished with overtime or over their budget. The report goes on to say that the causes for failure of software development projects relate to rapid technology changes, in-flux business requirements, or failure to attack risks.

The wasted revenue, argues Gopal Kapur [4], runs into trillions of dollars especially if one considers the lost opportunity costs which are not measurable.

According to Erdogmus [3], software development is rich in strategic opportunities, but it is subject to multiple sources and high levels of uncertainty. Since development costs are irrecoverable, it is important to manage the uncertainty. “Risk in itself is not bad; risk is essential to progress, and failure is often a key part of learning. But we must learn to balance the possible negative consequences of risk against the potential benefits of its associated opportunity” [20]. When managed properly, uncertainty creates value. For software development projects, like in other sectors, there are risk management supportive tools, developed for risk management activities. Most of these tools generally provide electronic risk repository to record and update risks, and making reports by compiling those records followed by analysis.

2. RESEARCH PROBLEM

Current risk assessment and management tools for software development projects are specific to the system or software development model used. There is need to develop tools which are system independent which can be used with any software development model (or combination thereof), agile or traditional.

To provide more effective ways of risk management, software tools, which are intelligent and adaptive to risk management strategies, are needed. The aim of this paper is to demonstrate, through literature and current technology review, the need for such tools. A description (model) of an intelligent risk management and assessment tool, and the eventual development of the tool, though an objective of later research, are not within the scope of this investigation.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SACLA '09, 29 June - 1 July, Mpekweni Beach Resort, South Africa
Copyright © 2009 ACM 978-1-60558-683-0/09/07 $10.00
3. RISK MANAGEMENT IN SOFTWARE DEVELOPMENT PROJECTS

3.1 Definition of Risk

Risk can be defined as the possibility of suffering loss. In a software development project, the loss could be in the form of diminished quality of the software product, increased development costs, delayed completion, or failure. The risks involved vary; from managing individuals and their emotions, managing resources, and managing the changing environment. These can be strategic, financial, operational, employee, political, or economic risks. Van Scoy [20] contends that technical risk lies at the heart of many problems causing the failure of software programs. He defines technical risk as “the possibility that the application of software engineering theory, principles, and techniques will fail to yield the right software product. Technical risk is comprised of the underlying technological factors that may cause the final product to be overly expensive, delivered late, or unacceptable to the customer.” Johnson [10] distinguishes between three types of events: problem, issue, and risk. A problem, according to Johnson, is an event that has happened and is having a negative effect on the project, while an issue is an event certain to happen and will have a negative effect on the project.

A risk, on the other hand, is given as an event that may or may not happen, but if it does, will have a negative effect on the project. The essence of risk classification is not in precisely defining risk categories, but in identifying and describing as many risks as possible on the project [2]. Project Management Body of Knowledge [14], interestingly, defines risk as “An uncertain event or condition that, if it occurs, has a positive or negative effect on a project objective.” Indeed, risk management is a project within a project.

3.2 Risk Management

3.2.1 What is Risk Management?

Wiegers [21] defines risk management as the application of appropriate tools and procedures to contain risk within acceptable limits by identifying, addressing, and eliminating potential problems before they damage a project. The Software Engineering Institute (SEI)’s [18] definition of risk management reads: “Continuous Risk Management is a software engineering practice with processes, methods, and tools for managing risks in a project. It provides a disciplined environment for proactive decision-making to: assess continuously what can go wrong (risk); determine what risks are important to deal with; implement strategies to deal with those risks.”

The SEI definition emphasizes the continuous aspect of risk management, hence the name Continuous Risk Management (CRM). A project with risk management aims at early identification and recognition of risks and then actively changes the course of actions to mitigate and reduce the risk[13].

The common perceptions in the above definitions are that risk management is a formalized, continuous process which is team-oriented (integrated) and requires open communication.

3.2.2 Approaches to Risk Management

Several approaches to software risk management have since been proposed and used in the software engineering context. However, despite of several studies and experiences published about risk management, the software industry, in a general way, does not seem to follow a model to analyze and control the risks through the development of their products [8]. According to Johnson [10] two approaches to software project management can be identified, traditional and risk-oriented. The traditional approach is reactive in nature and deals with problems generic to all software projects systemically and project specific problems as they arise. The later approach, however, is proactive as it seeks to identify and manage unique aspects of a specific project before they impact the project.

![Figure 1: Approaches to Project Management](image)

### 3.2.2.1 Software Engineering Institute (SEI)’s Software Risk Management (SRM) Methodologies

SEI’s SRM methodologies risk management framework for software risk management is supported by three groups of practices:

1. Software Risk Evaluation (SRE)
2. Continuous Risk Management (CRM)
3. Team Risk Management (TRM)

The goal of this framework is to enable engineers, managers, and other decision makers to identify, sufficiently early, the risks associated with software acquisition, development, integration, and deployment so that appropriate management and mitigation strategies can be developed on a timely basis.

The developed software risk methodologies have three fundamentally different, albeit complementary, objectives:

1. risk prevention
2. risk mitigation and correction
3. ensuring safe system failure
The following seven risk management principles are instrumental in the quest to achieve these three objectives [7]:

**Shared product vision**
- sharing product vision based upon common purpose, shared ownership, and collective commitment
- focusing on results

**Teamwork**
- working cooperatively to achieve a common goal
- pooling talent, skills, and knowledge

**Global perspective**
- viewing software development within the context of the larger system-level definition, design, and development
- recognizing both the potential value of opportunity and the potential impact of adverse effects, such as cost overrun, time delay, or failure to meet product specifications

**Forward-looking view**
- thinking toward tomorrow, identifying uncertainties, anticipating potential outcomes
- managing project resources and activities while anticipating uncertainties

**Open communication**
- encouraging the free flow of information between all project levels
- enabling formal, informal, and impromptu communication
- using consensus-based process that values the individual voice (bringing unique knowledge and insight to identifying and managing risk)

**Integrated management**
- making risk management an integral and vital part of project management
- adapting risk management methods and tools to a project’s infrastructure and culture

**Continuous process**
- maintaining constant vigilance
- identifying and managing risks routinely throughout all phases of the project’s life cycle

### 3.2.2.2 Capability Maturity Model Integration (CMMI)

CMMI in software engineering and organizational development is a process improvement approach that provides organizations with the essential elements for effective process improvement. CMMI was developed by a group of experts from industry, government, and the Software Engineering Institute (SEI).

CMMI currently addresses three areas of interest:

1. **Product and service development** - CMMI for Development (CMMI-DEV),
2. **Service establishment, management, and delivery** - CMMI for Services (CMMI-SVC), and
3. **Product and service acquisition** - CMMI for Acquisition (CMMI-ACQ).

Risk Management (RSKM) is a project management process area at maturity level 3. Its purpose is to identify potential problems before they occur so that risk-handling activities can be planned and invoked as needed across the life of the product or project to mitigate adverse impacts on achieving objectives.

Risk Management may initially focus simply on risk identification for awareness, and react to the realization of these risks as they occur. The Risk Management process area describes an evolution of these
specific practices to systematically plan, anticipate, and mitigate risks to proactively minimize their impact on the project.

3.2.2.3 Project Management Body of Knowledge (PMBOK)

PMBOK [16], by the Project Management Institute (PMI), is a project management guide, and an internationally recognized standard, that provides the fundamentals of project management as they apply to a wide range of projects, including construction, software, engineering, automotive, etc. According to this guide, risk management comprises a number of processes which are:

- Risk Management Planning
- Risk Identification
- Qualitative Risk Analysis
- Quantitative Risk Analysis
- Risk Response Planning
- Risk Monitoring and Control

Risk Management Planning
- Deliverable is the Risk Management Plan

Risk Identification
- Risk categories:
  - technical
  - project management
  - organizational
  - external

Qualitative Risk Analysis
- Define probability and consequences
- Data gathering
- Impact by objective
- Assumptions testing
- Data precision ranking

Quantitative Risk Analysis
- Individual and project risk
- Probability distributions
- Sensitivity and decision tree analysis
- Simulation methods

Risk Response Planning
- Responses should be:
  - Appropriate
  - Cost effective
  - Timely, realistic
  - Agreed (funded)

Risk Monitoring and Control
- Ongoing, continuous action
- Risks monitored
- New risks identified
- Effectiveness of risk management evaluated

A large number of processes have been generated in recent years to address the need for more effective risk management. According to Higuera and Haimes [7], the need to manage risk increases with system complexity. As the complexity of the system increases, both technical and non-technical (cost and schedule) risks increase. There is, therefore, an increasing need for more systematic methods and tools to supplement individual knowledge, judgment, and experience. Human traits (without tools), they argue, are often sufficient only at addressing less complex risks. They also note that many managers believe that they are managing risk in its multifaceted dimensions while the fact of the matter is that they are merely managing cost and schedule along with isolated cases of technical risk.

While agile methodologies have gained considerable industry attention, there exists little evidence that the practices recommended reduce risks.

3.3 Examples of Risk Management Tools

3.3.1 Riskit

Proposed in 1996 by Professor Jyrki Kontio, when he was a Researcher at the University of Maryland (UMD). It is a comprehensive risk management method based on theoretical principles with a comprehensive process definition that supports risk management activities.

3.3.1.1 The Riskit Process

Main characteristics of a Riskit process:
- Full operational definition of the process
- Risk management, scope, focus, authority and procedures defined together
- A specific step for identifying and defining the goals of the project

3.3.1.2 Risk Elements in Riskit

- Risk factor: a characteristic that affects the probability of a negative event occurring.
- Risk event: a stochastic phenomenon that represents an occurrence of a negative incident.
- Risk outcome: represents the situation after the risk event has occurred and before any corrective action.
- Risk reaction: a possible action as a response to risk event and resulting risk outcome.
- Risk effect set: the final impact of a risk event to the project. Considering the impact of reaction, it describes characteristics which were affected.

3.2.2.4 Other Approaches

Kimer and Concalves [8] present a model of risk management process (GRisk-Model), that covers all the stages of the software development process. The GRisk-Model was proposed with basis on the literature and from the experience of managers and senior software engineers of Brazilian software factories.
Figure 4: The Riskit Process

3.3.1.3 Steps in Riskit

1. Risk management mandate definition
   - the scope and frequency of RM are defined.
   - all relevant stakeholders are recognized.
   - output is the risk management mandate (why, what, when, who, how and for whom)

2. Goal review
   - the stated goals of the project are reviewed and refined, and implicit goals and constraints are defined explicitly.
   - stakeholders’ associations with the goals are analyzed.
   - output is explicit goal definitions

3. Risk identification
   - potential threats to the project are identified using multiple approaches.
   - output is a list of “raw” risks

4. Risk analysis
   - risks are classified and consolidated.
   - risk effects for all risk scenarios are estimated.
   - probabilities and utility losses of risk scenarios are estimated.
   - output is a completed risk analysis graph for all analyzed risks and ranked risk scenarios

5. Risk control planning
   - the most important risks are selected for risk control planning.

   - risk controlling actions for those important risks are proposed.
   - risk controlling actions are selected to be implemented.
   - output is selected risk controlling actions

6. Risk control
   - risk controlling actions are implemented.
   - output is reduced risks

7. Risk monitoring
   - the risk situation is monitored.
   - output is risk status information

3.3.2 Risk Guide

Risk Guide [14], implemented at the Politechnika Gdanska, is an Internet application and can be accessed simply by a web browser. It makes it applicable in distributed software projects. It supports risk reviews, indications, snapshots and reports to publish assessment results. Multiple project members can post risk indications simultaneously and those are then automatically lined up in the risk repository. The tool can support multiple projects at a time with independent risk identification and assessment processes. The system offers a knowledge base of checklists and the lists of common risks or complete list of schedule risks.

Three techniques to identify risks are supported:
• automatic generation of risk indications based on the answers to a questionnaire,
• explicit selection of a risk from a list of risks,
• supplying a new definition of a specific risk identified by intuition and/or engineering judgement.

The tool supports management of checklists and lists of risks. Once the analysis is completed, the resulting list of the most important risks can be published in a risk assessment report (see Figure 5).

Figure 5: Example of Risk Assessment Report
3.3.3 Risk Radar Enterprise (RRE)

RRE, by American Systems, is a commercial web-based application for enterprise-wide project risk management using MS Access database. It enables management and communication of project costs, schedule, technical and performance risk within a common enterprise framework. The vendor claims that RRE gives managers and their teams the visibility they need to proactively identify, analyze, track, control, mitigate, and report risk. It is said to support guidance from the PMI PMBOK and SEI CMMI.

![Figure 6: Example of RRE Risk State Screen](image)

4. The Gap

A large number of processes have been generated in recent years to address the need for more effective risk management. The risk management process provided in the PMBOK is a good overview of the typical processes, yet it is often too generic to meet the specific needs of software projects [11]. PMBOK and CMMI say that risk management is the project manager’s job. In practice, however, the project manager often digs himself so deep into the project that he is unable to have a “bird’s eye view” of the whole project. It is for this reason that risk management is often forgotten. Identified below are three areas where current risk management tools have noticeable shortcomings, namely: data models on which they are based; strong dependence on open communication; need for strong risk conceptualisation.

4.1 Data Model

Contemporary tools in risk assessment and management can be divided into two broad categories: repository-based management and knowledge-based risk assessment and management tools.

The first category is the one strongly represented with vendors asserting that they can handle a myriad of risks irrespective of scale or field of project development. Most of these tools generally provide electronic risk repository to record and update risks, and making reports by compiling those records followed by analysis. Such tools support a more reactive approach than a proactive approach to risk management. Examples are ARM-IR (Analytical Risk Management-Infrastructure Resilience Software Tool), Risky Project.

The second category of tools is based on the premise that risk assessment and management is a specialised field with its own specific demands for knowledge representation, retrieval and update. Information is thus kept in a repository (knowledge base) and the interface to this knowledge is the inference engine. As opposed to electronic repository systems in the first category, in general, a knowledge base is not a static collection of information, but a dynamic resource that may itself have the capacity to learn. This goes a long way in capturing project managers’ experiences and knowledge (tacit or otherwise) [6]. An example of a knowledge base tool is Risk Guide.

4.2 Open Communication

Miler and Górski [14] recognized effective, continuous and open communication as the prerequisite for successful risk management. There is need, therefore, to provide the project stakeholders a broad and highly available communication channel through which they can communicate risk-related information. Open and unrestricted communication facilitates the key activities related to risk management. Such a channel should be able to “absorb” information generated by using diverse identification techniques such as checklists, questionnaires, brainstorming sessions and individual observations. Moreover, it should be constantly open to protect against the risk-related information being lost.

Overdependence on such a communication channel shifts focus to the identification of communicated or documented risks which might not necessarily the risks inherent in the project activities. These ‘latent’ risks might actually be the ones to lead to project failure at a later stage. There is, therefore, need for intelligent tools which would capture such risks early without leaving everything to intra-team communication.

4.3 Conceptualisation

Current approaches to risk management require the visualization and formalization of risk information. Kontio et al [9] observed that an essential element of software engineering risk management is the conceptualization of potential risks to a project. Conceptualisation, they argue, is the basis of risk analysis and, even more importantly, it strongly influences how risks are communicated and understood by participants in a project.

Their study indicated that a defined and sufficiently expressive visualization approach can help capture more of the risk information than less formal methods.

Licorish [12] also notes that social risks (risks inherent in human collaboration) are not adequately addressed by standard risk management theories and that evaluation reveals that these risks are also not considered by existing software tools.

5. INTELLIGENT RISK MANAGEMENT TOOLS

Intelligence (natural or artificial) requires the ability to learn, which is the ability to acquire knowledge and then use to use that knowledge to effect a change in behaviour. Two of the major weaknesses of repository-based risk management tools is their lack of deductive power and that they tend to be too generic. This greatly reduces their effectiveness in software development projects where each project is unique. Knowledge-based tools, on the other hand, while exhibiting deductive capability, are tied to special technology or development method-
ologies. This reduces their independence from the technology and systems in their environment. In addition, in practice, software development is a hybrid of known, and often times not-so-known processes.

There is need, therefore, of risk management tools which are intelligent and independent of the software development methodologies and systems technology. Such tools have the ability to learn and change behaviour depending on what exactly transpires during a project’s life cycle. Further more, use of such a tool in future projects ensures continuity in the use of experience in risk management from previous projects.

Apart from the knowledge-based risk management tools which are based on artificial intelligence, there exist some tools based on mathematical modelling. Yacoub and Ammar [22] described a heuristic risk assessment methodology that uses dynamic metrics obtained from Unified Modeling Language (UML) specifications to determine the most risky components of the software architecture. It is mathematical analysis models derived from the UML diagram, and enables more attention to be placed in the areas of the system with highest risk.

6. PROPOSED FRAMEWORKS

6.1 Neural Network Approach

An artificial neural network (ANN) is an interconnected group of artificial neurons that uses a mathematical or computational model for information processing based on a connectionistic approach to computation. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network. A neural network stores information in much the same ways as the human brain, i.e. each piece of data is inter-related to each and every other piece.

Neural networks are non-linear statistical data modeling or decision-making tools that can be used to model complex relationships between inputs and outputs or to find patterns in data. Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by humans or other computer techniques.

This approach has been successfully applied in modeling of tools for customer and supply chain risk management[19]. Neural networks are ideal for handling large amounts of data and recognising patterns within complicated, vague or incomplete information [15]. This facilitates the identification of meaningful and predictive trends and patterns within data, and highlights anomalies that may be indicative of risk.

6.2 Intelligent Agents

According to Webster’s dictionary an agent is:

1. a person or thing that acts or brings about a certain result
2. one who is empowered to act for another

Russel and Norvig [17] define an agent as an entity that can be viewed as perceiving its environment through sensors and acting upon its environment through effectors. There is a consensus that autonomy, the ability to act without the intervention of humans or other systems, is a key feature of an agent.

Various authors have proposed different definitions of agents, these commonly include concepts such as

- **persistence** (code is not executed on demand but runs continuously and decides for itself when it should perform some activity)
- **autonomy** (agents have capabilities of task selection, prioritization, goal-directed behaviour, decision-making without human intervention)
- **social ability** (agents are able to engage other components through some sort of communication and coordination, they may collaborate on a task)
- **reactivity** (agents perceive the context in which they operate and react to it appropriately).

Capabilities of intelligent agents include:

- **ability to adapt**
  Adaptation implies sensing the environment and reconfiguring in response.
- **ability to learn**
  Learning can be through trial-and-error or by example and generalization

**Autonomous agents**

Autonomous agents are software agents that are autonomous, being self-contained and capable of making independent decisions, and taking actions to satisfy internal goals based upon their perceived environment.

**Distributed agents**

Very loosely coupled agents that can be executed as independent threads and on distributed processors.

**Fuzzy agents**

A fuzzy agent is a software agent that implements fuzzy logic which interacts with its environment through an adaptive rule-base.

Use of intelligent agents in a risk management tool will foster the application of the integration of formal methods and heuristic approaches to ensure support for the evaluation, comparison, analysis, and evolution of agent behaviour. Furthermore, if mobility is added to the agents, then one has a tool which is not confined to the realms of one project or geographical or physical location.

7. CONCLUSION

This paper has shown the need for risk management tools in software project since the complexity of risk management increases with the complexity of the developed system. The need for risk management tools which are intelligent has also been demonstrated. Such tools would have the capacity to be used with any development methodology, whether traditional, agile, or even a combination of them. The authors also proposed two frameworks for the development of intelligent risk management tools; neural networks and intelligent agent based.
8. REFERENCES


