Review of Computer Engineering Research
2015 Vol.2, No.1, pp.22-38
ISSN(e): 2410-9142
ISSN(p): 2412-4281
DOI: 10.18488/journal.76/2015.2.1/76.1.22.38

CLASSIFICATION AND IDENTIFICATION OF RISK MANAGEMENT TECHNIQUES FOR MITIGATING RISKS WITH FACTOR ANALYSIS TECHNIQUE IN SOFTWARE RISK MANAGEMENT

Abdelrafe Elzamly[1†] --- Burairah Hussin[2]

[1]*Information and Communication Technology, University Technical Malaysia Malaka (UTeM); Department of Computer Science, Faculty of Applied Sciences, Al-Aqsa University, Gaza, Palestine*

[2]*Information and Communication Technology, University Technical Malaysia Malaka (UTeM), Malaysia*

ABSTRACT

Regardless how much effort we put for the success of software projects, many software projects have a very high failure risk rate. The failure risk is not always avoidable, but it could be controllable by using risk management technique through the software development Lifecycle. The aim of this study is to present the factor analysis techniques to classify and identify the risk management techniques in the software development project. The best thirty risk management techniques were presented to respondents and all risk management techniques are used most of the time, and often. We have chosen three components: Planning and requirement techniques, communication techniques, models and tools for thirty risk management techniques. The study has been conducted on a group of software project managers in software development companies. We will intend to apply these study results on a real-world software project to verify the effectiveness of the risk management techniques on a software project for mitigating risks. Successful identifying of risk management techniques will greatly improve the probability of mitigating software risk.

Keywords: Software risk management, Software development project, Risk management techniques, Factor analysis technique.

Received: 16 August 2015/ **Revised:** 5 January 2015/ **Accepted:** 8 January 2015/ **Published:** 12 January 2015

Contribution/ Originality

This study is to present the factor analysis technique to classify and identify the risk management techniques (controls) in the software development project for mitigating risks.

† Corresponding author

1. INTRODUCTION

Despite much research and progress in the area of software project management, it still fails to deliver acceptable systems on time and within budget. Much of these failures could be avoided by managers pro-actively planning and dealing with risk management techniques rather than waiting for problems to occur. Due to the involvement of risk management in monitoring the success of a software project, analysing potential risks, and making decisions about what to do with potential risks, the risk management is considered the planned control of risk. The goal of risk management is identification and recognition of risks and risk management techniques at early stage and then actively changes the course of actions to mitigate and reduce the risk [1]. In the process of understanding the risk factors and risk management techniques that contribute to software project success, software risk management is becoming increasingly important. In this paper, we identify and classify risk management techniques that are guide software project managers to mitigate risks in software development projects. Risk management is a practice of controlling risk and practice consists of processes, methods, and tools, techniques for mitigating risks in a software project before they become problems [2]. Furthermore, successful software project risk management will greatly improve the probability of project success [3].

The objective of this study is: To identify risk management techniques of software projects in the software development organizations for mitigating software risk based on the literature review, to classify the risk management techniques for mitigating software risk in the software development organizations.

2. LITERATURE REVIEW

The new technique used the chi-square ($\chi 2$) test to control the risks in a software project [4]. However, we also used new techniques which are the regression test and effect size test proposed to manage the risks in a software project and reducing risk with software process improvement [5]. Also we improved quality of software projects of the participating companies while estimating the quality–affecting risks in IT software projects. The results show that there were 40 common risks in software projects of IT companies in Palestine. The amount of technical and non-technical difficulties was very large [6]. Furthermore, we used the new stepwise regression technique to manage the risks in a software project. These tests were performed using regression analysis to compare the controls to each of the risk factors to determine if they are effective in mitigating the occurrence of each risk factor implementation phase [7]. In addition, we proposed the new mining technique that uses the fuzzy multiple regression analysis techniques to manage the risks in a software project. However, these mining tests were performed using fuzzy multiple regression analysis techniques to compare the risk management techniques to each of the software risk factors to determine if they are effective in mitigating the occurrence of each software risk factor [8]. Further, the fuzzy regression analysis modelling techniques are used to manage the risks in a planning software development project. Top ten software risk factors in

planning phase and thirty risk management techniques were presented to respondents [9]. In addition, we identified and managed the maintenance risks in a software development project by using fuzzy multiple regression analysis [10]. Also, we proposed new mining techniques that uses the fuzzy multiple regression analysis techniques with fuzzy concepts to manage the software risks in a software project. Top ten software risk factors in analysis phase and thirty risk management techniques were presented to respondents. However, these mining tests were performed using fuzzy multiple regression analysis techniques to compare the risk management techniques with each of the software risk factors to determine if they are effective in reducing the occurrence of each software risk factor [11]. Also, the paper aimed to present new mining technique to identify the risk management techniques that are effective in reducing the occurrence of each software implementation risks [12]. Furthermore, we presented the new statistical techniques–namely, the stepwise multiple regression analysis techniques and Durbin Watson techniques to reduce software maintenance risks in a software projects [13]. The authors continue the effort to enrich the managing software project risks with consider mining and quantitative approach with large data set. The two new techniques are introduced namely stepwise multiple regression analysis and fuzzy multiple regression to manage the software risks [14]. This paper aimed to present new techniques to determine if fuzzy and stepwise regression are effective in mitigating the occurrence software risk factor in the implementation phase [15]. According to Dash and Dash [16] risk management consists of the processes, methodologies and tools that are used to deal with risk factors in the SDLC process of Software Project. In addition, the optimization method was tested with various software project risk prediction models that have been developed [17]. Finally, risk management methodology that has five phases: Risk identification (planning, identification, prioritization), risk analysis and evaluation (risk analysis, risk evaluation), risk treatment, risk controlling, risk communication and documentation these relied on three categories techniques as risk qualitative analysis, risk quantitative analysis and risk mining analysis throughout the life of a software project to meet the goals [18].

3. RISK MANAGEMENT TECHNIQUES

Through reading the existing literature on software risk management, we listed thirty risk management techniques that are considered important in mitigating the software risk factors identified. In the study, we summarize the best 30 risk management techniques in mitigating risk as follows:

C1: Using of Requirements Scrubbing

It is a best practice for software projects in which a product specification is carefully tested for unimportant or overly complex requirements, which are then deleted [19]. This is believed the reasons as the process of reviewing each requirement in detailed absolutely necessary for the

upcoming release and it can increase dramatically the chances of delivering software project on-time and within budgets [20].

C2: Stabilizing Requirements and Specifications as Early as Possible

The key to stabilizing requirements is through a partnership developed in software projects. Therefore, the functional manager plays vital role in transferring business knowledge to the software project team and participating in the process design and the requirements that support the process design [21]. Many software projects are faced with uncertainty when software requirements are first stated [22]. However, they referred to stabilize requirements and specifications as early as possible as a risk management techniques [4].

C3: Assessing Cost and Scheduling the Impact of Each Change to Requirements and Specifications

Indeed, they found that software failure risks are dramatically positive related with both overruns budgets and schedule [23], [24]. Hence, estimating cost and software project schedule impact is important to mitigate risk requirements and specifications and the successful software development [25].

C4: Develop Prototyping and have the Requirements Reviewed by the Client

Software prototype is a rapid software development for validating the requirements and help software team to understand the software [26]. In addition, it is clear that building early prototypes can help coin out some changes software development lifecycle. This is reported by Savolainen, et al. [27], as prototyping can reduce requirements creep and can be combined with other approaches. Furthermore, prototyping approach can used to mitigate risk issues as user interfaces, software/system interaction, or software performance [28], [29].

C5: Developing and Adhering a Software Project Plan

Some authors reported that developing and adhering a software project plan to deliver software project within the budget and on the schedule [22], [30]. In addition, he proposed application of software planning techniques to manage the multiple problems and the complexity associated with software planning [31].

C6: Implementing and Following a Communication Plan

Communication plan is crucial for monitoring progress [32] as each individual should feel suitable to provide inputs on raised problems. Progress knowledge should be shared with all concerned during or at the completion of each task before moving forward to the next.

C7: Developing Contingency Plans to Cope with Staffing Problems

Developing contingency actions that able to be taken if the software project turns into a risk failure [22]. Furthermore, creating risk contingency plans is risk mitigation for the group of facilities to be reduced risks.

C8: Assigning Responsibilities to Team Members and Rotate Jobs

Assigning clear responsibilities and roles for the members of the risk response team that contribute developing software project in software development lifecycle and to meet immediately with various aspects of disaster response, assessment, and recovery [22], [33]. It is important to assign the responsibilities clearly for the appropriate performing organizations in the early stage with lead [34]. It is also sometimes better to rotate developers and leaders the sections of the software project development to gain a variety of experiences [2].

C9: Have Team-Building Sessions

Clearly, when team building sessions were conducted by the software project manager throughout the entire software project lifecycle it contribute to software project success [35].

C10: Reviewing and Communicating Progress to Date and Setting Objectives for the Next Phase

The team manager need to review the progress in all phases such as number of units designed, reused, tested, and integrated module [2], [32].

C11: Dividing the Software Project into Controllable Portions

A software project manager need to break large software project into incremental small work elements to mitigate software project risks [22]. Furthermore, the methodology describes how a software project is divided into manageable stages enabling efficient control of resources and regular progress monitoring throughout the software development lifecycle.

C12: Reusable Source Code and Interface Methods

According to Jones and Sodhi and Sodhi [2], Jones [36], reusable source code and interface methods will impacted many new tools and programming languages such as Java, and object-oriented (OO) languages. Thus, reusable source code and interface method is useful to mitigate risk.

C13: Reusable Test Plans and Test Cases

A pre-release defect can be found in any of the software project [36]. Hence, reusable test plans and cases would speed up the process of creating testability of test plans and allow an easier test case generation [37].

C14: Reusable Database and Data Mining Structures

According to Jones [36], reusable database structures and data mining tools greatly improve the ability of the analyst to make data-driven discoveries, where most of the time spent in performing an analysis spent in data identification, gathering, cleaning and processing the data. This is similar to which proposed a method for generic and reusable text mining techniques in support of biological database [38].

C15: Reusable User Documents Early

According to Jones [36], referred to reusable user documents. In addition [39], proposed that explicit part of knowledge could be captured in several forms such as user manual, training documents, process design documents, and others. This will help software developers and used bind into standard communication approach [40].

C16: Implementing/Utilizing Automated Version Control Tools

According to Green [41], software developers need to have a version control systems for manage source code changes [41]. The version control tools are able to track evolving versions of a project's work products, and testing tools to aid in verifying the software. Fairley also commented that automated version control is essential for establishing and maintaining the baselines of various work products in various stages of development [41].

C17: Implement/Utilize Benchmarking and Tools of Technical Analysis

According to Jones [36], explained benchmarking, or comparing software productivity, quality, schedules, salaries, and methodologies, between companies was rare when the data for the first edition was assembled. Therefore, software benchmarking is continuing to expand in terms of the kinds of information collected and the number of companies that participate. Based on the ever-growing amount of solid data, the benchmarking is now a mainstream activity within the software world.

C18: Creating and Analyzing Process by Simulation and Modeling

Modelling and simulation of software development processes is gaining an increasing demand to reduce risks that focuses on a specific software development/maintenance/evolution process [29], [42]. In addition [43], described the process model simulation on risk occurrence probability do have an impact in software project. Furthermore, software processing simulation modelling (SPSM) has been emerging as a promising approach to address a variety of issues in software engineering area, including risk management [44].

C19: Provide Scenarios Methods and Using of the Reference Checking

According to Alhawari, et al. [45], described risk analysis phase by conducting scenarios for

major risks, and events to establish a probability of losses for every risk scenario. However Schmidt, et al. [46], suggested various methods for identifying software risk factors including scenarios. This will lead to allow more realistic plans and estimates to be prepared and identified risk [47].

C20: Involving Management during the Entire Software Project Lifecycle

The involvement of all members in software development team will reduce risk. This is because the nature of the work process and relations required more management involvement [48].

C21: Including Formal and Periodic Risk Assessment

According to Webern, et al. [49], risk analysis is a models for quantifying and evaluating a critical event occurrence. This is include a process of identifying relevant information of resources (software risk factors), discovering their relationships, and integrating them to form a risk assessment argument [50]. Hence, a model-based assessment that covers the formal and periodic risk should facilitate communication between internal and external factors in software project [51].

C22: Utilizing Change Control Board and Exercise Quality Change Control Practices

Contingency funds were managed centrally by the project through change control board procedures [52]. Really Fairley [20], can be defined Change Control Board as the minimum set of project stakeholders who need to review and approve any change request impacting the software project's critical success factors.

C23: Educating Users on the Impact of Changes during the Software Project

They integrated hardware/software approach is useful for educating users about software technology in software project is important to reduce risks [53].

C24: Ensuring that Quality-Factor Deliverables and Task Analysis

According to Bavani [54], ensuring high quality deliverables on schedule is important to mitigate risks in software project. Furthermore Keil, et al. [55], provided guidance on how to select members of review teams that help assure the quality of software project deliverables.

C25: Avoiding having too Many New Functions on Software Projects

Modern technical systems typically consist of multiple components and must provide many functions that are realized as a complex interaction of these components [56]. It is said that too many functions has difficult human interfaces for beginners, thus needs to implement new functionality on an incremental rather than too many new function [57].

C26: Incremental Development (Deferring Changes to Later Increments)

Increment development is not based on a certain scope (requirement subset) but is instead based on a measure of effort for improvement [58].

C27: Combining Internal Evaluations by External Reviews

Generally, the product will had internal evaluations by software project teams before delivering it to customers [54]. Moreover, reviewing, and evaluating strengths and weaknesses from a reviewer is one of the external factors to mitigate risk. The objective of internal and external is In addition, the objectives of external and internal is to have the consistency of all elements in software [59].

C28: Maintain Proper Documentation of Each Individual's Work

In the software industry, documented bi-directional traceability is needed needs to be maintained over the entire life cycle of the software project [60]. In addition, it is reported that substantial percentage amount of software firm do not maintain documented procedure for after sales service [61]. Overcome this issue can be treated with a control of the management process.

C29: Provide Training in the new Technology and Organize Domain Knowledge Training

According to Fairley [20], organizational training: To develop skills and knowledge among workers can perform their jobs efficiently and effectively.

C30: Participating Users during the Entire Software Project Lifecycle

Clearly, initiating user can be found from a group of users the one whose profile best matches to limit the risk [62]. This is because the set of participating users, hardware, and software in ubiquitous computing environments is highly dynamic and unpredictable [63]. The authors like [64] referred to participating users in the software development will enable more advantage during their communication with other user to specify the requirement.

4. EMPIRICAL STRATEGY

Data collection was achieved through the use of a structured questionnaire and historical data. Thirty risk management techniques were presented to respondents. The method of sample selection referred to as 'snowball' and distribution personal regular sampling was used. This procedure is appropriate when members of homogeneous groups (such as software project managers, IT managers) are difficult to locate. The software project managers that participated in this survey are coming from specific mainly software project manager in software development organizations. The factor analysis techniques used to classify the risk management techniques by the collected data. Respondents were presented with various questions, which used scale 1-7. For presentation purposes in this study and for effectiveness, the point scale as the following: Seven frequency categories were scaled into 'never' equal one and 'always' equal seven. Factor analysis

attempts to identify underlying variables, or factors, that explain the pattern of correlations within a set of observed variables. Also Factor analysis is often used in data reduction to identify a small number of factors that explain most of the variance observed in a much larger number of manifest variables (www.spss.com, 18/2/2013). We used the principal components method of extraction begins by finding a linear combination of variables (a component) that accounts for as much variation in the original variables as possible. Furthermore, to analyse the questionnaire inputs shown in below, we used the factor analysis approach, which is provided by SPSS statistical software.

4.1. Descriptive Statistics

Typically, the mean, standard deviation in the survey questionnaire are given. Looking at the mean, one can conclude that" reusable database and data mining structures" is the most important variable because it has the highest mean of 6.8.

Table-1. Descriptive Statistics

Variable	Mean	Std. Deviation
C1	5.50	1.504
C2	5.50	1.051
C3	5.60	1.188
C4	5.80	.768
C5	5.45	1.356
C6	5.50	1.539
C7	5.75	.639
C8	4.80	1.152
C9	4.40	1.353
C10	4.75	1.446
C11	4.75	1.372
C12	6.25	1.803
C13	6.20	1.704
C14	6.80	1.508
C15	6.15	1.954
C16	6.40	1.930
C17	6.35	1.899
C18	6.40	1.635
C19	6.40	2.113
C20	6.35	1.843
C21	6.20	1.908
C22	6.45	1.605
C23	6.25	1.293
C24	6.55	1.761
C25	6.45	1.468
C26	6.45	1.761
C27	6.40	1.698
C28	6.00	1.717
C29	6.05	1.731
C30	4.80	1.361

4.2. The Correlation Matrix

The next output from the factor analysis is the correlation coefficient. The correlation coefficient between a variable and itself is always one; hence, the principal diagonal of the correlation matrix contains 1s.

4.3. Communalities

However, communalities indicate the amount of variance in each variable that is accounted for Initial communalities are estimates of the variance in each variable accounted for by all components or factors. For principal components extraction, this is always equal to 1.0 for correlation analyses. The communalities in this Table 2 are all high, which indicates that the extracted components represent the variables well. Therefore, the next item from the output is a table of communalities, which shows how much of the variance in the variables has been accounted for by the extracted factors. For instance, over 92.8% of the variance in C5 is accounted for while 66.9% of the variance in C8 is accounted.

Table-2. Communalities

Variable	Initial	Extraction
C1	1.000	.906
C2	1.000	.802
C3	1.000	.761
C4	1.000	.884
C5	1.000	.928
C6	1.000	.772
C7	1.000	.862
C8	1.000	.669
C9	1.000	.790
C10	1.000	.885
C11	1.000	.788
C12	1.000	.868
C13	1.000	.706
C14	1.000	.729
C15	1.000	.855
C16	1.000	.891
C17	1.000	.858
C18	1.000	.790
C19	1.000	.911
C20	1.000	.888
C21	1.000	.870
C22	1.000	.802
C23	1.000	.736
C24	1.000	.829
C25	1.000	.757
C26	1.000	.844
C27	1.000	.897
C28	1.000	.755
C29	1.000	.854
C30	1.000	.782

Extraction Method: Principal Component Analysis.

4.4. Total Variance Explained

The next item shows all the factors extractable from the analysis along with their Eigen values, the percent of variance attributable to each factor, and the cumulative variance of the factor and the previous factors. Notice that the first factor accounts for 57.45% of the variance, the second factor 15.995% and the third factor8.789%. All the remaining factors are not significant. For the initial solution, there are as many components as variables, and in a correlations analysis, the sum of the eigenvalues equals the number of components. We have requested that eigenvalues greater than 1 be extracted, so the first three principal components of the extracted solution.

Table-3. Total Variance Explained

Component	Initial Eigenvalues			Extraction Sums of Squared Loadings			Rotation Sums of Squared Loadings		
	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %
1	17.235	57.450	57.450	17.235	57.450	57.450	13.064	43.548	43.548
2	4.799	15.995	73.446	4.799	15.995	73.446	5.886	19.621	63.169
3	2.637	8.789	82.235	2.637	8.789	82.235	5.720	19.065	82.235
4	.995	3.316	85.550						
5	.886	2.955	88.505						
6	.737	2.457	90.962						
7	.633	2.111	93.073						
8	.429	1.429	94.502						
9	.370	1.234	95.736						
10	.275	.916	96.652						
11	.229	.763	97.415						
12 –con. 30	4.99E-016	1.66E-015	100.000						

Extraction Method: Principal Component Analysis

4.5. Scree Plot

The eigenvalue of each component of the initial solution is plotted. The scree plot is a graph of the Eigenvalues against all the factors. It can be seen that the curve begins to flatten between components 3 and 4. Note also that factor 4 has an Eigen value of less than 1, the drop occurs between third and fourth components, so only the first three factors have been retained as Planning and requirement techniques, communication techniques, and models & tools.

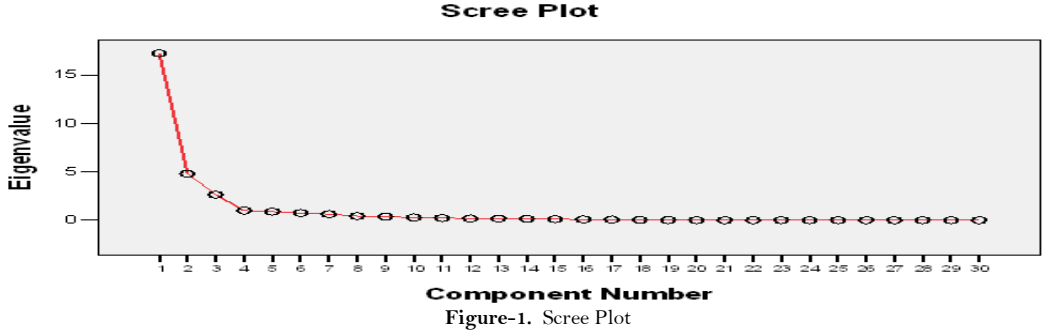

Figure-1. Scree Plot

4.6. Component (Factor) Matrix

Table 4 shows the loadings of the thirty variables on the three factors extracted. The higher the absolute value of the loading, the more the factor contributes to the variable. The gap on the

table represent loadings that are less than 0.5, this makes reading the table easier. We suppressed all loadings less than 0.5.

Table-4. Component Matrix

Variable	Component		
	1	2	3
C1	.756	.575	
C2	.815		
C3	.715		
C4	.508	.642	
C5	.829		
C6	.748		
C7			-.728
C8		.694	
C9	.584	.584	
C10		.597	.627
C11	.577	.647	
C12	.922		
C13	.816		
C14	.552	.534	
C15	.892		
C16	.938		
C17	.873		
C18	.867		
C19	.925		
C20	.919		
C21	.891		
C22	.823		
C23	.690	.509	
C24	.806		
C25	.736		
C26	.774		
C27	.894		
C28	.767		
C29	.907		
C30		.632	

Extraction Method: Principal Component Analysis.
a 3 components extracted.

4.7. Rotated Component (Factor) Matrix

The idea of rotation is to reduce the number factors on which the variables under investigation have high loadings. Also the rotated component matrix helps you to determine what the components Rotation does not actually change anything but makes the interpretation of the analysis easier. Looking at the Table 5, we can see that C1, C2, C3, C4, C5, C6 and are substantially loaded on Factor (Component) 1 while C8, C9, C10, C11 and C30 are substantially loaded on Factor (Component) 2. All remaining variables are substantially loaded on Factor (Component) 3. These factors can be used as variables for further analysis.

Table-5. Rotated Component Matrix

Variable	Component		
	1	2	3
C1			.507
C2			.585
C3			.585
C4			.841
C5			.800
C6			.740
C7			.927
C8		.803	
C9		.835	
C10		.927	
C11		.805	
C12	.794		
C13	.772		
C14	.812		
C15	.870		
C16	.827		
C17	.810		
C18	.767		
C19	.800		
C20	.785		
C21	.821		
C22	.805		
C23	.843		
C24	.862		
C25	.846		
C26	.918		
C27	.892		
C28	.789		
C29	.767		
C30		.864	

Extraction Method: Principal Component Analysis.
Rotation Method: Varimax with Kaiser Normalization.
a. Rotation converged in 6 iterations.

4.8. Comment

According to the factor analysis, there are three significant risk management techniques for mitigating risk factors, which are planning and requirement techniques (c1, c2, c3, c4, c5, c6, c7), communication techniques (c8, c9, c10, c11, c30), models and tools (c12, c13, c14, c15, c16, c17, c18, c19, c20, c21, c22, c23, c24, c25, c26, c27, c28, c29).

5. CONCLUSIONS

We presented an approach's factor analysis for classifying risk management techniques; we have chosen three components as planning and requirement techniques, communication techniques, and models and tools for thirty risk management techniques. The results also show that all risk management techniques are used most of the time, and often to mitigate risks. These tests were performed using factor analysis techniques, to classify the risk management techniques in a software project. As future work, we will intend to apply these study results on a real-world software project to verify the effectiveness of the risk management techniques on a software project for mitigating risks.

Funding: This work is supported by Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka (UTeM), Malaysia and Al-Aqsa University-Palestine

Competing Interests: The authors declare that they have no competing interests.

Contributors/Acknowledgement: All authors contributed equally to the conception and design of the study.

REFERENCES

[1] J. Miler and J. Górski, "Supporting team risk management in software procurement and development projects," presented at the 4th National Conference on Software Engineering, 2002.

[2] J. Sodhi and P. Sodhi, "It project management handbook: Management concepts (USA), ISBN:1-56726-098," p. 264, 2001.

[3] A. Elzamly, "Evaluation of quantitative and mining techniques for reducing software maintenance risks," *Appl. Math. Sci.*, vol. 8, pp. 5533–5542, 2014.

[4] K. Khanfar, A. Elzamly, W. Al-Ahmad, E. El-Qawasmeh, K. Alsamara, and S. Abuleil, "Managing software project risks with the chi-square technique," *Int. Manag. Rev.*, vol. 4, pp. 18–29, 2008.

[5] A. Elzamly and B. Hussin, "Managing software project risks with proposed regression model techniques and effect size technique," *Int. Rev. Comput. Softw.*, vol. 6, pp. 250–263, 2011.

[6] A. Elzamly and B. Hussin, "Estimating quality-affecting risks in software projects," *Int. Manag. Rev. Am. Sch. Press*, vol. 7, pp. 66–83, 2011.

[7] A. Elzamly and B. Hussin, "Managing software project risks (Implementation Phase) with proposed stepwise regression analysis techniques," *Int. J. Inf. Technol.*, vol. 1, pp. 300–312, 2013.

[8] A. Elzamly and B. Hussin, "Managing software project risks (Design Phase) with proposed fuzzy regression analysis techniques with fuzzy concepts," *Int. Rev. Comput. Softw.*, vol. 8, pp. 2601–2613, 2013.

[9] A. Elzamly and B. Hussin, "Managing software project risks (Planning Phase) with proposed fuzzy regression analysis techniques with fuzzy concepts," *Int. J. Inf. Comput. Sci.*, vol. 3, pp. 31–40, 2014.

[10] A. Elzamly and B. Hussin, "Identifying and managing software project risks with proposed fuzzy regression analysis techniques: Maintenance phase," presented at the 2014 Conference on Management and Engineering (CME2014), 2014.

[11] A. Elzamly and B. Hussin, "Managing software project risks (Analysis Phase) with proposed fuzzy regression analysis modelling techniques with fuzzy concepts," *J. Comput. Inf. Technol.*, vol. 22, pp. 131–144, 2014.

[12] A. Elzamly and B. Hussin, "Modelling and mitigating software implementation project risks with proposed mining technique," *Inf. Eng.*, vol. 3, pp. 39–48, 2014.

[13] A. Elzamly and B. Hussin, "Mitigating software maintenance project risks with stepwise regression analysis techniques," *J. Mod. Math. Front*, vol. 3, pp. 34–44, 2014.

[14] A. Elzamly and B. Hussin, "A comparison of stepwise and fuzzy multiple regression analysis techniques for managing software project risks : Analysis phase," *J. Comput. Sci.*, vol. 10, pp. 1725–1742, 2014.

[15] A. Elzamly and B. Hussin, "Acomparison of fuzzy and stepwise multiple regression analysis techniques for managing software project risks: Implementation phase," *Int. Manag. Rev.*, vol. 10, pp. 43–54, 2014.

[16] R. Dash and R. Dash, "Risk assessment techniques for software development," *Eur. J. Sci. Res.*, vol. 42, pp. 629–636, 2010.

[17] F. Reyes, N. Cerpa, A. Candia, and M. Bardeen, "The optimization of success probability for software projects using genetic algorithms," *J. Syst. Softw.*, vol. 84, pp. 775–785, 2011.

[18] A. Elzamly and B. Hussin, "An enhancement of framework software risk management methodology for successful software development," *J. Theor. Appl. Inf. Technol.*, vol. 62, pp. 410–423, 2014.

[19] S. McConnell, *Rapid development: Taming wild software schedules*: Microsoft Press, 1996.

[20] R. Fairley, *Managing and leading software projects*: Wiley-IEEE Computer Society Press, ISBN: 978-0-470-29455-0, 2009.

[21] J. Ferraro, *Project management techniques for non-project managers*: AMACOM, ISBN: 0814417361, 2012.

[22] T. Addison and S. Vallabh, "Controlling software project risks – an empirical study of methods used by experienced project managers," in *Proceedings of SAICSIT*, 2002, pp. 128 – 140.

[23] K. Na, J. Simpson, X. Li, T. Singh, and K. Kim, "Software development risk and project performance measurement: Evidence in Korea," *J. Syst. Softw.*, vol. 80, pp. 596–605, 2007.

[24] J. Ropponen and K. Lyytinen, "Components of software development risk: How to address them? A project manager survey," *IEEE Trans. Softw. Eng.*, vol. 26, pp. 98–112, 2000.

[25] K. R. Linberg, "Software developer perceptions about software project failure: A case study," *J. Syst. Softw.*, vol. 49, pp. 177–192, 1999.

[26] A. Puntambekar, *Software engineering*: Technical Publications, ISBN: 8184316054, 2009.

[27] P. Savolainen, J. Ahonen, and I. Richardson, "Software development project success and failure from the supplier's perspective: A systematic literature review," *Int. J. Proj. Manag.*, vol. 30, pp. 458–469, 2012.

[28] B. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy, and R. Selby, "Cost models for future software life cycle processes: COCOMO 2.0," *Ann. Softw. Eng.*, vol. 1, pp. 57–94, 1995.

[29] D. Surie, "Evaluation and integration of risk management in CMMI and ISO / IEC 15504," presented at the 8th Student Conference in Computing Science, Umeå, Sweden, 2004.

[30] D. Dufner, O. Kwon, and A. Doty, "Improving software development project team performance: A web-based expert support system for project control," in *Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences (HICSS-32)*, 1999, pp. 1-10.

[31] V. Anantatmula, "Project planning techniques for academic advising and learning," *Int. J. Scholarsh. Teach. Learn*, vol. 6, pp. 1-19, 2010.

[32] F. Sarfraz, "Managing for a successful project closure," presented at the PICMET '09 - 2009 Portland International Conference on Management of Engineering & Technology, 2009.

[33] S. Grabski, S. Leech, and B. Lu, "Risks and controls in the implementation of ERP systems," *Int. J. Digit. Account. Res.*, vol. 1, pp. 47–68, 2001.

[34] G. Schulmeyer, "Handbook of software quality assurance," presented at the Fourth. ARTECH HOUSE, INC, 2008.

[35] J. Jianga, G. Kleinb, H. G. Chenc, and L. Lind, "Reducing user-related risks during and prior to system development," *Int. J. Proj. Manag.*, vol. 20, pp. 507–515, 2002.

[36] C. Jones, *Applied software measurement global analysis of productivity and quality*, 3rd ed.: McGraw-Hill Companies, ISBN: 0071502440, 2008.

[37] J. Kasurinen, O. Taipale, and K. Smolander, "Test case selection and prioritization: Risk-based or," in *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement ESEM '10*, 2010, pp. 1–10.

[38] O. Miotto, T. Tan, and V. Brusic, "Supporting the curation of biological databases with reusable text mining," *Genome Informatics*, vol. 16, pp. 32–44, 2005.

[39] P. Kanjanasanpetch and B. Lgel, "Managing knowledge in enterprise resource planning (ERP) implementation," presented at the Managing Technologically Driven Organizations: The Human Side of Innovation and Change, Engineering Management Conference, IEMC '03, 2003.

[40] R. M. Shand, "User manuals as project management tools: Part II-aractical applications," *IEEE Trans. Prof. Commun.*, vol. 3, pp. 123–142, 1994.

[41] R. Green, "Documentation meets version control: An automated backup system for HTML-based help," in *Professional Communication Conference, Proceedings of 2000 Joint IEEE International and 18th Annual Conference on Computer Documentation (IPCC/SIGDOC 2000)*, 2000, pp. 541–548.

[42] K. Kouskouras and A. Georgiou, "A discrete event simulation model in the case of managing a software project," *Eur. J. Oper. Res.*, vol. 181, pp. 374–389, 2007.

[43] G. Jiang and Y. Chen, "Coordinate metrics and process model to manage software project risk," presented at the IEEE International Engineering Management Conference, 2004.

[44] D. Liu, Q. Wang, and J. Xiao, "The role of software process simulation modeling in software risk management: A systematic review," presented at the Third International Symposium on Empirical Software Engineering and Measurement, 2009.

[45] S. Alhawari, F. Thabtah, L. Karadsheh, and W. Hadi, "A risk management model for project execution," presented at the The 9th IIBIMA Conference on Information Management in Modern Organizations, 2008.

[46] R. Schmidt, K. Lyytinen, M. Keil, and P. Cule, "Identifying software project risks: An international delphi study," *J. Manag. Inf. Syst.*, vol. 17, pp. 5–36, 2001.

[47] A. Azari, N. Mousavi, and S. Mousavi, "Risk assessment model selection in construction industry," *Expert Syst. Appl.*, vol. 38, pp. 9105–9111, 2011.

[48] T. Dyba and T. Dingsoyr, "Empirical studies of agile software development: A systematic review," *Inf. Softw. Technol.*, vol. 50, pp. 833–859, 2008.

[49] P. Webern, G. Medina-Oliva, C. Simon, and B. Iung, "Overview on bayesian networks applications for dependability, risk analysis and maintenance areas," *Eng. Appl. Artif. Intell.*, vol. 42, pp. 115–125, 2010.

[50] S. Lee, "Probabilistic risk assessment for security requirements: A preliminary study," presented at the Fifth International Conference on Secure Software Integration and Reliability Improvement, 2011.

[51] J. Aagedal, F. Braber, T. Dimitrakos, B. Gran, D. Raptis, and K. Stølen, "Model-based risk assessment to improve enterprise security," in *Proceedings of The Sixth International Enterprise Distributed Object Computing Conference (EDOC'02)*, 2002, p. 12.

[52] C. Strawbridge, "Project management in large collaborations: SNS lessons learned for ITER," presented at the Twenty-First IEEE/NPS Symposium on Fusion Engineering, 2005.

[53] K. Persohn and D. Brylow, "Interactive real-time embedded systems education infused with applied internet telephony," presented at the 35th IEEE Annual Computer Software and Applications Conference, 2011.

[54] R. Bavani, "Global software engineering: Challenges in customer value creation," presented at the 2010 International Conference on Global Software Engineering, 2010.

[55] M. Keil, L. Li, L. Mathiassen, and G. Zheng, "The influence of checklists and roles on software practitioner risk perception and decision-making," *J. Syst. Softw.*, vol. 81, pp. 908–919, 2008.

[56] J. Greenyer, A. Sharifloo, M. Cordy, and P. Heymans, "Efficient consistency checking of scenario-based product-line specifications," presented at the 20th IEEE International Requirements Engineering Conference (RE), 2012.

[57] M. Oda, "The characteristics of the use of twitter by beginners: Study of the applicability to the e-healthcare," presented at the International Conference on Systems, Man, and Cybernetics (SMC), IEEE, 2010.

[58] D. M. Brandon, *Project management for modern information systems*: IRM Press, Idea Group Inc., PSBN:1-59140-693-5, 2006.

[59] A. Peppen and M. Ploeg, "Practicing what we teach: Quality management of systems-engineering education," *IEEE Trans. Syst. Man, Cybern. C Appl. Rev.*, vol. 30, pp. 189–196, 2000.

[60] J. Chen and S. Huang, "An empirical analysis of the impact of software development problem factors on software maintainability," *J. Syst. Softw.*, vol. 82, pp. 981–992, 2009.

[61] Z. Begum, M. Khan, M. Hafiz, M. S. Islam, and M. Shoyaib, "Software development standard and software engineering practice: A case study of Bangladesh," *J. Bangladesh Acad. Sci.*, vol. 32, pp. 131–139, 2008.

[62] M. Li, S. Yu, N. Cao, and W. Lou, "Privacy-preserving distributed profile matching in proximity-based mobile social networks," *IEEE Trans. Wirel. Commun.*, vol. 12, pp. 2024– 2033, 2013.

[63] V. Cahill, A. Fox, T. Kindberg, and B. Noble, "Building and evaluating ubiquitous system software," *Pervasive Comput. IEEE CS IEEE Com Soc.*, vol. 3, pp. 20–21, 2004.

[64] J. Jin and B. Li, "Cooperative multicast scheduling with random network coding in WiMAX," presented at the 2009 17th International Workshop on Quality of Service, 2009.