*Software Risk Management Techniques: A review*

*Dr. Mohamed Hagal*

Faculty of Information Technology

University of Benghazi

Benghazi, Libya

Email: mohamed.hagal@uob.edu.ly

*Mr. Salah Ablelsattar Algnashi*

Faculty of Information Technology

University of Benghazi

Benghazi, Libya

Email: [salah.algnashi@uob.edu.ly](mailto:salah.algnashi@uob.edu.ly)

*Prof. Omar M. Sallabi*

Faculty of Information Technology

University of Benghazi

Benghazi, Libya

Email: omar.sallabi@uob.edu.ly

*Dr. Salwa Elakeili*

Faculty of Information Technology

University of Benghazi

Benghazi, Libya

Email: salwa.elakeili@uob.edu.ly

**الملخص:**

يواجه مطورو البرمجيات تحديات كبيرة لتحقيق أهداف البرنامج التي تم تصميمها من أجله. قد يكون تحقيق هذه الأهداف صعبًا في ظل وجود مخاطر يصعب إدارتها ، وبالتالي ، يحاول العديد من الباحثين دراسة العديد من تلك المخاطر وتطوير العديد من الأدوات والتقنيات للمساعدة في التخفيف منها. حيث تتناول كل من هذه الأدوات جوانب محددة من هذه المخاطر ، وبالتالي ، فإن دراسة ومقارنة عدد من هذه الأدوات كانت أحد أهداف هذه الورقة. تتناول هذه الورقة دراسة ومقارنة أكثر ثماني أدوات مشهورة ، والتي أشار إليها العديد من الباحثين. حيث ان هذا البحث سيساعد المطورين على اختيار الأداة المناسبة لبيئة العمل المناسبة للتخفيف من المخاطر التي تؤثر على نجاح برامجهم.

***Abstract:***

*software developers facing great challenges to achieve the software goals that was designed. Achieving those goals may be difficult in the presence of risks that are difficult to manage, and therefore, many researchers are trying to study many of those risks and develop many tools and techniques to help mitigate them. Each of these tools addresses specific aspects of these risks, and therefore, the study and comparison of a number of these tools were one of the objectives of this paper. This paper deals with the study and comparison of the eight most famous of these tools, which are referred to by many researchers. This will help developers choose the right tool to mitigate the risks that affect their software success.*

***Keywords:*** ***Risk management, Project management, Risk management methods, PMI standard.***

# ***Introduction***:

In the software development field, risks have been presented over time as well as many risk management techniques. A number of researchers and institutions carried out many investigations, their contribution was intended to be helping that software development can be done without the involvement of risks (Shahzad & Al-Mudimigh, 2010). In recent years, regardless of many risk management models were proposed by several researchers, a lot of software projects still have a high degree of failures. Incorrect risk assessment during software development was the major reason behind these unsuccessful projects. The risk analysis process was done on overall projects rather than in each stage of the Software Development Life Cycle (SDLC) (Roy & Dasgupta, 2015). The key challenges for most researchers were identifying, mitigating or avoiding these risks as early as possible in the projects (Shahzad & Al-Mudimigh, 2010) (Roy & Dasgupta, 2015).

Although there are several methods in software risk management, software development projects still have a high

rate of risks that could lead to system failures. Most of the failures could be avoided pro-actively by dealing with risk factors rather than waiting for problems to occur and then trying to react. It is very clear that software development process is considered to be a very risky process, however, there should be some techniques, models and frameworks that could be used to identify, mitigate, or avoid these risks in early stages in SDLC (Elzamly & Hussin, 2016) (Hijazi, Alqrainy, Muaidi, & Khdour, 2014).

Most problems in the software industry are faced just because of the poor software risk handling mechanisms or due to the absence of any such mechanism at all (Shahzad & Al-Mudimigh, 2010). Due to that, a study and comparison of the existing methods are highly required to help software development teams in selecting the suitable method.

There are many tools, techniques and models for risk management which makes the process of selecting the appropriate tool depends on the knowledge of the developer or his experience. This may affect the choice of the appropriate tool. So, the evaluation and recommendations of these tools and techniques are crucial in order to facilitate the process of monitoring risks (Roy & Dasgupta, 2015).

The aim of this study is to compare some risk management methods based on Project Management Institute (PMI) standard for project risk management, to help software development teams in selecting the appropriate method for their project. These methods are used in the requirement stage of the SDLC.

The reminder of this paper is organized as follows. Section II gives some related work about software risk management methods and techniques. Section III explains risk management methods. Section IV illustrates the comparison between different risk management methods based on PMI standard. Section V presents discussion and results. Section VI draws conclusions.

# ***Related Work***

Several studies have been conducted on risk management methods and techniques. Some researchers reported that there is no method that is better than another because each method has its own nature and environment. Some other researchers concluded that risk management in software development is not yet totally controlled, and therefore it still needs more research (Ghule, 2014).

Hijazi et al. (Hijazi, Alqrainy, Muaidi, & Khdour, 2014) made a comprehensive theoretical study of the main risk factors occurring in each stage of SDLC. They concluded that their research can be useful for guiding project teams in identifying potential risk factors and help them in designing strategies to mitigate or avoid these risks.

Elzamly & Hussin (Elzamly & Hussin, 2016) presented a review of the quantitative and intelligent risk models in software risk management for software development projects. Most of the articles they reviewed were only focused on traditional techniques and models used to identify software risks. However, they believed that software project managers must use and combine between techniques during all the software project lifecycle regarding software risk methodology practice.

Rabbi & Mannan (Rabbi & Mannan, 2016) conducted a study on the risk management system, and to find tools and techniques recommended by most journals and articles. They have gone through several approaches in the context of risk management. They used risk management paradigm introduced by the Software Engineering Institute as a standard to evaluate several techniques. They noticed several features as well as a shortcoming. After comparative analysis, it was clear that no single tool or technique stand-alone is perfect for dealing with risks in software development because of their different features and working environments. However, their comparative analysis helped to provide basic ideas to the organizations in the selection of risk management techniques depending on their developments.

# ***Risk Management Methods***

In this section, we will focus on some well-known software risk management methods, which is recommended at the SDLC stage of requirement (Roy & Dasgupta, 2015). The methods are:

## **BOEHM**

Boehm's model aids in identifying the primary sources of risk, and eventually analyses and resolves them. The main purpose of this model is to focus on important success elements and to offer the strategies that allow the project to deal with them. The risk assessment and risk control strategies presented in the model, offer the foundation for the capability needed to implement a risk-oriented approach (Boehm, 1991).

Fig. 1 illustrates BOEHM’s model for software risk management.

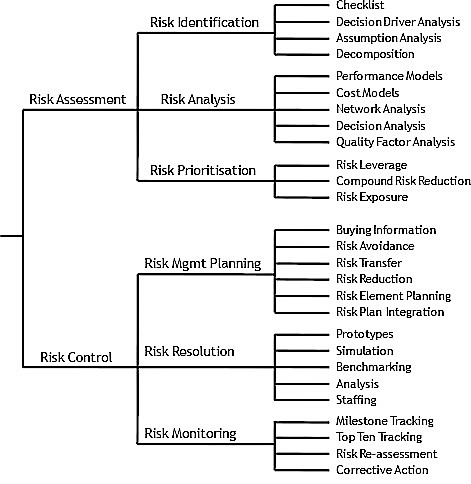


Figure 1.BOEHM’s model for software risk management (Boehm, 1991).

## **SEI-SRE**

SRE method is a process for identifying, analyzing, and developing mitigation strategies for risks in a software-intensive system while it is under development. The SRE method has been developing on the SEI since 1992. The SRE addresses the identification, analysis, planning, and communication elements of the SEI risk Paradigm. The analysis element is also included complete with the aid of SRE activities. Planning factors are partly addressed through the construction of high-level mitigation strategy plans. The SRE additionally contributes significantly to the communication element. The remaining elements of the paradigm, tracking and control, aren't addressed throughout the SRE. (Williams, Pandelios, & Behrens, 1999).

Fig. 2 shows SEI-SRE method.

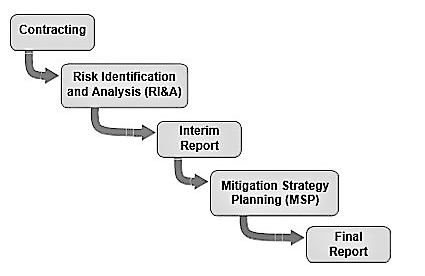


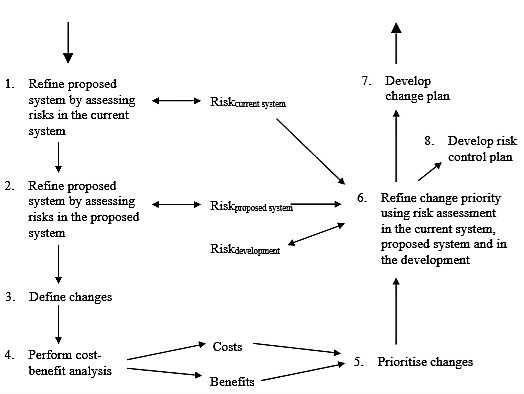
Figure 2 SEI-SRE Method (Williams, Pandelios, & Behrens, 1999)

## **Riskit**

The Riskit provides a systematic way that helps in identifying, analyzing and controlling risks that cause potential threats to project goals. The various Riskit steps may be enacted in parallel and multiple steps can be completed in a single session. Further, it is occasionally necessary to complete the risk management cycle quickly before other risks have even been analyzed – so that effective control actions can be implemented immediately. (Kontio, 2001).

Fig. 3 shows the steps of the Riskit method.

Figure 3 SERUM Method (Hijazi, Alqrainy, Muaidi, & Khdour, 2014)



## **SERUM**

This method is used for handling changes in the software development process, and the risks associated with these changes. It combines the use of implicit risk management with explicit risk management techniques. It uses two specific techniques: Checkland and Wilson’s Soft Systems Methodology (SSM); and Gilb’s Evolutionary Delivery. The method is based on a combined cost-benefit and risk analysis of the current system, the proposed system and the change process (Greer & Bustard, 1997).

The steps of the serum method are presented in Fig. 4.

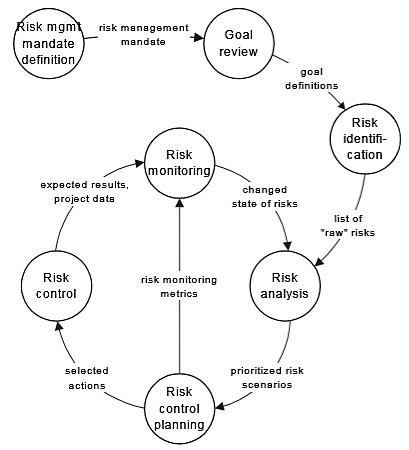


Figure 4 Riskit Method (Kontio, 2001).

## **SRAM**

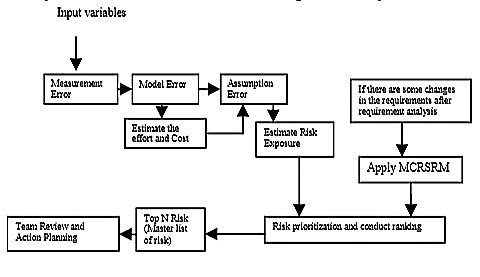
The SRAM method includes the following nine critical risk elements: the complexity of software, staff involved in the project, targeted reliability, product requirements, method of estimation, method of monitoring, development process adopted, the usability of software and tools used for development. Then, a set of questions is carefully chosen for each one of these elements with three answers to each question. Answers are ordered in the risk order. The model also helps in calculating levels of risk associated with Quality, Schedule and Cost of a project separately. It is noticed that the model was able to assess the level of project risk and predict the possible outcome with reasonable accuracy. It is worth mentioning that the SRAM only focuses on the development risk and does not assess the marketing risk (Foo & Muruganantham, 2000).

## **SRAEM**

This model makes clear estimates by measuring the sources of uncertainty using Measurement error, Model error and Assumption error. This model assesses not only the risk but also estimates the risk. The author in (Gupta & Sadiq, 2008) used the function point in the measurement process, as it is an important software metrics, which is used to calculate the approximate Line of Code (LOC), Cost and effort of software. The model is based on the risk assessment as well as software project estimation. Risk assessment includes risk identification, risk analysis, and risk prioritization according to the traditional risk management model. This model is different from the traditional model as the traditional model does not use the estimation sources of uncertainty. Therefore, it is easy to calculate the risks at different phases as the software projects progress from phase to phase.

An overview of SRAEM method is presented in Fig. 5.

Figure 5 SRAEM Method (Gupta & Sadiq, 2008).



## **SRAEP**

SRAEP uses the model-based approach and the Software Fault Tree (SFT) concepts as a major step in order to identify the risk. Unlike SREAM model, the SREAP model takes into account the software complexity, which is important in determining the risk for the software projects. It also takes into account the issues related to requirements. The SRAEP model estimates the risk at each stage of software while it keeps preceding from stage to another (Sadiql, Ahmad, Rahmani, & SherJung, 2010).

An overview of the SRAEP model is presented in Fig. 6.

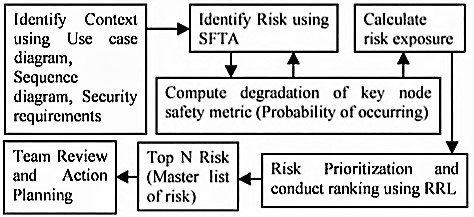


Figure 6 SRAEP Method (Sadiql, Ahmad, Rahmani, & SherJung, 2010).

## **SPRMQ**

SPRMQ model tries to overcome the weaknesses of the current frameworks of software risk management. SPRMQ attempts to improve software product risk management by applying four sequential processes relying upon quality attributes and during the operational life cycle of the product. SPRMQ framework is developed essentially relying on the five characteristics of software product quality: functionality, reliability, performance, efficiency, and maintainability. With SPRMQ, project managers or risk managers can control a software product by measuring the impact of risks that can occur at three levels: high, medium, or low using SPRMQ processes and strategies that depend on quality attributes (MOFLEH & ZAHARY, 2011).

The four processes of SPRMQ is shown in Fig. 7.

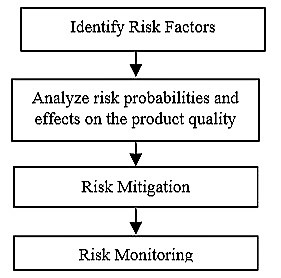


Figure 7 SPRMQ Method (MOFLEH & ZAHARY, 2011).

# ***RISK Management Comparison Based on PMI Standard***

In this section, we will provide a brief description about PMI standard to understand its core principles and guidelines, as well as making the comparison between the methods presented in the previous section based on the PMI standard.

## **PMI Standard**

PMI standard provides a benchmark for the project management career that defines elements of project risk management that may most of the time be defined as good practice in most projects. It also presents a standard that is globally applicable and consistently applied. The standard covers risk management as it is implemented to single projects only ( PRACTICE STANDARD FOR PROJECT RISK MANAGEMENT , 2009).

The PMI standard for project risk management outlines the principles of effective risk management as follows ( PRACTICE STANDARD FOR PROJECT RISK MANAGEMENT , 2009):

* Plan Risk Management.
* Identify Risks.
* Perform Qualitative Risk Analysis.
* Perform Quantitative Risk Analysis.
* Plan risk Responses.
* Monitor and Control Risks.

This practice standard can be used by project management practitioners to validate the risk management process being employed in a specific situation, project or organization. This practice standard encourages project managers to make use of those principles as they are fundamental to effective, comprehensive, and successful Project Risk Management. Those principles can be used as a check for software organization’s processes. Development team can establish processes specific to their particular situation, project, or organization and then compare them with these principles, thus validating them against good Project Risk Management practice ( PRACTICE STANDARD FOR PROJECT RISK MANAGEMENT , 2009). More details on the standard can be found in the reference.

## **The Comparison**

After reviewing the risk management methods and PMI standard for risk management in the previous sections, we are now prepared for the comparison.

First, we will determine the basic criteria according to the PMI standard, then, each of the previous methods will be compared with these criteria.

We will use the previously mentioned principles of the effective risk management according to the PMI standard, and compare the previously mentioned methods with it.

Table I summarized the results of the comparison.

Table I. Risk Management Comparison Based on PMI Standard

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Methods | Criteria | | | | | | Total |
| Plan Risk Management | Identify Risks | Perform Qualitative Risk Analysis | Perform Quantitative Risk Analysis | Plan risk Responses | Monitor and Control Risks |
| BOEHM |  |  |  |  |  |  | 5 |
| SEI-SRE |  |  |  |  |  |  | 5 |
| RISKIT |  |  |  |  |  |  | 5 |
| SERUM |  |  |  |  |  |  | 5 |
| SRAM |  |  |  |  |  |  | 4 |
| SRAEM |  |  |  |  |  |  | 4 |
| SRAEP |  |  |  |  |  |  | 4 |
| SPRMQ |  |  |  |  |  |  | 4 |

It is noted that the overall performance of the comparative methods of risk management was decent. It is also observed that the limitations of each method are minor and thus it can be enhanced. The results of this comparison are discussed in details in the next section.

# **Discussion**

According to the PMI standard for risk management, the main processes of any effective risk management method, model, or framework should be, planning for risk management, identifying Risks, qualitative and quantitative analysis, plan risk responses, and monitor and control the risks. All these processes to mitigate risk items before they become a real threat to project success.

The first four methods are shown in Table I (BOEHM, SEI-SRE, RISKIT, SERUM) were slightly better than the others, although, each one of them has its limitations.

BOEHM model satisfies the PMI criteria except for one principle which is plan risk management. It was not clear how the risk management processes will be performed, and how to integrate project risk management with all other project management activities.

SEI-SRE showed good results too, but it also has its shortcoming, where it lacks for the monitoring and controlling mechanism.

RISKIT lacks the plan risk responses process, this process is tended to provide the best response after qualitative and quantitative analysis, instead, it makes a control planning immediately after the analysis.

Similar to BOEHM model, SERUM is also lacking plan risk management process, as it is focusing on change management. It starts with identifying risks related to the change process, and then take the appropriate actions.

SRAM and SPRMQ lack two criteria from PMI standard, plan risk management and qualitative risk analysis.

SRAEM and SRAEP are quite similar to each other, although, there are few differences. The first difference is that SRAEM does not have information regarding the identification of risk while SRAEP has. The second difference is that SRAEP makes plan risk management at the beginning while SRAEM does not have information about it.

# **Conclusions**

This paper has studied and compared some common risk management methods, based on the PMI standard for risk management. The findings of this study indicate that there is no perfect method standalone, and thus, several methods can be used together at various stages of SDLC.

Although the methods showed a decent performance, but they have some minor limitations, and they can be enhanced.

It is noticed that risk management can be handeled better at early stages of SDLC, and thus, any further enhancement on those methods should consider this point.

We believe that our research will provide developers with basic information regarding selecting the best method for their project.

**References**

B. Shahzad and A. S. Al-Mudimigh, (2010), "Risk Identification, Mitigation and Avoidance Model for Handling Software Risk," in Second International Conference on Computational Intelligence, Communication Systems and Networks, Liverpool.

B. Roy and R. Dasgupta, (2015), "A Study on Software Risk Management Strategies and Mapping with SDLC," in 2ND INTERNATIONAL DOCTORAL SYMPOSIUM ON APPLIED COMPUTATION AND SECURITY SYSTEMS, KOLKATA.

A. Elzamly and B. Hussin, (2016), "Quantitative and Intelligent Risk Models in Risk Management for Constructing Software Development Projects: A Review," International Journal of Software Engineering and Its Applications , vol. 10, no. 2, pp. 9-20.

H. Hijazi, S. Alqrainy, H. Muaidi and T. Khdour, (2014), "RISK FACTORS IN SOFTWARE DEVELOPMENT PHASES," European Scientific Journal , vol. 10, no. 3, pp. 213-231.

S. Ghule, (2014), "Risk Analysis and Mitigation Plan in Software Development," International Journal Of Engineering Sciences & Research Technology, vol. 3, no. 8, pp. 546-548.

M. F. Rabbi and K. O. B. Mannan, (2016), "A Short Review for Selecting the Best Tools and Techniques to Perform Software Risk Management," European Journal of Advances in Engineering and Technology, vol. 3, no. 6, pp. 1-7.

B. W. Boehm, (1991), "Software Risk Management: Principles and Practices," IEEE Software, vol. 8, no. 1, pp. 32-41.

R. C. Williams, G. J. Pandelios and S. G. Behrens, (1999), "Software Risk Evaluation (SRE) Method Description (Version 2.0)," Software Engineering Institute , Pittsburgh.

J. Kontio, (2001), "Software Engineering Risk Management: A Method, Improvement Framework, and Empirical Evaluation," in Helsinki University of Technology , Helsinki.

D. Greer and D. Bustard, (1997), "SERUM - Software Engineering Risk: Understanding and Managemen," Journal of Project and Business Risk Management, vol. 1, no. 4, pp. 373-388.

S.-W. Foo and A. Muruganantham, (2000), "Software risk assessment model," in Proceedings of the 2000 IEEE International Conference on Management of Innovation and Technology, Singapore.

D. Gupta and M. Sadiq, (2008), "Software Risk Assessment and Estimation Model," in International Conference on Computer Science and Information Technology , Singapore.

M. Sadiql, M. W. Ahmad, M. K. I. Rahmani and SherJung, (2010), "Software Risk Assessment and Evaluation Process (SRAEP) using Model Based Approach," in International Conference on Networking and Information Technology , Manila.

H. M. MOFLEH and A. ZAHARY, (2011), "A Framework For Software Product Risk Management Based On Quality Attributes And Operational Life Cycle (SPRMQ)," in The 12th International Arab Conference on Information Technology, Riyadh.

PRACTICE STANDARD FOR PROJECT RISK MANAGEMENT, (2009), Pennsylvania : Project Management Institute, Inc.