Journal of AL-Qadisiyah for computer science and mathematics Vol.11 No.1 Year 2019
ISSN (Print): 2074 – 0204 ISSN (Online): 2521 – 3504

Monitoring software risks based on integrated AHP-ANN method

Jaber Ibrahim Naser
Msc(IS), Directorate of Education in Al-Qadisiya.
jaberalidami@gmail.com

Hussein Ali Ghadhban Alsalman
Msc(IS), Directorate of Educatio in Basrah.
Hussein.alsalman85@gmail.com

Recived : 2\10\2018 **Revised //** **Accepted : 14\10\2018**

Available online : 25/1/2019

DOI: 10.29304/jqcm.2019.11.1.463

Abstract

Software risk management refers to systematic process for analyzing and identifying the project risks. The present paper provides a hybrid method for IT software risks identification. Software projects possess different features which increase the project failure possibilities. Therefore, the present work integrate the Artificial Neural network with the Analytic Hierarchy Process (AHP-ANN) in order to solve the problem of software project estimation in early stage. The questionnaire developed to find out the risk functional model and provide the proposed method with proper data. The results observe a major common risk in software projects is the insufficient knowledge based on different software project life cycle stages. Also, there are some other important factors in software projects such as lack of good estimation in project scheduling, poor definition of project requirements which cause human errors.

Keywords: ANN, AHP, risk identification .

1. Introduction

Risk management can be defined as a method to identify the software project threats in order to enhance the software firms organization. The risks sources can be the erroneous strategic in project management or the external challenges. Therefore, there is a need for operation enhancement of software project in order to develop the software efficiency and flexibility [1]. Most of studies investigated the risk factors and provided some useful techniques to specify the effectiveness of them. The ranking based on risks importance is made in light of analysis, planning, maintenance, design and implementation [2]. Classifying risk factors can be considered risk attributes as the main issue in developing risk project. Development of risk management software can be classified into scheduling risks and quality risks. Also, it can be grouped into performance risks, cost risks support risks and schedule risks [1][3]. These classifications were very helpful in monitoring and controlling risks in software projects. More importantly, the top ten software risk factors in developing software were chosen and utilized for analysis [4].

Some authors apply Artificial Neural Networks to identify the risks and to develop an application for risks management during software development [5]. Many other techniques have been used in this field such as regression analysis, expert systems, stochastic models, Monte Carlo Simulation, Decision Tree and Analytic Hierarchy Process AHP [6].

finally, theresearchers develop some techniques to tag the same goal such as Singular Value Decomposition SVD technique [7]. In this work, Artificial Neural Networks have been integrated with Analytic Hierarchy Process (AHP) method for risk control as a tool for risk management.

2. Software Risk identification

Software risk identification is considered the activity of the potential risks which can effect on the project development and determination. The risk check list can be creadt based on the identified risk concepts [5]. It occurs when the organization faces uncertainties from limited capacity and costs in its pursuit for opportunities. In this regard, an effective risk management initiative coupled with suitable risk management strategies can help mitigate the cost and stress brought on by risk issues [8]. Risk identification is a critical process, the risk management mostly depends on identifying all possible risks that may face the project during development [1]. The result of software risk identification is the risk factor list. The identification of risk factors will be followed by risk analysis. The quantitative risk analysis simulates each critical risk effect. Elzamly in 2014 brought forward new methods using quantitative and mining methods to conduct comparisons among risk management methods in the lifecycle of software development [4].

Artificial Neural Networks (ANNs) were developed by Gandhi et al. in 2014 for the prediction of the level of risks in software projects, where risks were detected prior to the project implementation and the steps taken to mitigate them ensures higher rate for successful projects [5]. Hojjati and Noudehi in 2015 applied Monte Carlo simulation for risks assessment. The study evaluated project risks in the IT domain and utilized the Primavera Risk Analysis software to quantitatively analyze management [9]. Paraschivescu in 2016 brought forward integrated quality and risk management concepts resulting in an integrated management system risk that sheds light on new dimensions and perspectives. Also, Elzamly et al. in 2016 identify software risks and software development controls [10]. The study ranked the risk factors in software based on their importance and how often they occurred in a data source. The ANNs applicability was examined in Andreas's study in an attempt to analyze survey data concerning risk management practices effectiveness in the context of product development (PD) projects and forecasting of project outcomes [7]. They explained the relationships between risk management factors that influence successful PD project (e.g., cost). Salman in 2018 apply the maintenance risk factors in Singular Value Decomposition (SVD) correlated with the traditional risk factor calculations to estimate the software maintenance projects[7]. Based on the present review, the researchers specifies the main risk factors that can be used in this work as in the next section.

3. Selection of Risk Factors

Risk management is a process to develop strategies for identifying and estimating their impact. The steps taken for risk management process in the present work are as follows;

1. Risk identification, it represents the activity of detecting the effected potential risk in the project that affected the project development. In the present work, the researcher developed a questionnaire using the taxonomy based risk identification presented by Marvin J. Carr [11].
2. Risk analysis, it represents the process of understanding of where, when and why the risk appear. This process take place based on direct queries about the impact and probability of the risk elements. Traditional risk analysis focuses on the potential impacts to a human population due to the presence of an introduced substance or event, for example the presence of pesticides in a body of water used for human consumption, or an oil spill. A broad variety of techniques are used to evaluate risk in these situations. Risk analysis typically involves four steps: hazard identification, risk assessment, determining the significance of the risks, and risk communication. The traditional risk management in the present paper focuses on pure risk and refers to individual risks as if they don't interact (Simona-Iulia, 2014). Based on the present two bases in the software project risk management, the researchers listed the risk factors.

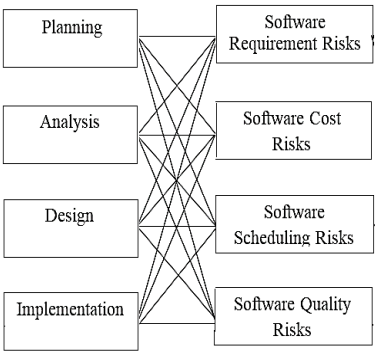
The 'Top 10 software risk factors' lists differ to some extent from author to author, but some essential software risk factors that appear almost on any list can be distinguished. These factors need to be addressed and thereafter need to be controlled. Consequently, the list consists of the 10 most serious risks of a software project ranked from one to ten, each risk's status, and the plan for addressing each risk [10] [4]. However, the software risk factors listed in Table 1 below are considered in this study. In addition, these factors are the most common factors used by researchers and experts when studying the software risk factors in software development lifecycle.

Hoodat and Rashid classify the software risks and specify the relations between these risks. They used the risk tree structure correlated with the probabilistic calculation. The analysis helps qualitative and quantitative assessment of risk of failure. Also, its help software risk management process [2]. Therefore, this classification used in this research as a base of study which is correlated with the Software Development Life Cycle. Figure 1 shows the project scheme.

Table 1. Illustrate Top Software Risk Factors in Software Project Lifecycle

Software Requirement Risks	A1	Poor definition of requirements
	A2	Inadequate of requirements
	A3	Invalid requirements
	A4	Lack of good estimation
	A5	Lack of accurate system domain definition
Software Cost Risks	B1	Unrealistic finance schedule
	B2	Lack of good cost estimation
	B3	Lack of monitoring
	B4	Complexity of architecture
	B5	Human errors
Software Scheduling Risks	C1	Inadequate knowledge about Techniques
	C2	Lack of accurate system domain definition
	C3	Lack of employment of manager experience
	C4	Lack of good estimation in project implementation
	C5	Lack of skill
Software Quality Risks	D1	Lack of skill
	D2	Lack of good estimation in projects
	D3	Human errors
	D4	Lack of employment of manager experience
	D5	Lack of project standard

Figure 1. The project scheme



This scheme will be used in ANN based on AHP technique. The risk parameters selected form Hoodat (2009) based on the top ten risks presented by Elzamly (2016) and Salman (2018) [2],[7],[10].

4. Risk Factors Evaluation

The study developed a questionnaire that comprised of questions relating to chosen 34 risks maintenance risk factors adopted from Lopez and Salmeron (2012). The questions were chosen with the hope of the works of Marvin (1993) and Webster (2006) and the risk factor values calculated two types of questions, positive and negative [11],[13],[6]. The former type represents the questions that had yes answers, while the latter type represented those that had no answers. The sum of the questionnaire list of questions for every type of risk can be represented by the following formula[5],[7]

$$RF = \sum_{i=1}^{N} (Q_i W_i) \qquad (1)$$

Where RF = Risk Factor Value, Q =value of each question, W = weight

Thus, the boundary condition is represented as:

$$Q = \begin{cases} positive\ questions & \begin{matrix} if\ yes = 1 \\ if\ no\ \ = 0 \end{matrix} \\ negative\ questions & \begin{matrix} if\ yes = 0 \\ if\ no\ \ = 1 \end{matrix} \end{cases} \qquad (2)$$

Collection of data was conducted using questionnaire to determine the commonly occurring risks in majority of software projects in the software companies. The respondents were then presented with the 20 software risk factors. The study sample comprising of 150 persons worked in specific IT organizations in Iraq. These peoples represent the Software Life Cycle user areas. The collected data reflect the selected software risk factors which developed to be used in ANN.

5. Methodology of Risk Factor Specification

In order to specify the risk factors in software project life cycle, the researcher integrate the AHP technique with ANN as in the following:

a) Artificial Neural Networks (ANNs)

The neural network can be defined as a parallel distributed processor. The main processing unit is inspired by the way of biological nervous system, such as the process information of human brain. The potential system of ANN involves several layers developed by computing elements and called nodes. The system operation of neural network depends on the signal transmission.

When the nods receives the input signal from input representation of the system, it will transfer the signal to the next step node. This process will mining the transferred the data in order to find out the specific correlation in input data. The first layer represents the input layer and the last layer considered the output layer. The input layer is received the data of the case study which represent the statistical data. The last layer produce the solution of the problem which represent the predicted or identified data. In between, there are hidden layers which operate the complex data to identify proper pattern using system of specific formulas. The reason for using the neural network ire as in follows:

1. It must have the ability to learn the neural system how to do tasks. The tasks done based on the given tanning data.

2. It must have the ability to generalize the internal system operation. It must produces reasonable outputs without paying attention how deal with the internal processes.

b) AHP

Analytical Hierarchy Process (AHP) methodology has been applied to the evaluation of risk related to software project. Five risks are evaluated and defined in each project stage as presented in table 1. The criteria weights can be more precisely defined by the AHP methodology using "Saaty scale" than using the digital logic method. However, subjectivity is playing a great role in both of methods. Subjectivity is included to the comparison of alternatives by the original AHP methodology, also. Contrary, by using other method there is no subjectivity concerned of alternatives comparisons because of dealing with transformed values of criteria. The ranking of all alternatives can be performed, by obtaining the priorities. The weights present the relative importance of each criterion compared to the goal. Finally, alternatives present the group of feasible solutions of the decision problem.

6. Experimental results

The methodology of the present paper is to integrate the ANN with the AHP technique. The AHP will present a pattern to the ANN. Based on the results, the software project risks were important in the perspective of the project managers, whereas all controls are used most of the time, and often. The risks were ranked on importance in light of analysis, planning, design and implementation. In particular, top of software risk factors in software development Lifecycle were very important, aggregating the responses resulted in the following ranking of the importance of the listed risks. The AHP model in this study is formed to prioritize the various risks within the software project. The result observe the factor priority, for instance the software requirement results can be seen in table 2 and 3.

Table 2. Analytical Heirarchy Process Matrix

	A1	A2	A3	A4	A5
A1	1	0.6	0.375	0.5	0.3
A2	1.666667	1	0.625	0.833333	0.5
A3	2.666667	1.6	1	1.333333	0.8
A4	2	1.2	0.75	1	0.6
A5	3.333333	2	1.25	1.666667	1
COL. TOTAL	10.66667	6.4	4	5.333333	3.2

Table 3. Normalized Score Table

A1	0.09	0.09	0.09	0.09	0.09	0.47	9.38
A2	0.16	0.16	0.16	0.16	0.16	0.78	15.63
A3	0.25	0.25	0.25	0.25	0.25	1.25	25.00
A4	0.19	0.19	0.19	0.19	0.19	0.94	18.75
A5	0.31	0.31	0.31	0.31	0.31	1.56	31.25
COL. TOTAL	1	1	1	1	1	5	

In the present method, the ANN trained based on the conjugate gradient backpropagation algorithm.It represents a proper choice for problem of classifications. It is used less memory requirements and provide faster response than gradient decent algorithms.

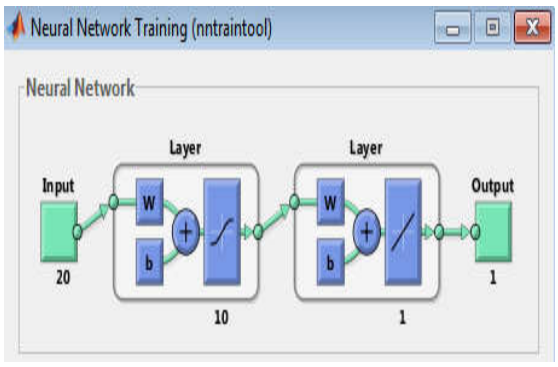
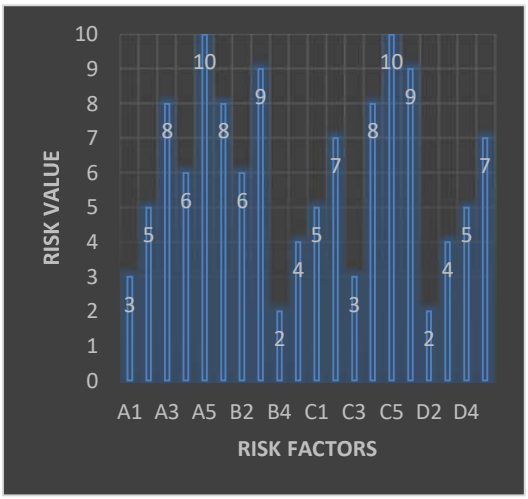


Figure 2. ANN with one hidden layer and ten hidden nodes

The plotted results shown in Figure 3 indicate that results observe the same responses of the risk factor effects.

The yellow bars represent the ANN results, while the blue bars represent the integrated method. The bar graph shows that the risk identification due to the present method. These valid results highlight the largest problem on IT software risk factors which represents inadequate knowledge/skills, insufficient expertise and Insufficient/inappropriate staffing. The results observe a big effect by the insufficient expertise in the applied software management as shown in figure



3.

Figure 3: Identifying the Software Improvements Needs

This factor are critical in development the Risk Management. It can provide important information regarding that risk improvement and risk management practices. The higher risk in software project phases based on the four phases project life cycle came from identifying software improvements needs (phase 1) which observe 41%, while the other three risk groups observe 21% and 22% and 16% as shown in Figure 4.

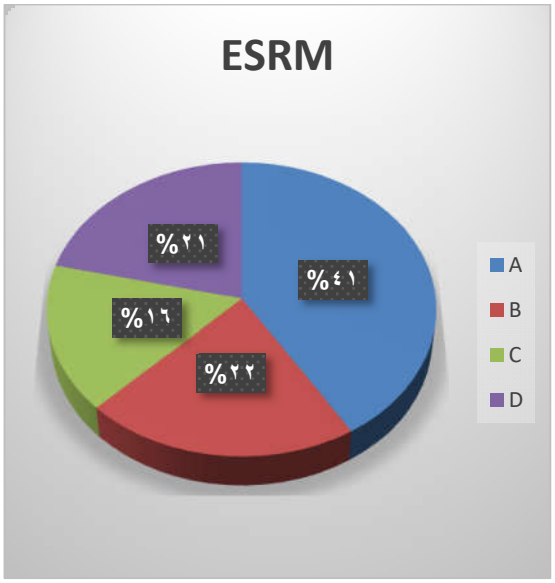


Figure 4: Identifying Risks in risk Phases

8. Conclusion

In this work we apply AHP technique with ANN to support Risk Management. The results show that that AHP technique is simple and efficient for total variance in common questionnaire of each of the software risk factors to model if they are effective in mitigating the occurrence of each risk factor. The result of AHP is presented as a pattern to ANN. As a conclusion, this method can be used effectively to identify the risk effect in all project phases. The used method specify the Risks in three reasons. It specifies the root of risk problem and the effective phase of project.

9. References

[1] S. K. A. Al-balqa and T. J. K. Al-balqa, "Risk Factors in Software Development Phases," no. January, 2014.

[2] H. Hoodat and H. Rashidi, "Classification and Analysis of Risks in Software Engineering," pp. 446–452, 2009.

[3] M. Boban, Ž. Požgaj, and H. Sertić, "STRATEGIES FOR SUCCESSFUL SOFTWARE," pp. 77–92, 2003.

[4] A. Elzamly, "Evaluation of Quantitative and Mining Techniques for Reducing Software Maintenance Risks," vol. 8, no. 111, pp. 5533–5542, 2014.

[5] A. Gandhi, A. Naik, K. Thakkar, and M. Gahirwal, "Risk Management in Software Development using Artificial Neural Networks," *Int. J. Comput. Appl.*, vol. 93, no. 19, pp. 1–7, 2014.

[6] S. M. Neves and C. E. S. da Silva, "Risk management applied to software development projects in incubated technology-based companies: literature review, classification, and analysis," *Gest. Prod., São Carlos*, vol. 23, no. 4, pp. 798–814, 2016.

[7] H. A. Salman, "G lobal J ournal of E ngineering S cience and R esearch M anagement," vol. 5, no. 5, 2018.

[8] E. Angelini and A. Roli, "A neural network approach for credit risk evaluation," vol. 48, pp. 733–755, 2008.

[9] S. N. Hojjati, "The use of Monte Carlo simulation in quantitative risk assessment of IT projects," vol. 2621, pp. 2616–2621, 2015.

[10] A. Elzamly, B. Hussin, and N. Salleh, "Top Fifty Software Risk Factors and the Best Thirty Risk Management Techniques in Software Development Lifecycle for Successful Software Projects," vol. 9, no. 6, pp. 11–32, 2016.

[11] M. J. Carr and S. L. Konda, "Taxonomy-Based Risk Identification," no. June, 1993.

[12] "COMPARATIVE STUDY BETWEEN TRADITIONAL AND ENTERPRISE RISK," pp. 276–282.

[13] P. B. Webster, M. De Oliveira, N. Anquetil, B. Av, and W. Norte, "A Risk Taxonomy Proposal for Software Maintenance."

مراقبة مخاطر البرمجيات بالاعتماد على طريقة دمج (الشبكة العصبية الاصطناعية ــ عملية التحليل الوراثي)

جابر ابراهيم ناصر
مديرية التربية في القادسية
jaberalidami@gmail.com

حسين علي غضبان
مديرية التربية في البصرة
Hussein.alsalman85@gmail.com

المستخلص :

ادارة مخاطر البرمجيات تشير الى المعالجة المنظمة لتحليل و تحديد مخاطر المشروع . هذا البحث يوفر طريقة هجينة لتحديد مخاطر برامجيات تكنولوجيا المعلومات . مشاريع البرامجيات تمتلك خصائص مختلفة تزيد من احتماليات فشل المشروع. لذلك فأن العمل الحالي يدمج (الشبكة العصبية الذكية) مع (عملية التحليل الوراثي) من اجل حل مشكلة تخمين مشروع البرامجيات في مرحلة متقدمة . طورت الاستبيانات لايجاد النموذج الوظيفي للخطر و كذلك توفير طريقة مقترحة مع بيانات مناسبة. النتائج رصدت الخطر الشائع و الرئيسي في مشاريع البرمجيات و هو المعرفة غير الكافية بالاعتماد على مراحل دورة حياة البرامجيات المختلفة. و كذلك، هنالك بعض العوامل المهمة الاخرى في مشاريع البرامجيات مثل الافتقار الى التخمين الجيد في جدولة المشروع ، ضعف تعريف متطلبات المشروع والتي تسبب اخطاء بشرية .

الكلمات المفتاحية : الشبكة العصبية الاصطناعية ، عملية التحليل الوراثي ، تحديد الخطر .