The Incompleteness of the Risk Assessment Methods

Konstantina Georgieva
Otto-von-Guericke-University Magdeburg, Department of Computer Science, Institute for Distributed Systems, Software Engineering Group, P. O. Box 4120, 39106 Magdeburg, Germany
ina@ivs.cs.uni-magdeburg.de

Abstract

Risk assessment methods are one of the most important elements in the process of risk management. These methods consider numerous aspects while assessing and estimating the risks. Since software development is a human intensive activity, diverse factors related to human behavior also play a key role in this situation. Software risk assessment methods should take into account all these factors in combination to each other. This paper analyses the current risk assessment methods for their consideration of human factors. Only by taking the software development process within its real complexity, which means not only software and hardware resources but also the humans as a corn bean could be achieved a satisfactory level of the risk assessment process.

1 Introduction and motivation

Risk is the possibility of loss or injury in some situations. In the today's software development high project failure rate is a major concern and it is becoming more so as software projects grow in complexity. In order to manage this we need to apply risk management. The process of risk management embodies the identification, analysis, planning, tracking, controlling, and communication of risk. This gives a structured mechanism to provide visibility into risks and to achieve project success. The pioneer in this area is Barry Boehm, in 1986 he developed the first method for risk management (Boehm, 1986), and later on (Boehm, 1991) he divided the process of risk management into two main phases: risk assessment and risk control. These two steps have their sub-steps: assessment is divided into – identification, analysis and prioritization and risk control is divided into – planning, reduction and monitoring.

Observing the principles of risk management given by the International Organization for Standardization, described in (ISO, 2009) it is clear to see the following statement: '*Risk management should take into account human factors. The organization's risk management should recognize the capabilities, perceptions and intentions of external and internal people that may facilitate or hinder attainment of the organization's objectives*'. This statement gives a strong motivation to our thesis that the human factors are in the center of the risk management process and that they should be a part of the risk assessment methods. Other evidences emphasizing the role of human factors in software engineering and software development process include the People Capability Maturity Model and the pair programming development technique.

Based on the Boehm's classification of risk management we will focus on methods for risk analysis and the lack of consideration of the human factors in them. The methods for risk assessment are very important in the process of risk management because they give the possibility to predict the success of a particular project. Realizing their crucial role in the process of risk management we have to realize also that the main actors in every process are the humans, and their actions give rise to different issues or problem situations. We can have a look over a simple example: in the medicine, the safety of different machines is maintained by people. So it is clear to see how important the people are in this case. Any mistake can lead to a death of a patient. It is the same in the software development process, any risk brought by a human can be crucial for the whole system.

The paper attempts to perform an exhaustive survey of the methods for risk assessment since 1995 until today. They have been summarized shortly in order to see their mechanism of work. These methods are very different by their nature: they explore different structures in the software development process, use different techniques, and are applied over different phases in the development process. So, we are able to see the great variety of techniques for risk assessment. The methods will

particularly be investigated for their consideration of human factors while assessing and estimating risks. Our goal is to put a stress on the importance of the humans in the development process as the people stay at the basic level and they should not be underestimated. It is not possible to achieve a complete risk assessment, or risk management over a system if we do not include in it also the human factors.

There exist different types of human factors studies: human error analysis, human factors engineering and human reliability analysis (Baybutt, 1996). The errors that people commit can be seen in different perspectives, for example in the process of work of: people with other people, people with equipment/with procedures, tasks and others. Because of this a basic classification of the human errors (Baybutt, 1996) can look in the following way: slips, mistakes, violations, socio-technical and coming from the management.

The objectives of this paper are to provide researchers with a survey over the current methods for risk assessment and to provide an overview over different human risk factors and to stress their importance in the process of risk management.

The rest of the paper is organized as follows: section 2 presents a short overview over the human risk factors, section 3 presents a survey over the methods for risk assessment, section 4 is a discussion about the observed problem and section 5 concludes the paper.

2 Human risk factors in software development

The human being or the main actor in every process is by itself a very complex system for analysis. In the process of software development there exist a lot of methods for risk assessment, but no one of them is taking into consideration the human like a main actor in this kind of play. Because of this the human factors in the software development will be discussed in order to see how important they are when we want to achieve a good risk assessment process.
Humans, involved in developing, managing and using the software may commit different kinds of errors (already mentioned) because of security perceptions, skills, knowledge, personal perception, motivation, or other personal psychological reasons.
The complexity of the human behaviors does not allow us to make an easy classification of the problems that may occur. Into consideration should be taken, the personal skills, education and experience, the motivation of the worker, the personal health, psychological and social condition, and the working atmosphere, etc. There exist studies also for preventing the harms coming from workers motivated to harm the company, the so called Fraud (Wikipedia, n.d.) management techniques.
Because the people usually do not work alone, it is very important to observe the picture in the process of cooperation between different people, people and equipment, people and tasks/workplace, etc. This could be seen in the perspective of human into an organization.
The roles of humans involved in software development fall into four categories, which are: individual, team, management and stakeholder (Higuera & Haimes, 1996). The risk factors involved in the individual and team work are the following: (Islam & Dong, 2008)

- Personal competency of employing the development methods, language and tools.
- Experience and leadership of the team leader
- Team performance
- Availability of skilled personnel
- Commitment to the project
- Personnel loyalty to the organization
- Skills of identifying and analyzing the factors in risk management, etc.

Having in mind the picture of the risk factors connected with the humans we can understand the great importance that they have and realize the criticality of the human role. This motivates our idea that the risk assessment should be oriented not only to examine different processes in the software development but also to take into consideration the human factors involved in it.

3 Methods for risk assessment

The following methods for risk assessment are grouped according to the base technique that they use in the process of assessing the risk. Every method is described shortly and is analyzed for all types of factors that it considers with a special focus on existence of human factors among them.

3.1.1 Neural networks

Artificial Neural Networks (ANN or just Neural Networks) are modeled after the biological neurons in brain structures. The individual neuron models may be combined into various networks made up of many individual nodes, each with its own set of variables. These networks have an input layer, an output layer, and one or more hidden layers. The hidden layers provide

connectivity between the inputs and outputs. The network may also have feedback, which will take result variables and use them as input to prior processing nodes. With the help of NN it is possible to be modeled different possible directions in the process of software development and in this way to find the potencies for risk.

- *Using Influence Diagrams for Software Risk Analysis (Chee et al, 1995)*

Input: software metrics data collected at various stages of software development
Technology: influence diagrams, kinds of NN, used for probabilistic and decision analysis models
How it works: The method uses the conditional independence implied in the influence diagrams in order to determine the information needed for solving of a problem. Influence diagrams are used to provide quantitative advice for software risk management, improving upon traditional ad-hoc software management techniques.

- *An Enhanced Neural Network Technique for Software Risk Analysis (Neumann, 2002)*

Input: software metric data
Technology: principal component analysis and artificial neural networks (PCA-ANN). Uses pattern recognition, multivariate statistics and NN.
How it works: This is a technique for risk categorization in which principal component analysis is used for normalizing and orthogonalizing the input data. A neural network is used for risk determination/classification. The special feature in the approach, the so called cross-normalization is used to discriminate data sets, containing disproportionately large numbers of high-risk software modules.

- *A Neural Networks approach for Software Risk analysis (Yong et al, 2006)*

Input: software risk factors, obtained through interviews/questionnaires
Technology: combination of principal component analysis, genetic algorithms and neural networks
How it works: Based on the SEI and interviews with professionals in the field, is created taxonomy and factors for software risk. This data after processing is used as an input for the NN analysis. The method is divided in the following steps: 1) predict the risks with standard NN; 2) predict with the combination of NN and PCA; 3) predict with the combination of GA and NN; and 4) combine the three steps and make an overall prediction.

- *Analyzing Software System Quality Risk Using Bayesian Belief Network (Young et al, 2007)*

Input: project risk factors selected through a Delphi method based on historical project data
Technology: Bayesian Belief Network, Delphi method
How it works: The method is based on BBN and predicts and analyzes the changing risks of software development based on facts such as project characteristics and two-side (contractors and clients) cooperation capability at the beginning of the project. BBN are used for the analysis of uncertain consequences or risks and Delphi method is used for the network structure needed for the BBN.
The method is used to evaluate the software development risks in organizations.

In the system for risk assessment, proposed in the method is to be seen, that the following input points are taken into consideration: lack of management support; lack of users' participation; inexperienced project manager; inexperienced technology leader. This means that here we have the human factor taken into consideration, but not into its full extend because the human risk factors are too complex. It is not possible considering only the inexperience to cover all problems that the human being can 'contribute' with.

3.1.2 Qualitative methods – questionnaires

Qualitative methods are methods that take into consideration different qualities. They collect information with the help of different questionnaires. So they analyze not numerical but qualitative data and after this based on it they give the possibility for risk analysis.

- *SRE from the SEI Risk Management Paradigm (Williams et al, 1999)*

Input: software risk information, obtained through interviews/questionnaires
Technology: questionnaires
How it works: The SRE addresses the identification, analysis, planning, and communication elements of the SEI Risk Paradigm. The method implies the following:

- trains teams to conduct systematic risk identification, analysis, and mitigation planning

- focuses upon risks that can affect the delivery and quality of software and system products
- provides project manager and personnel with multiple perspectives on identified risks
- creates foundation for continuous and team (customer/supplier) risk management

3.1.3 Based on different software metrics

''A **software metric** is a measure of some property of a piece of software or its specifications.'' (Wikipedia, n.d.) There exist different types of software metrics, which will not be discussed here. It is important only to understand that the metrics give quantitative information about different characteristics of the software, which could be used for risk analysis.

- *Software Risk Assessment and Estimation Model (Gupta & Sadiq, 2008)*

Input: Measurement error, Model error, Assumption error in function point estimation
Technology: risk exposure and Mission Critical Requirements Stability Risk Metrics
How it works: The risk is estimated using risk exposure and software metrics of risk management, which are used when there are changes in requirements. Initially the model estimates the sources of uncertainty using Measurement error, Model error and Assumption error.

- *A Risk Assessment Model for Software Prototyping Projects (Nogueira et al, 2000)*

Input: requirement, personnel and complexity metrics
Technology: different software metrics
How it works: The method introduces metrics and a model that can be integrated with prototyping development processes. It claims to address to some extent the issue of human dependency in risk assessment but it is not clear how exactly, because there are no mentioned metrics for that.

This method claims to take into consideration some of the human factors, but anyway it is not clear what exactly has been achieved because no concrete information is mentioned.

- *Source-Based Software Risk Assessment (Deursen & Kuipers, 2003)*

Input: source code information
Technology: code metrics, questionnaires
How it works: The method focuses on "primary and secondary facts''. Primary facts are obtained through automatically analyzing the source code of a system with code metrics, and secondary facts are obtained from people through different questionnaires, who are working with or on the system. The both type of facts are of different type information, so there is needed a bridging between them and after this the obtained information is used to advise a minimizing of the potential risk.

3.1.4 Based on early risk estimation

Analyzes in the early stages of the software development is one of the desired perspectives in the process of risk estimation and mitigation. It is much cheaper if we can encounter and overcome the problems in the early stages as if we do this at a late stage of the software development process. At the early stage we have only the functionality of the desired software product, or only its structure and we discover the risks based on that.

- *A Methodology for Architecture-Level Reliability Risk Analysis (Yacoub & Ammar, 2002)*

Input: severity of complexity and coupling metrics derived from software architecture
Technology: dynamic metrics, architecture elements
How it works: That is a heuristic risk assessment methodology for reliability risk assessment, based on dynamic complexity and dynamic coupling metrics that are used to define complexity factors for the architecture elements. Severity analysis is executed with Failure Mode and Effect Analysis, applied over the architectural models. A combination between severity and complexity factors is used in order to identify the heuristic risk factors for the architecture components and connectors.

- *Software Risk in Early Design Method (Vucovich et al, 2007)*

Input: software functionality, Historical Function-Failures, Historical Failure Severities
Technology: Function-Failure Design Method
How it works: This method identifies and analyzes the risk presented by potential software failures. With the Software Function-Failure Design Method it is demonstrated the corresponding Risk in Early Design

method to the software domain, to provide a software risk assessment based on functionality, which is often the only available information in the early stages of design. RED allows the early assessment of risk, which can guide more-detailed risk assessment, provide a test-case development guide, and help in deciding on whether a software product has been tested enough.

Observing the described methods for risk assessment we can make the following statement: all the methods take as their input different type of data, that could be generalised like: architecture, design and code metrics data as visualized in Fig.1. Only few of these methods - (Young et al, 2007), (Nogueira et al, 2000) consider some sort of human factors. Although that this attempt does not seem to be comprehensive, it is a good example which gives as much importance to human factors as to the others in assessing software risks.

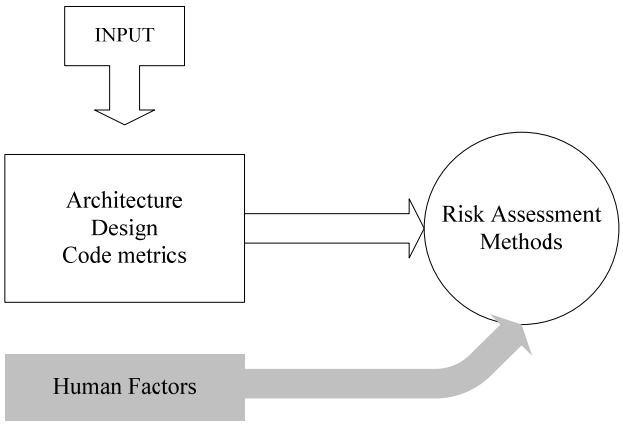

Figure 1.Input for the Risk Assessment Methods

4 Discussion

The main point of our discussion will be to underline the need of taking into consideration the whole software management system in the process of risk assessment and after this of risk management. As we have seen until the moment one software environment where some product is produced cannot be divided into separate parts in order to analyze each one of them.

If we take for example the hardware resources and analyze the risks that they can give, and after this we concern the software resources and the human factors, we will not be able to visualize the complete picture of complexity and connectivity between these components. In this way we will not be able to assess the risk in its completeness. The assessments will be incomplete and with questionable accuracy.

As can be seen in Fig.2 the risk sources in the software production process are: people, development process, software and hardware resources. These four elements give us the complete software development system, which should be analyzed in its full complexity in order to achieve an adequate risk assessment process.

Taking into consideration the information obtained from the analyzed risk assessment methods, which is that they do not consider the people like a major source of risk and analyzing the software system in its complete view Fig.2 and knowing how crucial can be the role of the human being in every activity, we can conclude that there exist an incompleteness of the existing methods for risk assessment and new methods should be developed which cover the already mentioned human factors. Only applying these new methods over the whole software development system will be possible to assess the risk in its real complexity and variability.

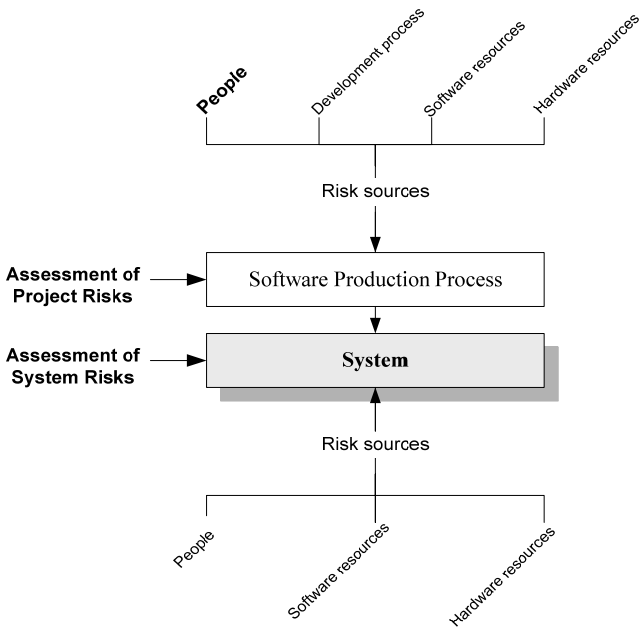

Figure 2.Risk in the different stages of the development process

5 Conclusion

In the paper we have conducted a survey over the existing methods for risk assessment from 1995 until the moment. There has been done a summary of every method and an analysis over the inputs that every method takes. It is clearly seen that all these methods do not take into consideration the human risk factors; they focus only on the software. Only two of them are trying to include some human factors, but they do not cover the real complexity of these types of risks.

The logical consequence of this research is that there is an unarguable need of development of risk assessment methods that focus on the whole software production process, take into consideration the all system without breaking it into diverse parts, and cover not only the software, hardware resources and the development process but also the people - the point that results with the most crucial and unexpected risks.

6 References

Baybutt, P., 1996. Human factors in process safety and risk management: needs for models, tools and techniques. In *International Workshop on Human Factors in Offshore Operations, US Mineral Mangement Service*. New Orleans.

Boehm, B., 1986. A spiral model of software development and enhancement. *SIGSOFT Softw. Eng. Notes*, 11, pp.14--24.

Boehm, B., 1991. Software Risk Management: Principles and Practices. *IEEE Softw.*, 8, pp.32--41.

Chee, C.L., Vij, V. & Ramamoorthy, C., 1995. Using Influence Diagrams for Software Risk Analysis. In *TAI '95: Proceedings of the Seventh International Conference on Tools with Artificial Intelligence*. IEEE Computer Society.

Deursen, T. & Kuipers, A.v., 2003. Source-Based Software Risk Assessment. In *ICSM '03: Proceedings of the International Conference on Software Maintenance*. IEEE Computer Society.

Gupta, D. & Sadiq, M., 2008. Software Risk Assessment and Estimation Model. In *International Conference on Computer Science and Information Technology*.

Higuera, R.P. & Haimes, Y.Y., 1996. *Software risk management*. CMU/SEI-TR-012

Islam, S. & Dong, W., 2008. Human factors in software security risk management. In *LMSA '08: Proceedings of the first international workshop on Leadership and management in software architecture*. Leipzig ACM.

ISO, 2009. *ISO/FDIS 31000 Risk management -- Principles and guidelines*.

Neumann, D.E., 2002. An Enhanced Neural Network Technique for Software Risk Analysis. *IEEE Trans. Softw. Eng.*, 28, pp.904--912.

Nogueira, J., Luqi & Bhattacharya, S., 2000. A Risk Assessment Model for Software Prototyping Projects. In *RSP '00: Proceedings of the 11th IEEE International Workshop on Rapid System Prototyping (RSP 2000)*. IEEE Computer Society.

Vucovich, J.P., Stone, R.B., Liu, X. & Tumer, I.Y., 2007. Risk Assessment in Early Software Design Based on the Software Function-Failure Design Method. In *COMPSAC '07: Proceedings of the 31st Annual International Computer Software and Applications Conference*. IEEE Computer Society.

Wikipedia, n.d. *Wikipedia*. [Online]. Available at: HYPERLINK "http://en.wikipedia.org/wiki/Software_metric" http://en.wikipedia.org/wiki/Software_metric [accessed 27 april 2009]

Wikipedia, n.d. *Wikipedia*. [Online]. Available at: HYPERLINK "http://en.wikipedia.org/wiki/Fraud" http://en.wikipedia.org/wiki/Fraud [accessed 27 April 2009]

Williams, R.C., Pandelios, G.J. & Behrens, S.G., 1999. *Software Risk Evaluation Method Description (version 2)*. CMU/SEI-99-TR-029, ESC-TR-99-029. Software Engineering Institute

Yacoub, S.M. & Ammar, H.H., 2002. A Methodology for Architecture-Level Reliability Risk Analysis. *IEEE Trans. Softw. Eng.*, 28, pp.529--547.

Yong, H. et al., 2006. A Neural Networks Approach for Software Risk Analysis. In *ICDMW '06: Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops*. IEEE Computer Society.

Young, H. et al., 2007. Analyzing Software System Quality Risk Using Bayesian Belief Network. In *GRC '07: Proceedings of the 2007 IEEE International Conference on Granular Computing*. IEEE Computer Society.