Fuzzy Logic Driven Expert System for the Assessment of Software Projects Risk

Mohammad Ahmad Ibraigheeth[1], Syed Abdullah Fadzli[2]
Faculty of Informatics and Computing, Universiti Sultan ZainalAbidin, 21300 Kuala Terengganu, Malaysia

***Abstract*—This paper presents an expert risk evaluation system developed and based on up-to-date empirical study that uses a real data from huge number of software projects to identify the most factors that affect the project success. Software project can be affected by a range of risk factors through all phases of the development process. Therefore, it has become necessary to consider risk concerns while developing the software project. Risk assessment and management play a significant role in avoiding failure of the software project, and can help in mitigating the effect of the undesirable events that could affect the project outcomes. In this paper, the researchers have developed a novel expert fuzzy-logic tool that can be used by project decision makers to evaluate the expected risks .The developed tool helps in estimating the risk probability based on the software project's critical success factors. A user-friendly interface is created to enable the project managers to perform general risk evaluation during any stage of the software development process. The proposed tool can be helpful in achieving effective risk control, and therefore improving the overall project outcomes.**

***Keywords*—Risk assessment; critical success factors; fuzzy expert systems; fuzzy rule-base; risk probability**

I. Introduction

Risk is a probable event that might lead to undesirable impact on software project outcomes. Software risk is an unexpected problem occurs during software operations that might cause software failure [1]. Project risk assessment and management can help in mitigating the effect of the undesirable events. Identification of probable risk factors is one of the major issues in software project management. Today, the software systems are widely used by people to control and manage their daily routines, due to this fact; it has been a must to consider risk concerns when developing any software project.

Developing tools to assess and manage software risks have become increasingly important for measuring the health of the software project during all phases of the software development process. All organizations should focus on managing risks related to their software projects. When risk factors are reported, risk mitigation strategies should be developed in order to avoid potential project failure. Although considering software risk concerns has become critical, there is a limited number of developed tools that can be used by the project decision makers in evaluating and mitigating the probable risks.

This paper aims to develop a new expert fuzzy tool that can help project managers to evaluate the expected project risk. This tool evaluates the project risk probability based on ten critical success factors. Using fuzzy set theory is advantageous for recording linguistic variables that are usually used by project managers to describe parameters in the project development environment.

A fuzzy based user-friendly tool to evaluate "risk probability" of the software project is developed to support general software project risk assessment through any phase of the software development process. The percentages of presence of ten success factors identified in CHAOS report are used as input to the model. A linguistic variable used for each input, and two membership functions are defined: NO and YES. Fuzzification process then is used to map the crisp values specified by the model users to the fuzzy space Mamdani interference system with rules base includes 1024 if-then rules used to evaluate the project risk as a fuzzy number. Finally, Defuzzification module converts this number into crisp value that represents risk probability of the software project.

The developed model can be used as a tool to guide the software project decision makers in making critical decisions in early stages throughout the software development process, and in identifying alternative strategies to avoid the software probable risks.

This research presents two contributions: First, it develops an expert risk evaluation system based on up-to-date survey conducted by Standish organization that uses a real data from 50,000 projects to identify the most factors that affect the project success. Second, it provides general and easy-to-use tool with user-friendly interface that enables project managers to assess the project risk during any phase of software development process.

The rest of this paper is organized as follow: Section 2 describes software project success factors. Section 3 reviews the related literature. Section 4 explains the proposed model. Section 5 describes the risk evaluation tool design. Section 6 provides experimental work and analyses the behavior of the system. Section 7 concludes the research, describes its limitations, and suggests future work.

II. Software Project Success Factors

Many software project success and failure factors have been described in the literature [2-5]. In this paper, we investigate the effect of project success factors identified in CHAOS report. The report identifies ten software project success factors ranked according to their influence on the project success as shown in Table 1 [6].

TABLE I. SOFTWARE PROJECT SUCCESS FACTORS

Factors of Success	Impact
Executive Sponsorship	15%
Emotional Maturity	15%
User Involvement	15%
Optimization	15%
Skilled Resources	10%
Standard Architecture	8%
Agile Process	7%
Modest Execution	6%
Project Management Expertise	5%
Clear Business Objectives	4%

The CHAOS success factors presented in Table 1 can be defined as a following [6]:

- Executive sponsorship: when the executives provide a suitable financial and emotional supports, they will increase the opportunity to implement a successful software project.
- Emotional maturity: this relates to project environment and how the project team work together. Having the skills to manage relationships, self-managed and socially aware, can help in producing more successful projects.
- User involvement: when users are not involved, the project will perform poorly. User participation in project decision making, and through requirements understanding phase has a major positive effect on project success.
- Optimization: optimization of some project aspects can maximize the project efficiency. This includes optimization the scope based on the project sponsorship capabilities, and identifying the optimal team size.
- Skilled resources: the project success is made up by staff who have the necessary skills to understand and perform the project requirements.
- Standard architecture management environment (SAME): SAME is defined by the Standish Group as a collection of consistent behaviors including the integration of services, practices, and products in software development process.
- Agile process: it describes a set of values including adaptive planning, flexible response to change, early delivery, and continuous improvements. These principles support producing successful projects.
- Modest execution: it takes place when the process has few and simple moving parts, and when the tools used in project development process have few features used sparingly.
- Project management expertise: is the use of knowledge, skills, procedures and techniques in the project development activities to achieve the desired project goals, and meet the organization requirements.

III. RELATED WORK

Numerous techniques have been used to address and manage the software risks. A software risk management framework is proposed by Boehm [7]. He defined list of top software risks depending on his experience. There were some limitations in his study. No theoretical foundations were presented in his work. Also, as he identified the risks in 1991, these risks have become inadequate as the software development environment has increasingly become more complex and diverse.

Another survey was conducted by Barki et al. [8]. A list of 23 software risks is identified and classified into five sets. The complexity of assessment scale that was used for each risk posed a limitation.

Schmidt et al. [9] also conducted a survey by integration of many experts opinion to identify 53 software risks. These risks were grouped into 14 sets. As the experts were from different countries, the study declared that the list could be affected and have become inapplicable.

Wallace et al. [10] defined 27 software risks and classified them into 6 dimensions (i.e., user, requirements, complexity, planning, staff, and development environment) by performing cluster analysis to develop model that measure the software project risk.. Performing cluster analysis is helpful in finding variable similarities to perform accurate prediction.

Artificial intelligent approaches also used widely to counter and manage the software risks. A regression analysis method is used in research proposed by Jiang and Klein [11] to define the most risk factors that affect the process of project development. The impact of applying a certain management activities on the software project outcomes is considered [12]. A genetic algorithm combined with decision trees is an approach for risk prediction by using certain software metrics developed by Xu z et al. [13]. A fuzzy logic is used in developing system to evaluate the software risks through earlier phase of software development cycle [14]. Yavari et al. [15] proposed a method based on Wallace's [10] work to assess software risk using fuzzy logic. Neural networks are used to identify software projects with high risk [16]. Hu Y et al. [17] proposed a framework for risk analysis based on risk causality using Bayesian networks. Each of these techniques has its own advantages. For example, regression analysis is suitable for risk prediction as it can find the relationships between variables. Applying decision trees is fast and simple while neural network is suitable when the relationships between the system variables are non-linear. Applying Bayesian network with considering causality dependencies can perform better prediction.

The main advantage of our approach is developing a novel tool for software risk assessment based on critical success factors. The primary objective of our work is to perform general risk evaluation that can be done through any stage of software development life cycle (SDLC). The proposed tool can be helpful in achieving effective risk control, and therefore improving the overall project outcomes.

IV. Proposed Software Risk Assessment Model

In this paper, ten success factors that are identified in CHAOS report [6] are used (refer to Table 1). Fig. 1 shows our model. The final output of this model is the software risk probability due to the mentioned ten factors.

Fuzzy Logic toolbox in MATLAB is used to implement Mamdani inference system. The following steps (shown in Fig. 2) explain how the model works:

Step 1: Fuzzification

In this step, crisp values (within the range of 0 to 100) for the ten input variables are measured. A scale mapping then performed for these inputs to obtain their membership values within the range of 0 to 1.Two trapezoidal membership functions (similar to Fig. 3). We might interpret NO as: input percentage of presence below 50%, and YES as: input percentage of presence higher than 50%.

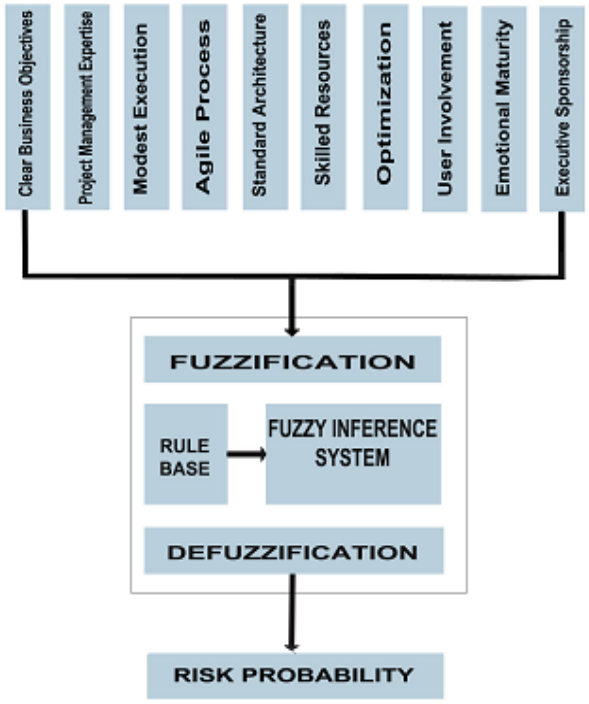

Fig. 1. Risk Evaluation Model.

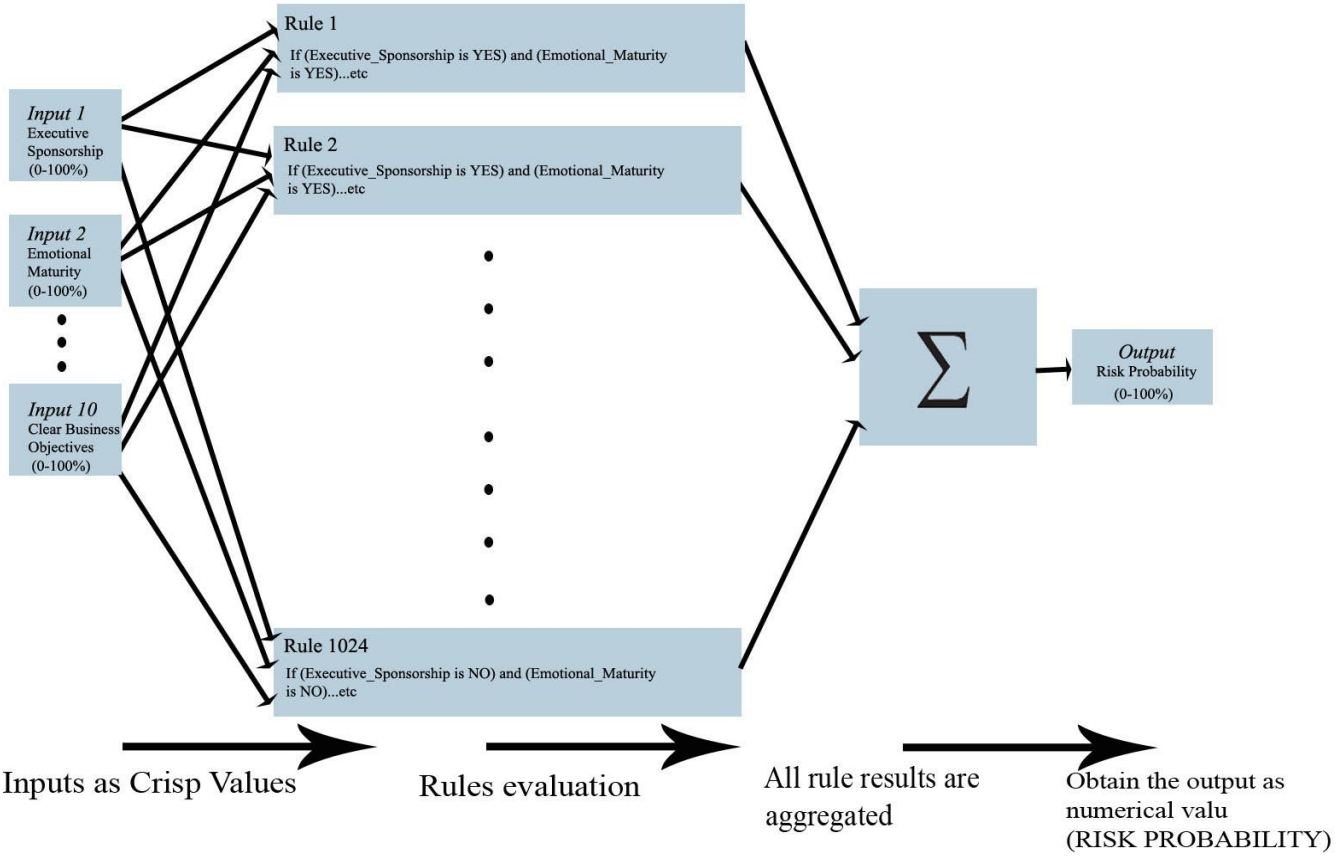

Fig. 2. Risk Evaluation Steps.

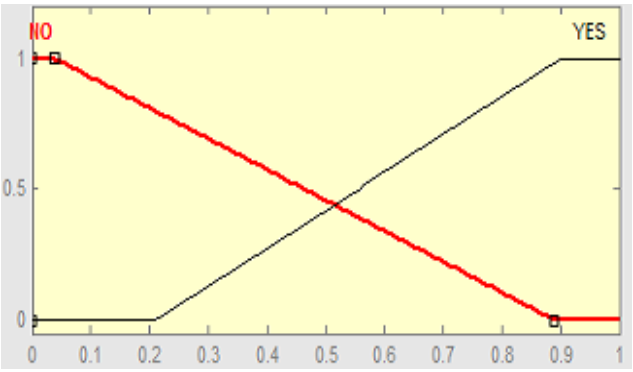

Fig. 3. Trapezoidal Membership Functions (Trapmf).

Step 2: Rules Evaluation

The rule base includes 1024 IF-THEN rules. The following are samples of the created rules:

- Rule 1:If (Executive_Sponsorship is YES) and (Emotional_Maturity is YES) and (User_Involvement is YES) and (Optimization is YES) and (Skilled_Resources is YES) and (Standard_Architecture is NO) and (Agile_Process is YES) and (Modest_Execution is YES) and (Project_Management_Expertise is YES) and (Clear_Business_Objectives is YES) then (RiskProbability is NONRISKY)
- Rule 10: If (Executive_Sponsorship is YES) and (Emotional_Maturity is YES) and (Us)
- er_Involvement is YES) and (Optimization is YES) and (Skilled_Resources is YES) and (Standard_Architecture is NO) and (Agile_Process is NO) and (Modest_Execution is YES) and (Project_Management_Expertise is YES) and (Clear_Business_Objectives is NO) then (RiskProbability is NONRISKY)
- Rule 127: If (Executive_Sponsorship is YES) and (Emotional_Maturity is YES) and (User_Involvement is YES) and (Optimization is NO) and (Skilled_Resources is NO) and (Standard_Architecture is YES) and (Agile_Process is NO) Optimization is YES) and (Skilled_Resources is YES) and (Standard_Architecture is NO) and (Agile_Process is NO) and (Modest_Execution is NO) and (Project_Management_Expertise is YES) and (Clear_Business_Objectives is YES) then (RiskProbability is RISKY)
- Rule 397:If (Executive_Sponsorship is YES) and (Emotional_Maturity is NO) and (User_Involvement is NO) and (Optimization is YES) and (Skilled_Resources is YES) and (Standard_Architecture is NO) and (Agile_Process is NO) and (Modest_Execution is NO) and (Project_Management_Expertise is YES) and (Clear_Business_Objectives is YES) then (RiskProbability is RISKY)
- Rule 1024:If (Executive_Sponsorship is NO) and (Emotional_Maturity is NO) and (User_Involvement is NO) and (Optimization is NO) and (Skilled_Resources is NO) and (Standard_Architecture is YES) and (Agile_Process is NO) and (Modest_Execution is NO) and (Project_Management_Expertise is NO) and (Clear_Business_Objectives is NO) then (RiskProbability is RISKY)

The fuzzified inputs that are obtained in step 1 are applied to the antecedent parts of rules in the rule base. As the fuzzy rule has multiple antecedents, we apply AND operator, with product (prod) method to produce single value that represents the evaluation of each rule antecedent parts .A fuzzy implication operator (minimum method) then is applied to clip the membership values of the rule consequent parts based on membership values of antecedents. The model output is categorized in two linguistic variables that are Risky and Non-Risky. Also, two linguistic variables are used for each input, namely: NO and YES.

Step 3: Outputs aggregation

In this step, the previously truncated membership functions of rule consequents are combined to obtain single fuzzy set.

Step 4: Defuzzification

Defuzzification is used to calculate the output as numerical value. Centroid method is applied to obtain the value that represents the software project risk probability.

Fig. 4 shows the model's fuzzy inference system (FIS) represented by using MATLAB FIS editor. It includes ten input variables, and one output named RiskProbability.

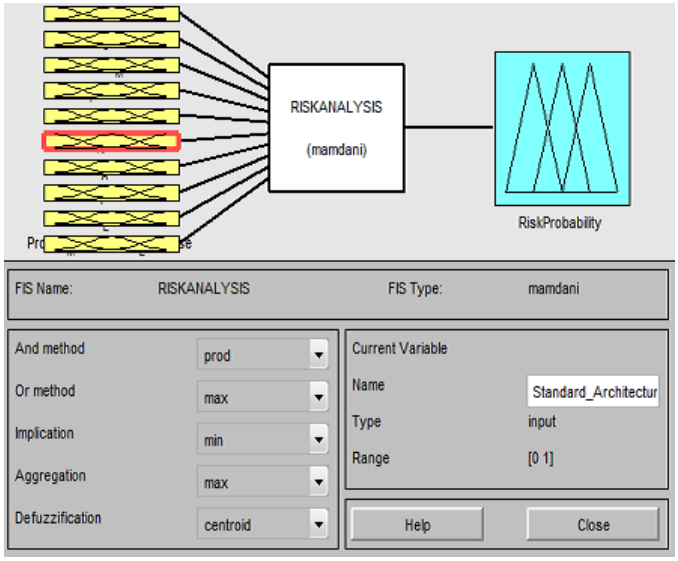

Fig. 4. FIS Input and Output Variables.

V. Tool Design

The graphical user interface (GUI) shown in Fig. 5 is developed to enable software project decision makers to easily access our risk assessment tool. The user first specifies percentages of presence of the ten items (inputs) in his project, and then he presses "Estimate Risk Score" button to evaluate the project risk probability.

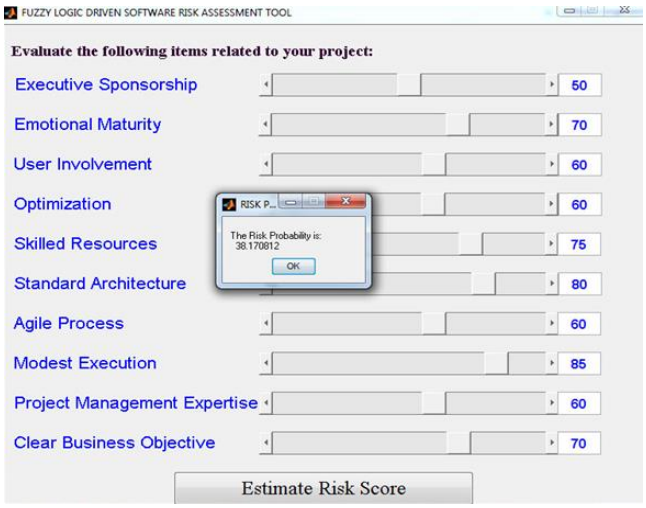

Fig. 5. Software Risk Assessment.

It is strongly recommended that the tool to be used earlier or during any phase of the project development process to determine the current state of the software project and to identify the possible improvements to mitigate risk and avoid the project failure. The goal of this tool is assigning one of the following labels to the project under evaluation:

- Low risk (LOW): If the risk probability is less than 40%, this indicates that the project is healthy and expected to be successfully completed. However, there are no fully guaranteed successful projects, therefore project managers should be aware of individual items with low scores, and these items should be tracked and controlled during all stages of the project development process.
- Medium risk (MED): If the risk probability in range of 40 to 60%, the project should be identified as a medium risk, and efforts must be made to avoid occurrence of the undesirable events. Improvement should be applied to those individual items with low scores that can mitigate the overall project probability of risk.
- High Risk (HIGH): If the risk probability is greater than 60%, this indicates that the project has run into serious risks that can cause failure if process improvement methods are not applied. The stakeholder should be reported that there is imminent danger of project failure. All project phases have to be kept under monitoring, and a quality reports should be regularly carried out. If the risk is still high after applying mitigation methods, it could be better to decide not to proceeding with this project implementation.

VI. Experimental Work and Discussion

To analyze the behavior and sensitivity of our risk assessment tool, we assumed that it is applied on eight virtual projects. Descriptions of these projects and results of their assessments by the risk tool are presented in Table 2.

TABLE II. Tool Assessment Results for Eight Software Projects

Success Factor	*Project ID*							
	Project A	**Project B**	**Project C**	**Project D**	**Project E**	**Project F**	**Project G**	**Project H**
Executive Sponsorship	80%	47%	88%	55%	40%	80%	80%	40%
Emotional Maturity	75%	38%	90%	50%	35%	90%	40%	40%
User Involvement	75%	60%	85%	53%	30%	80%	33%	30%
Optimization	70%	50%	86%	40%	40%	77%	30%	30%
Skilled Resources	85%	70%	70%	70%	40%	60%	71%	60%
Standard Architecture	80%	66%	70%	55%	48%	63%	45%	63%
Agile Process	60%	55%	50%	50%	44%	44%	40%	33%
Modest Execution	85%	70%	50%	80%	70%	50%	66%	70%
Project Management Expertise	87%	55%	60%	72%	70%	60%	46%	60%
Clear Business Objectives	80%	50%	60%	75%	72%	40%	51%	61%
Risk Probability	20.63%	48.62	20.05	50	60.92	29.3	56.8	62.44
Risk Classification	**LOW**	**MEDIUM**	**LOW**	**MEDIUM**	**HIGH**	**LOW**	**MEDIUM**	**HIGH**

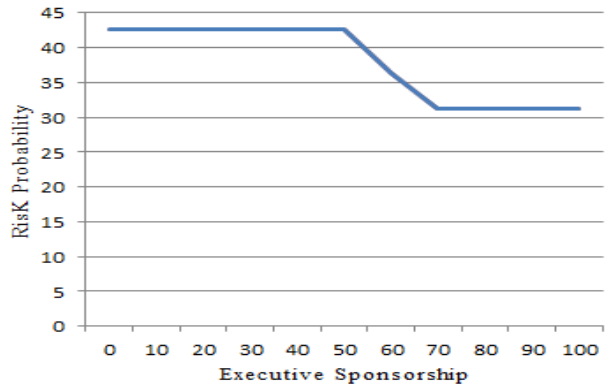

Fig. 6. Effect of "Executive Sponsorship" on the Project Risk Probability.

In all projects, we observed that some factors have higher impacts on the project risk than others. For example in project E, even it has some factors with high score (i.e. factors with percentage of presence higher than 60%), namely, Modest Execution Project Management Expertise, and Optimization. Similarly, the risk probability for project F is "LOW" even it has six factors with low scores (i.e. factors with percentage of presence lower than 60%). This is due to the high scores of the first four factors that have higher effect on the project success.

Also, a sensitivity analyses can be performed for the individual factors. Fig. 6 presents sensitivity analysis for "Executive Sponsorship" factor to show its effect on the total project risk probability. The percentage of presence of the considered factor is changed within the range of 0 to 100% while all other input parameters are kept fixed.

VII. Conclusion

In this paper, a fuzzy based user-friendly tool to assess "risk probability" for the software projects is presented. This tool is developed based on software project success factors identified by Standish organization. These factors correspond to real data collected through survey involved about 50,000 projects.

The developed tool supports a general assessment of project risk at any phase of development process. The percentages of presence of ten success factors are used as input to the system that produces a numerical value which presents the total project risk probability. The result can be used to assign one of three labels namely: low, medium, or high risk to the software project. The developed tool can be used to guide the decision makers in making critical decisions early to avoid undesired events that might cause project failure. The system behavior and sensitivity are analyzed using eight virtual projects and the impacts of various factors are observed.

The presented tool has two limitations. First, the proposed approach did not consider correlations between factors. For example, the percentage of presence of "User Involvement" factor may be correlated with the percentage of presence of "Clear Business Objectives" factor. Second, we have not applied the tool to actual software projects. Even though our model is implemented based on actual empirical data to be a supportive tool that can be used for observing the current state of the project "during development process" (i.e. this tool is not designed to be applied on already released projects), it might be useful to involve software companies to verify the results of the proposed tool.

Future research work will investigate how the system prediction accuracy can be improved by using learning algorithms based on historical data from previous projects.

References

[1] Xu Z, Khoshgoftaar TM, Allen EB, Application of fuzzy expert systems in assessing operational risk of software, Info.Soft. tech., 45 (7) (2003) 373-388.

[2] Ewusi-Mensah, K. (2003). Software Development Failures: Anatomy of Abandoned Projects. *Cambridge: MIT Press.*

[3] GAO Report *(14-705T).* (2014). Preliminary Results of Undercover Testing of Enrollment Controls for Health Care Coverage and Consumer Subsidies Provided Under the Act.

[4] The Standish Group. (2013). The Chaos Manifesto.*The Standish Group.*

[5] Ibraigheeth, M., & Fadzli, S. A. (2019). Core Factors for Software Projects Success. JOIV: International Journal on Informatics Visualization, 3(1).

[6] Hastie S, Wojewoda S. , Standish group 2015 chaos report-q&a with Jennifer Lynch, (2016).

[7] Boehm BW, Software risk management: principles and practices, IEEE software 8(1)(1991), 32-41.

[8] Barki H, Rivard S, Talbot J.,Toward an assessment of software development risk, J. manag. Info. sys, 10(2) (1993) 03-25.

[9] Schmidt R, Lyytinen K, Keil M, Cule P, Identifying software project risks: An international Delphi study". J. manag. Info. Sys,17(4) (2001)5-36.

[10] Wallace L, Keil M, Rai A, How software project risk affects project performance: An investigation of the dimensions of risk and an exploratory model, Deci. sc., 35(2)(2004) 289-321.

[11] Jiang JJ, Klein G. ,Risks to different aspects of system success, Info. &Manag., 36(5) (1999) 263-272.

[12] García MN, Román IR, Peñalvo FJ, Bonilla MT, An association rule mining method for estimating the impact of project management policies on software quality, development time and effort. Exp. Sys. App., 34(1) (2008)522-529

[13] Xu Z, Yang B, Guo P, Software risk prediction based on the hybrid algorithm of genetic algorithm and decision tree.in Int. Conf. on Intelligent Computing, Berlin, 2007(Springer, Berlin, Heidelberg)pp. 266-274.

[14] Xu Z, Khoshgoftaar TM, Allen EB, Application of fuzzy expert systems in assessing operational risk of software, Info.Soft.Tech., 45(7) (2003)373-88.

[15] Yavari A, Golbaghi M, Momeni H, DAssessment of Effective Risk in Software Projects based on Wallace's Classification Using Fuzzy Logic, Int. J. Info. Eng. Electronic Bus., 5(4) (2013) 58

[16] Neumann DE. , An enhanced neural network technique for software risk analysis, IEEE Trans. on Software Eng.,28 (9) (2002) 904-912.

[17] Hu Y, Zhang X, Ngai EW, Cai R, Liu M. ,Software project risk analysis using Bayesian networks with causality constraints, Deci.Sup. Sys., 56 (2013) 439-49.