Journal of Computer Science 10 (10): 1725-1742, 2014
ISSN: 1549-3636

doi:10.3844/jcssp.2014.1725.1742 Published Online 10 (10) 2014 (http://www.thescipub.com/jcs.toc)

A COMPARISON OF STEPWISE AND FUZZY MULTIPLE REGRESSION ANALYSIS TECHNIQUES FOR MANAGING SOFTWARE PROJECT RISKS: ANALYSIS PHASE

[1]Abdelrafe Elzamly and [2]Burairah Hussin

[1,2]Information and Communication Technology, Universiti Teknikal Malaysia Melaka (UTeM), Malaysia
[1]Department of Computer Science, Faculty of Applied Sciences, Al-Aqsa University, Gaza, Palestine

Received 2014-01-18; Revised 2014-03-25; Accepted 2014-04-22

ABSTRACT

Risk is not always avoidable, but it is controllable. The aim of this study is to identify whether those techniques are effective in reducing software failure. This motivates the authors to continue the effort to enrich the managing software project risks with consider mining and quantitative approach with large data set. In this study, two new techniques are introduced namely stepwise multiple regression analysis and fuzzy multiple regression to manage the software risks. Two evaluation procedures such as MMRE and Pred (25) is used to compare the accuracy of techniques. The model's accuracy slightly improves in stepwise multiple regression rather than fuzzy multiple regression. This study will guide software managers to apply software risk management practices with real world software development organizations and verify the effectiveness of the new techniques and approaches on a software project. The study has been conducted on a group of software project using survey questionnaire. It is hope that this will enable software managers improve their decision to increase the probability of software project success.

Keywords: Software Project Management, Software Risk Management, Software Risk Factors, Risk Management Technique, Stepwise Regression Analysis Techniques, Fuzzy Multiple Regression Analysis, Analysis Phase

1. INTRODUCTION

Despite much research and progress in the area of software project management, software development projects still fail to deliver acceptable systems on time and within budget. For some of these reasons corrective action is often complex to cost-justify or to implement efficiently in a software practice (Masticola, 2007). According to (Yassin, 2010), identifying the risks that facing software projects and reasons behind their failure has haunted project managers, software industry consultants and academician for a long time. Therefore, management is still unable to effectively manage the risks involved in these software projectsDue to the involvement of risk management in monitoring the success of a software project, analyzing potential risks and making decisions about what to do with potential risks, the risk management is considered the planned control of risk. In addition, risk is an uncertainty that can have a negative or positive effect on meeting project objectives. This study incorporates between risk management approach and software development life cycle to mitigate software failure. Risk management is the process of identifying, analyzing and controlling risk throughout the lifecycle of a software project to meet the software project objectives (Schwalbe, 2010). Clearly, the success or failure of software projects is generally assessed with dimensions such as budget, schedule and quality (Miler, 2005). In this study, we identify software risk factors and risk management techniques on a large

Corresponding Author: Abdelrafe Elzamly, Department of Computer Science, Faculty of Applied Sciences, Al-Aqsa University, Gaza, Palestine

set that are guided software project managers to mitigate risks in a software project. According to (Hoffer *et al*., 2011), Software Development Life Cycle (SDLC) is the process of creating and methodologies that can use to develop a software project which including phases as planning, analysis, design, implementation and maintenance. In addition, we focused on analysis phase: It includes looking at any existing system to see what it is doing for the organization and how well that system is doing its job. According to (Taylor, 2004), we should apply approaches and techniques consistently throughout the software risk management. Risk management is a practice of controlling risk such as processes, tools and methods for managing risks in a software project before the occuring of risks (Sodhi and Sodhi, 2001). Therefore, previous study had considered many aspect of risk management approach, including principles and practices for risk identification, risk analysis, risk prioritization and risk mitigation (Boehm, 2003).

The objective of this study is: Only identify software risk factors and risk management techniques in analysis phase is considered from various literatures, to rank the software risk factors and risk management techniques according to their importance, severity and occurrence frequency, later, the realtionship between the software risk factors and risk management techniques is develop using stepwise multiple regresion anlayis and fuzzy multiple regression analysis, the comparison between botth techniques is conducted using evaluation techniques such as MMRE and Pred (25).

The organization of this study as will be as follows. Section 2% an overview of the literature. Section 3 introduces the software risk factors (analysis phase) relevant to the study. Section 4 introduces the most common risk management techniques to these risks. Section 5% the empirical work. Section 6 concludes the article and glimpses on future work.

2. LITERATURE REVIEW

There is no structure way of managing software risk, project manager using their experience, opinion and self judgment to mitigate risk. Hence, techniques or model to mitigate software risks in software project is necessary. In the literature, many considered on mitigate risk by qualittive and quantitative technques, but rarely combine between software development life cycle and risk mannagement based on quantitative and mining techniques to mitigate software risks. In addition, a few authors combine between software risk factors and risk management techniques to reduce riks in SDLC. However (Khanfar *et al*., 2008), used chi-square ($\chi 2$) to mitigate risks in a software project by using control factors. And proposed new techniques the regression test and effect size test to manage the risks in a software project (Elzamly and Hussin, 2011a). Furthermore, the new stepwise regression technique used to manage the risks in a software project (Elzamly and Hussin, 2013a). Indeed, the multiple regression analysis techniques with fuzzy concepts is used to mitiagte the risks in a software project in design phase (Elzamly and Hussin, 2013b). In addition, they proposed Techno-Portfolio Advisor to helps the investors to understand the critical relations and support mutual funds selection during the Asset Management Companies in India. Further the new mining technique that uses the fuzzy regression analysis modelling techniques to manage the risks in a software planning development project. Top ten software risk factors in planning phase and thirty risk management techniques were presented to respondents (Elzamly and Hussin, 2014b).

However, Oracle corporation described risk management solutions enable a standardized approach for identifying risk, assessing risk and mitigating risk throughout the software project lifecycle (Oracle, 2010). Previous studies had shown that risk mitigation in software project can be classified by 3 categories such as qualitative, quantitative and mining approaches. Quantitative risk is based on statistical methods that deal with accurate measurement about risk or leading to quantitative inputs, that helped forming a regression model to understand how software project risk factors influence project success (Bhoola *et al*., 2014) such as network analysis and regression models and other objective approach, but qualitative risk techniques lead to subjective opinions expressed or self judgment by software manager (Bhoola *et al*., 2014) such as scenario, Delphi analysis, brainstorming session and other subjective approach to mitigate risks. Mining approach is a new way of identifying risk from data that create relationships between data and find the optimum result from them. This includes techniques such as simulation analysis, fuzzy logic models, fuzzy multiple regression, neural network models, genetic algorithm and heuristic algorithm. The goal of mining and statistic techniques is predicted to select the best model based on modelling and their prediction accuracy to mitigate risks. The new framework software risk management methodology proposed for successful software project that including 5 phases such as

identification risk, risk analysis and evaluation, risk treatment, risk controlling, risk communication and documentation for software development life cycle which relied on three categories techniques as qualitative analysis, quantitative analysis and mining analysis to meet the goals (Elzamly and Hussin, 2014a).

3. TOP 10 SOFTWARE RISK FACTORS (ANALYSIS PHASE)

We display the top ten software risk factors in software development lifecycle (analysis phase) that common in the literature review. We present 'top-ten' based on (Boehm, 1991; Miler, 2005), etc. 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. 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 software risk. In this section the top software risk associated with analysis phase is discussed. In addition, the software risk factors (analysis phase) listed below in **Table 1** are considered in this study.

Risk 01: Unclear, Incorrect, Continually and Rapidly Changing Software Project Requirements

This risk is mentioned clearly by several authors such as (Addison, 2003; Boehm, 2001; 2007; CHAOS, 1995; Chen and Huang, 2009; Elzamly and Hussin, 2011a; Han and Huang, 2007; Jalote, 2002; Keil *et al*., 2002; Ewusi-Mensah, 2003; Maglyas, 2009; Schmidt *et al*., 2001; Sodhi and Sodhi, 2001; Sumner, 2000) in the work. It means that uncontrolled and unpredictable change of system functions, features and essential requirements is contributed to software fail (Boehm, 1991; Elzamly and Hussin, 2011a; Jalote, 2002; Ropponen and Lyytinen, 2000; Selby, 2007). The continuous changing requirements can affect the cost, schedule, scope, budget and quality of a software project lead to inconsistence the software requirements (Hayat *et al*., 2010).

Risk 02: Failure to Incomplete or Missing Detailed Requirements Analysis and Specification Documentation

This risk is referred to incomplete or missing detailed requirements as terminations from this source are more likely to come from mismanaged software projects as reported by (Boehm, 2001; 2007; CHAOS, 1995; Chen and Huang, 2009; Elzamly and Hussin, 2011a; 2011b; Khanfar *et al*., 2008; Maglyas, 2009; Sumner, 2000). In the missing detailed requirements analysis, the attributes are unverified through inadequate measuring. Indeed, missing detailed requirements analaysis will lead unvaluable input for measuring the resources required to develop the software development lifecycle (Galorath, 2006; Keil *et al*., 1998).

Risk 03: Developer Software Gold-Plating

According to (Boehm, 1991; Dash and Dash, 2010; Elzamly and Hussin, 2011a; Khanfar *et al*., 2008; Maglyas, 2009; Sodhi and Sodhi, 2001; Surie, 2004), reported that developer when taking work on a software project put an extra effort and put value in it Therefore, it focused as a bad software project management practices, techniques and methods as this approach give to extra adding unnecessary features occur to software project because of professional interest or user's demands (Horine, 2009; Ropponen and Lyytinen, 2000). Furthermore, developer software gold plating can result in wasting resources on implementing functionality that is not of real value or that's never actually used (Westfall, 2006). This would introduce new quality risks into the software project but to do nothing to improve the actual deliverable quality, yet they can require additional time and costs (Fairley, 2009; Horine, 2009).

Risk 04: Lack of IT Management

Referred to (Boehm and Basili, 2001; CHAOS, 1995; Ewusi-Mensah, 2003; Nakatsu and Iacovou, 2009) lack of IT management is another risk factor to consider in software developmnet. IT management is the process that allows software project managers to balance the operational and economic costs of software project (Aziz and Salleh, 2011; Rodriguez-Repisoa *et al*., 2007; Veracode, 2008). The lack of IT management may lead to inconsistencies in software system requirements, unfollowing communication plan and failure of follow good practices and policies in an organized manner (Lientz and Larssen, 2006).

Risk 05: Software Project Requirements not Adequately Identified and Mismatch

According to (Boehm, 2007; Han and Huang, 2007; Nakatsu and Iacovou, 2009; Schmidt *et al*., 2001; Wallace and Keil, 2004), software project requirements not adequately identified and found a mismatch between them.

Table 1. Illustrate top ten software risk factors in analysis phase based on researchers

Phase	No	Software risk factors (analysis phase)	Frequency
Analysis	1	Unclear, incorrect, continually and rapid changing software project requirements (Addison and Vallabh, 2002; Addison, 2003; Aloini *et al*., 2007; Boehm, 1991; 2002a; 2002b; 2007; CHAOS, 1995; Chen and Huang, 2009; Elzamly and Hussin, 2011a; Han and Huang, 2007; Keil *et al*., 2002; Khanfar *et al*., 2008; Ewusi-Mensah, 2003; Nakatsu and Iacovou, 2009; Schmidt *et al*., 2001; Sumner, 2000), Boehm's Top 10 Risk Items: 1989 and 1995 survey (Boehm, 2002b)	19
	2	Failure to incomplete or missing detailed requirements analysis (Addison, 2003; Boehm, 2007; CHAOS, 1995; Han and Huang, 2007; Keil *et al*., 2002; Khanfar *et al*., 2008; Nakatsu and Iacovou, 2009; Schmidt *et al*., 2001; Sumner, 2000).	9
	3	Developer software gold-plating (Addison and Vallabh, 2002; Aloini *et al*., 2007; Boehm, 1991; 2002b; Elzamly and Hussin, 2011a; 2011b; Khanfar *et al*., 2008)	7
	4	Lack of IT management (Addison, 2003; Boehm, 1991; 2002b; CHAOS, 1995; Ewusi-Mensah, 2003; Nakatsu and Iacovou, 2009)	6
	5	Software project requirements not adequately identified and mismatch (Boehm, 2002a; 2002b; 2007; Han and Huang, 2007; Nakatsu and Iacovou, 2009; Schmidt *et al*., 2001)	6
	6	Inadequate knowledge about tools and programming techniques (Aloini *et al*., 2007; Chen and Huang, 2009; Elzamly and Hussin, 2011a; Ewusi-Mensah, 2003; Nakatsu and Iacovou, 2009)	5
	7	Lack of traceability, confidentiality, correctness and inspection of the software project planning (Addison, 2003; Aritua *et al*., 2011; Chen and Huang, 2009; Elzamly and Hussin, 2011a)	4
	8	Major requirements change after software project plan phase (Boehm, 1991; 2002a; Ewusi-Mensah, 2003)	3
	9	Changing software project specifications (CHAOS, 1995; Elzamly and Hussin, 2011b)	2
	10	Inadequate value analysis to measure progress (Aloini *et al*., 2007; Han and Huang, 2007)	2
	Total frequency		63

The software projects is most likely to perform poorly if software manager is unable to effectively manage the requirements of the software project life cycle and does not well-plan nor monitor the software risk management plan (Han and Huang, 2007). therefore, software project manager need to better manage the requirements that change or evolve in a software project (Verner *et al*., 2006).

Risk 06: Inadequate Knowledge about Tools and Programming Techniques

According to (Aloini *et al*., 2007), the inadequate use of tools for software project. In addition, lack of tools and methods in software programming is also abig contributor to software risk (Bennatan, 2006; Chen and Huang, 2009; Elzamly and Hussin, 2011b; Ewusi-Mensah, 2003; Nakatsu and Iacovou, 2009; Pandian, 2006). Thus, adequate knowledge about tools and techniques of software project may lead to better practices for software development lifecyle.

Risk 07: Lack of Traceability, Confidentiality, Correctness and Inspection of the Software Project Analysis

Addison (2003) reported that insufficient procedures to ensure transaction traceability, confidentiality and correctness is the most common risk in analysis stage. In addition (Aritua *et al*., 2011; Chen and Huang, 2009; Elzamly and Hussin, 2011b; Kontio, 2001) also referred to lack of traceability such as difficult to trace back to design specifications and user requirements as another changlleges. Traceability establishes logical links between two work process stated by (Fairley, 2009). Therefore, the lack of traceability between requirements and tests will resulted in unfocused or incomplete testing (Sodhi and Sodhi, 2001).

Risk 08: Major Requirements Change after Software Project Plan Phase

This risk relates to uncontrolled software requirements and inconsistency of software requirement, with plan phase as reported by (Boehm, 1991; Elzamly and Hussin, 2011b; Fairley, 2009; Ewusi-Mensah, 2003; Sodhi and Sodhi, 2001). Consequently, it may be lead to slow the progress to date and delays the objectives for the next.

Risk 09: Changing Software Project Specifications

According to (Blackmore and Nesbitt, 2008; CHAOS, 1995; Elzamly and Hussin, 2011b; Grady, 2010; Kumar, 2002), software project specifications are

corrected when necessary, by means of either a change or a revision (Grady, 2010). This will lead to request bout new functionality and need to rewrite the specification many times during software project lifecycle (Maglyas, 2009). As aconsequences will change specifications unimproved consistency, increase development time and limitatation of available implementation technology (Schulmeyer, 2008).

Risk 10: Inadequate Value Analysis to Measure Progress

Accordingly (Aloini *et al*., 2007; Elzamly and Hussin, 2011b; Jones, 2009; Ewusi-Mensah, 2003; Pandian, 2006), measurement is needed to measure software project progress. It is reported by (Han and Huang, 2007; Huang and Han, 2008), if a software project progress not monitored closely enough. It may lead to software failure.

4. RISK MANAGEMENT TECHNIQUES

Through reading the existing literature on software risk management, we listed the most common thirty risk management techniques that are considered important in mitigating the software risk factors (analysis phase) identified; these controls are:

C1: Using of Requirements Scrubbing

Requirements scrubbing is a best practice for software projects in which a product specification is carefully examined for unnecessary or overly complex requirements, which are then removed (Boehm and Ross, 1989; Boehm, 1991; McConnell, 1996). This is believed the reasons as the process of reviewing each requirement in detailed absolutely necessary for the upcoming release and it can dramatically increase the chances of delivering software project on-time and within budget (Fairley, 2009; Miller, 2004).

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 (Ferraro, 2012). Many software projects are faced with uncertainty when software requirements are first stated (Addison and Vallabh, 2002). However, they referred to stabilize requirements and specifications as early as possible as a control factor (Khanfar *et al*., 2008).

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

According to (Na *et al*., 2007; Ropponen and Lyytinen, 2000), they found that risk is positively related to both cost and schedule overruns. Hence, estimating cost and software project schedule impact is important to mitigate risk requirements and spicifications and the successful software development (Jones, 2008; Kamaruddin, 2006; Linberg, 1999).

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 sofwtare (Puntambekar, 2009). It is a software model that created to represent a user interface or a function for the purpose of better understanding the requirements and the feasibility of the proposed solution (Tsui, 2004). In aaddition, it is clear that building early prototypes can help coin out some changes software development lifecycle (Boehm, 1991). This is reported by (McConnell, 1996; Savolainen *et al*., 2012), as prototpying can reduce requirements creep and can be combined with other approaches (McConnell, 1996; Savolainen *et al*., 2012). Furthermore, prototyping approach can used to mitigate risk issues as user interfaces, software/system interaction, or software performance (Boehm *et al*., 1995; Surie, 2004).

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 (Addison and Vallabh, 2002; Dufner *et al*., 1999). Software project planning should be allocated to each of these sofwtare phases to manage and reduce potential occering failure (Cantor, 1998; Westfall, 2001). In addition, he proposed application of software planning techniques to manage the multiple problems and the complexity associated with software planning (Anantatmula, 2010). A risk management plan techniques will lead to mitigate the potential occurring risks and the overall impact of risks in software project (Westfall, 2001). Finally, they referred to need for planning approach to mitiagite mutilple risk in software project (Chapman and Thomas, 2007).

C6: Implementing and Following a Communication Plan

Communication plan is crucial for monitoring progress (Sarfraz, 2009) as each individual should feel comfortable to provide inputs on raised problems. Progress information should be shared with all concerned during or at the completion of each task before moving forward to the next. Risk management communication planning techniques implemented to be a continuous feedback loop through extra information risk and developed (Westfall, 2001).

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 (Addison and Vallabh, 2002; Westfall, 2001). Furthermore (RM: GBP, 2003), contingency plans are developed as a result of a risk failure being identified and pre-defined action plans that able to be implement if identified risks occur really. Creating risk contingency plans is risk mitigation for the facility or group of facilities to be reduced risks (Chapman and Thomas, 2007).

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 (Addison and Vallabh, 2002; Grabski *et al*., 2001). It is importanat to assign the responsibilities clearly for the appropriate performing organizations in the early stage with lead (Schulmeyer, 2008). It is also sometimes better to rotate developers and leaders the sections of the software project development to gain a variety of experiences (Lientz and Larssen, 2006; Sodhi and Sodhi, 2001). The software project team member's roles and responsibilities have been well established to identify and address issues (Cantor, 1998).

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 that reportected by (Boehm, 1991; Holcombe, 2008; Jianga *et al*., 2002; Tomczyk, 2005).

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 that reported by (Kouskouras and Georgiou, 2007; Ma *et al*., 2009; Munch and Heidrich, 2004; Sarfraz, 2009; Sodhi and Sodhi, 2001; Tayntor, 2006).

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 (Addison and Vallabh, 2002). According to (SPM: MT, 2004), 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, 2008; Sodhi and Sodhi, 2001), 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 (Galorath and Evans, 2006).

C13: Reusable Test Plans and Test Cases

A pre-release defect can be found in any of the software project (Emam, 2005; Jones, 2008). Hence, reusable test plans and cases would speed up the process of creating testability of test plans and allow an easier test case generation (Kasurinen *et al*., 2010).

C14: Reusable Database and Data Mining Structures

According to (Jones, 2008), 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 (Miotto *et al*., 2005).

C15: Reusable user Documents Early

According to (Jones, 2008), referred to reusable user documents. In addition (Kanjanasanpetch and Igel,

2003), 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 a strandard communication a pproach (Shand, 1994).

C16: Implementing/Utilizing Automated Version Control Tools

According to (Cantor, 1998; Green, 2000), software developers need to have a version control systems for manage source code changes (Green, 2000). the version control tools are to able to track evolving versions of a project's work products and testing tools to aid in verifying the software (Fairley, 2009). Fairly also commented that automated version control is essential for establishing and maintaining the baselines of various work products in various stages of development.

C17: Implement/Uilize Benchmarking and Tools of Technical Analysis

According to (Hill, 2010), providing information and practical estimating techniques-primarily based on the International Software Benchmarking Standards Group will assist project managers with the task of estimating the three key variables that follow the establishment of software project requirements, namely: Size, effort and duration. In addition (Jones, 2008), 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

Modeling and simulation of software development processes is gaining an increasing demand to reduce risks that focuses on a specific software development/maintenance/evolution process (Kouskouras and Georgiou, 2007; Surie, 2004). In addition (Jiang and Chen, 2004), described the process model simulation on risk occurrence probability do have an impact in software project. Furthermore, software processing simulation modeling (SPSM) has been emerging as a promising approach to address a variety of issues in software engineering area, including risk management (Liu *et al*., 2009).

C19: Provide Scenarios Methods and Using of the Reference Checking

According to (Alhawari *et al*., 2008), described risk analysis phase by conducting scenarios for major risks and events to establishes a probability of losses for every risk scenario. However (Schmidt *et al*., 2001), 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 (Azari *et al*., 2011; Smith *et al*., 2006).

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 (Dyba and Dingsoyr, 2008; Lyons and Skitmore, 2004).

C21: Including Formal and Periodic Risk Assessment

According to (Webern *et al*., 2010), 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 (Lee, 2011). Hence, a model-based assessment that covers the formal and periodic risk should facilitate communication between internal and external factors in software project (Aagedal *et al*., 2002).

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

Contingency funds were managed centrally by the project through change control board procedures (Strawbridge, 2005). Author like (Kandt, 2003) reported that requirements should be managed using a defined configuration management process that uses change control boards and automated change control tools that manage each requirement. Realy (Fairley, 2009; Horine, 2009; Hayat *et al*., 2010; IEEE, 2011; Kandt, 2003), 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 (Persohn and Brylow, 2011). Software risk management is affected by several factors as well educated practitioners for usres (Islam, 2009).

C24: Ensuring that Quality-Factor Deliverables and Task Analysis

According to (Bavani, 2010; Tsui, 2004), ensuring high quality deliverables on schedule is important to mitigate risks in software project. Furthermore (Keil *et al*., 2008), 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 (Greenyer *et al*., 2012). 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 (Oda, 2011).

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 (Brandon, 2006). Genarlly, stakeholders specify the requirements and evaluate the increment, thus, the change suggested in the current increment are implemented in the next increment (Puntambekar, 2009).

C27: Combining Internal Evaluations by External Reviews

Generally, the product will had internal evaluations by software project teams before delivering it to customers (Bavani, 2010). 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 (Peppen and Ploeg, 2000).

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 (Chen and Huang, 2009). In addition, it is reported that substantial percentage amount of software firm do not maintain documented procedure for after sales service (Begum *et al*., 2008). 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, 2009), organizational training: To develop skills and knowledge among workers can perform their jobs efficiently and effectively. Furthermore, training plans should be developed and implemented to ensure that all personnel that involved in service management initiatives are given the opportunity to develop their skills, knowledge and competences (Rudd, 2010).

C30: Participating Users During the Entire Software Project Lifecycle

According to (Dzung and Briod, 2005; Li *et al*., 2013), initiating user can be found from a group of users the one whose profile best matchesto limit the risk. This is because the set of participating users, hardware and software in ubiquitous computing environments is highly dynamic and unpredictable (Cahill *et al*., 2004). The authers like (Jin and Li, 2009) referred to participating users in the software development will enable more advantage during their communication with other user to specify the requirement.

5. EMPIRICAL STRATEGY

Data collection was achieved through the use of a structured questionnaire in estimating the quality of software. Top ten software risk factors and thirty control factors 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 seventy six software project managers have participated in this study.

Respondents were presented with various questions, which used scales1-7. All questions in software risk factors were measured by a 7-point Likert scale from

unimportant to extremely important and risk management techniques were measured by a seven-point scale from never to always. The data is collected from various to IT manager, software project managers from software organazitions in Palestine. In order to find the relation among risks and risk management techniques, we introduce stepwise multiple regression analysis, fuzzy multiple regression, later the evalaution techniques that the best way to measure error, compare acccuracy level in models such as MMRE and Pred (l).

5.1. Stepwise Regression (Adds and Removes Variables)

According to (Lan and Guo, 2008; Lin *et al*., 2012), stepwise regression method combines and alternates between forward selection and backward elimination. At each step, the best remaining variable is added, provided it passes the significant at 5% criterion, then all variables currently in the regression are checked to see if any can be removed. Also, the stepwise- multiple regression method that systematically adds and removes modal elements depend on statistical test to automatically identify the risks for a large scale data in operation (Zhou *et al*., 2012). Therefore (Lin *et al*., 2012), this technique is particularly useful when we need to predict a set of dependent variables from a large set of independent variables.

5.2. Regression Analysis Model with Fuzzy Concepts

Fuzzy multiple regression analysis is an extension of the traditional regression analysis in which some elements of the models are represented by fuzzy numbers (Dom *et al*., 2012). On the other words, fuzzy multiple regression analyis in that response variable is fuzzy variable and part of the covariates are crisp variables (Lin *et al*., 2012). However, identify the various data types that may appear in a questionnaire. Thus, we introduce the survey questionnaire data mining risk and define the rule patterns that can be mined from survey questionnaire data (Chen and Huang, 2009). Therefore, we must extend the crisp association rules to fuzzy association rules from questionnaire data.

5.3. Fuzzy Concepts with Membership Function

Fuzzy concepts help to find the deviation of each data from goodness equation, so we define a normal distribution membership function as follow (Marza and Seyyedi, 2009) Equation 1:

$$U_i = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{Yi-\mu}{\sigma}\right)^2} \quad (1)$$

Where:
μ = Average of sample points and
σ = Square root of variance math

If we add fuzzy domain to regression technique, the effect of discrete data points on the goodness result will be reduced and the effect of concentrated data points on the fitness result will be enhanced. Indeed, a membership function is a curve that defines how each point in the input space is mapped to a membership value between 0 and 1 (Dom *et al*., 2012).

5.4. Fuzzy Parameters

A group of these equations to obtain the fuzzy parameters are provided as (Gu *et al*., 2006; Popescu and Giuclea, 2007) Equation 2:

$$\begin{aligned}
&s11b1+s12b2+\ldots\ldots\ldots\ldots+s1kbk = s1y\\
&s21\ b1 +s22b2+\ldots\ldots\ldots\ldots+s2kbk = s2y\\
&s31\ b1 +s32b2+\ldots\ldots\ldots\ldots+s3kbk = s3y\\
&s41\ b1 +s42b2+\ldots\ldots\ldots\ldots+s4kbk = s4y\\
&s51\ b1 +s52b2+\ldots\ldots\ldots\ldots+s5kbk = s5y\\
&\ldots\ldots\ldots\ldots\\
&sk1\ b1 +sk2b2+\ldots\ldots\ldots\ldots+skkbk = sky
\end{aligned} \quad (2)$$

Here

$$sij = \sum u \sum uXiXj - \sum uXi \sum uXj$$

and

$$siy = \sum u \sum uXiy - \sum uXi \sum uy$$

According to the group of equations, first we can obtain the values of variables b1,b2,…, bk and finally b0 is gained by Equation 3:

$$b0 = \frac{\sum uy}{\sum u} - b1\frac{\sum ux1}{\sum u} - b2\frac{\sum ux2}{\sum u} - \ldots.. - bk\frac{\sum uxk}{\sum u} \quad (3)$$

5.5. Evaluation Techniques Criteria

In order to validate the model with respect to its fitting accuracy, we use the Mean Magnitude of Relative Error (MMRE) and Pred (25%) (Sentas *et al*., 2003). A common criterion for measurement of software effort estimation model performance (Kumar *et al*., 2011), which is calculated for each observation, is the Magnitude of Relative Error (MRE) that the absolute value of the relative error (Alyahya *et al*., 2009; Marza and Seyyedi, 2009) Equation 4:

$$MREi = \frac{\left|Actual\,Effort_i - Predicted\,Effort_i\right|}{Actual\,Efforat_i} \quad (4)$$

We evaluated the impact of estimation accuracy by using (MRE, MMRE) evaluation criteria, for each model.

The Mean Magnitude of Relative Error (MMRE) is the average of all magnitudes of relative errors. Pred (25%) is the percentage of software projects with an MRE of 25% or less (Sentas *et al*., 2003). Therefore, with aggregation of MRE of all data set, the Mean Magnitude of Relative Error (MMRE) is achieved with the equation below Equation 5:

$$MMRE = \frac{1}{n}\sum_{i=1}^{i=n}\frac{\left|E_i - E_t\right|}{E_i} \quad (5)$$

Therefore, we used Pred (25) or more than according to the Equation 6:

$$Pred(l) = \frac{k}{N} \quad (6)$$

To explain parameters k is the number of observations, where MRE is less than or equal to l, for example, Pred (25) gives the percentage of projects which were predicted with a counting the number of MRE less or equal than 0.25 and dividing by the number of projects (Alyahya *et al*., 2009). However, the accuracy of an estimation technique is proportional to Pred (25), Pred (25) and inversely proportional to the MMRE (Marza and Seyyedi, 2009; Martin *et al*., 2005). According to (Stensrud, 2003), MMRE is used for two kinds of assessments (at least). MMRE is to select between competing prediction models and to provide a quantitative measure of the uncertainty of a prediction.

5.6. Importance of Software Risk Factors in Analysis Phase

Table 2 illustrates all respondents indicated that the risk of "developer software gold-plating" was the highest software risk factors and very important. In fact, the software risk factors in the analysis phase from risk number 3, 4, 5, 6, 1, 8, 2 were identified as very important, the software risk factors from risk number 9, 7, 10 in descending means were identified as important.

5.7. Frequency of Occurrence of Controls

Table 3 shows the mean and the standard deviation for risk management techniques. The results of this study show that the risk management techniques are used most of time and often.

Table 2. The mean scores for each risk factor (analysis phase)

Risk	Mean	Std. Deviation	Percent
r13	4.145	0.743	82.895
r14	4.092	0.819	81.842
r15	4.079	0.796	81.579
r16	4.026	0.748	80.526
r11	4.026	0.588	80.526
r18	4.013	0.792	80.263
r12	4.000	0.849	80.000
r19	3.947	0.728	78.947
r17	3.921	0.963	78.421
r20	3.895	0.793	77.895
Total	4.014	0.544	80.289

Table 3. The mean scores for each control factor

Control	Mean	Std. Deviation	Percent
c29	4.408	0.803	88.15789
c30	4.368	0.907	87.36842
c20	4.184	0.668	83.68421
c27	4.171	0.755	83.42105
c21	4.171	0.700	83.42105
c19	4.158	0.612	83.15789
c28	4.158	0.767	83.15789
c25	4.132	0.718	82.63158
c26	4.118	0.653	82.36842
c23	4.105	0.741	82.10526
----	-----	-----	----------
c13	3.868	0.754	77.36842

5.8. Relationships between Risks and Risk Management Techniques

Regression technique was performed on the data to determine whether there were significant relationships between risk management techniques and software risk factors in anlaysis phase. These tests were performed using fuzzy multiple regression analysis and stepwise multiple regression analysis, to compare the risk management techniques to each of the software risk factors to identiiy if they are effective in reducing the occurrence of each software risk factor. Relationships between software risks and controls, which were significant and insignificant, any control is no significant.

5.8.1. Comparison between Estimation Stepwise and Fuzzy Multiple Regression by Evaluation Techniques

Table 4 illustrates an evaluation between stepwise multiple regression and fuzzy multiple regression by using MMRE and Pred (l) that comparing among various software project risk models. Thus, the model's accuracy slightly improves in stepwise multiple regression than fuzzy multiple regression.

Table 4. Comparison between estimation stepwise and fuzzy multiple regression by evaluation techniques

M	Technique	Stepwise multiple regression	Fuzzy multiple regression
R1	MMRE	0.079417544	0.085328352
	Pred (25)	0.894736842	0.894736842
R2	MMRE	0.119657237	0.124959482
	Pred (25)	0.855263158	0.868421053
R3	MMRE	0.106673434	0.120916932
	Pred (25)	0.881578947	0.868421053
R4	MMRE	0.111418233	0.117501692
	Pred (25)	0.894736842	0.855263158
R5	MMRE	0.118881000	0.119694000
	Pred (25)	0.881578947	0.855263158
R6	MMRE	0.103023000	0.101225000
	Pred (25)	0.907895000	0.921053000
R7	MMRE	0.163832640	0.182417840
	Pred (25)	0.842105263	0.776315789
R8	MMRE	0.113644267	0.120718232
	Pred (25)	0.947368421	0.868421050
R9	MMRE	0.1070663220	0.1054268500
	Pred (25)	0.9605263160	0.9210526320
R10	MMRE	0.1297785400	0.1269114020
	Pred (25)	0.8552631580	0.9078947370

Table 5. Software risk factors in the analysis phase mitigated by using risk management techniques

Software risk factors (analysis phase)	Risk management techniques
Unclear, incorrect, continually and rapid changing software project requirements.	C1: Using of requirements scrubbing.
Failure to incomplete or missing detailed requirements analysis and specification documentation.	C3: Assessing cost and scheduling the impact of each change to requirements and specifications, C7: Developing contingency plans to cope with staffing problems.
Developer software gold-plating.	C7: Developing contingency plans to cope with staffing problems, C25: Avoiding having too many new functions on software projects
Lack of IT management.	C6:Implementing and following a communication plan, C3:Assessing cost and scheduling the impact of each change to requirements and specifications
Software project requirements not adequately identified and mismatch.	C1: Using of requirements scrubbing, C21: Including formal and periodic risk assessment, C20: Involving management during the entire software project lifecycle.
Inadequate knowledge about tools and programming techniques.	C1: Using of requirements scrubbing, C7: Developing contingency plans to cope with staffing problems, C16: Implementing/Utilizing automated version control tools.
Lack of traceability, confidentiality, correctness and inspection of the software project planning.	C8: Assigning responsibilities to team members and rotate jobs, C1: Using of requirements scrubbing.
Major requirements change after software project plan phase.	C10: Reviewing and communicating progress to date and setting objectives for the next phase, C7: Developing contingency plans to cope with staffing problems.
Changing software project specifications.	C21: Including formal and periodic risk assessment, Involving management during the entire software project lifecycle, C3: Assessing cost and scheduling the impact of each change to requirements and specifications.
Inadequate value analysis to measure progress.	C23: Educating users on the impact of changes during the software project, C20: Involving management during the entire software project lifecycle.

Also, all models in stepwise and fuzzy acceptable value for MMRE less than 0.25 and Pred(0.25) greater than 0.75 is desirable (Basha and Ponnurangam, 2010). This be explained by the non-deterministic (fuzzy) nature or fuzzy regression. If the problem at hand, involves non-determinstic (fuzzy) variable (fuzzy regression) is recommended which supports the need to use hybrid models in future research as proposed by (Martin *et al*., 2005).

5.8.2. Software Risk Factors Identification Checklists and Risk Management Techniques

Table 5 shows software risk factors identification checklist with risk management techniques based on a questionnaire of experienced software project managers. He can use the checklist on software projects to identify and mitigate software risk factors on lifecycle software projects by using risk management techniques.

6. CONCLUSION

The concern of this study is the managing risks of software projects. The results show that all risks in software projects were very important and important in software project manager's perspective, whereas all controls are used most of time and often. These tests were performed using stepwise multiple regression analysis, fuzzy multiple regression analysis techniques proposed, to compare the controls to each of the software risk factors to identify if they are effective in mitigating the occurrence of each software risk factor. However, we referred the risk management techniques were mitigated on software risk factors in **Table 5**. Through the results, we found out that some control haven't impacted, so the important controls should be considered by the software development companies in Palestinian.

Table 4 illustrates after applying MRE, the results show that the most value of MMRE in fuzzy multiple regression model were slightly higher than or equal stepwise multiple regression except risk 6, 9, 10 models. Therefore, the most value of Pred (25) stepwise multiple regression for software risks were slightly higher than or equal the fuzzy multiple regression model value of Pred (25) except 2, 6, 10. The model's accuracy slightly improves in stepwise multiple regression rather than fuzzy multiple regression. However, all models in stepwise and fuzzy acceptable value for MMRE less than 0.25 and Pred (0.25) greater than 0.75 is desirable.

In addition, We cannot obtain historical data from database by using some techniques. Therefore, some of software project managers didn't give us a historical template to follow up software risks. In addition , there is no software clearly to compute the fuzzy regression analysis and combine between linear and nonlinear technique. Unfortunately, there is no software included mining and statistical techniques to mitigate risks 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 new techniques and approach on a software project. Likewise, we can use other nonlinear techniques to find the relation between software risk and risk management techniques that we believe can contain better result. Also, we could hybridize the techniques with other artificial intelligence approach to improve the models.

7. ACKNOWLEDGEMENT

This study is supported by Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka (UTeM) and Al-Aqsa University, Gaza, Palestine.

8. REFERENCES

Aagedal, J.O., F. den Braber, T. Dimitrakos, B. Gran and D. Raptis *et al*., 2002. Model-based risk assessment to improve enterprise security. Proceedings of the 6th International Enterprise Distributed Object Computing Conference, Sept. 20-20, IEEE Xplore Press, pp: 51-62. DOI: 10.1109/EDOC.2002.1137696

Addison, T., 2003. E-commerce project development risks: Evidence from a Delphi survey. Int. J. Inform. Manage., 23: 25-40. DOI: 10.1016/S0268-4012(02)00066-X

Addison, T. and S. Vallabh, 2002. Controlling software project risks: An empirical study of methods used by experienced project managers. Proceedings of the Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on Enablement through Technology, (TET 02), pp: 128-140.

Alhawari, S., F. Thabtah, L. Karadsheh and W. Hadi, 2008. A risk management model for project execution. Proceedings of the 9th IIBIMA Conference on Information Management in Modern Organizations, (MMO' 08), pp: 887-893.

Aloini, D., R. Dulmin and V. Mininno, 2007. Risk management in ERP project introduction: Review of the literature. Inform. Manage., 44: 547-567. DOI: 10.1016/j.im.2007.05.004

Alyahya, M., R. Ahmad and S. Lee, 2009. Effect of CMMI-based software process maturity on software schedule estimation. Malaysian J. Comput. Sci., 22: 121-137.

Anantatmula, V., 2010. Project planning techniques for academic advising and learning. Int. J. Scholarship Teach. Learn., 6: 1-19.

Aritua, B., N.J. Smith and D. Bower, 2011. What risks are common to or amplified in programmes: Evidence from UK public sector infrastructure schemes. Int. J. Project Manage., 29: 303-312. DOI: 10.1016/j.ijproman.2010.04.004

Azari, A., N. Mousavi and S. Mousavi, 2011. Risk assessment model selection in construction industry. Expert Syst. Applic., 38: 9105-9111. DOI: 10.1016/j.eswa.2010.12.110

Aziz, M. and H. Salleh, 2011. People critical success factors of IT/IS implementation: Malaysian perspectives. World Acad. Sci. Eng. Technol., 80: 75-82.

Basha, S. and D. Ponnurangam, 2010. Analysis of empirical software effort estimation models. Int. J. Comput. Sci. Inform. Security 7: 68-77.

Bavani, R., 2010. Global software engineering: Challenges in customer value creation. Proceedings of the 5th IEEE International Conference on Global Software Engineering, Aug. 23-26, IEEE Xplore Press, Princeton, NJ., pp: 119-122. DOI: 10.1109/ICGSE.2010.21

Begum, Z., M. Khan, M. Hafiz, M.S. Islam and M. Shoyaib, 2008. Software development standard and software engineering practice: A case study of Bangladesh. J. Bangladesh Acad. Sci., 32: 131-139. DOI: 10.3329/jbas.v32i2.2432

Bennatan, E., 2006. Catastrophe Disentanglement: Getting Software Projects Back on Track. 1st Edn., ADDISON WESLEY Publishing Company Incorporated, Upper Saddle River, ISBN-10: 0321336623, pp: 270.

Bhoola, V., S. Hiremath and D. Mallik, 2014. Identifying and evaluation of risks in software projects in Indian IT industries. Int. J. Res. Bus. Technol., 4: 346-353.

Blackmore, K. and K. Nesbitt, 2008. Identifying risks for cross-disciplinary higher degree research students. In Proceedings of the 10th Conference on Australasian Computing Education, Jan. 2-25, Wollongong, Australia, pp: 43-52.

Boehm, B., 1991. Software risk management: Principles and practices. IEEE Software, 8: 32-41. DOI: 10.1109/52.62930

Boehm, B., 2001. Project termination doesn't equal project failure. IEEE Comput., 33: 94-96. DOI: 10.1109/2.868706

Boehm, B., 2002a. Boehm's top 10 2002-software risks. The USC Center for Systems and Software Engineering.

Boehm, B., 2002b. Software risk management: Overview and recent developments. University of Southern California.

Boehm, B., 2003. Value-based software engineering. ACM Software Eng. Notes, 28: 1-12.

Boehm, B., 2007. USC CSSE workshop overview: Top 3 software-intensive systems risk items. SlideServe.

Boehm, B. and V. Basili, 2001. Software defect reduction top 10 list. Computer, 34: 135-137. DOI: 10.1109/2.962984

Boehm, B., B. Clark, E. Horowitz, C. Westland and R. Madachy *et al*., 1995. Cost models for future software life cycle processes: COCOMO 2.0. Ann. Software Eng., 1: 57-94. DOI: 10.1007/BF02249046

Boehm, B. and R. Ross, 1989. Theory-W software project management principles and examples. IEEE Trans. Software Eng., 15: 902-916. DOI: 10.1109/32.29489

Brandon, D.M., 2006. Project management for modern information systems. Electronic Library, 25: 379-380.

Cahill, V., A. Fox, T. Kindberg and B. Noble, 2004. Guest editors' introduction: Building and evaluating ubiquitous system software. IEEE Pervasive Comput., 3: 20-21. DOI: 10.1109/MPRV.2004.1321023

Cantor, M., 1998. Object-Oriented Project Management with UML. 1st Edn., Wiley, New York, ISBN-10: 0471253030, pp: 343.

CHAOS, 1995. The Standish group report. CHAOS.

Chapman, R. and D. Thomas, 2007. A Guide to Printed and Electronic Resources for Developing a Cost-Effective Risk Mitigation Plan for New and Existing Constructed Facilities. 1st Edn., National Institute of Standards and Technology Building and Fire Research Laboratory, pp: 162.

Chen, J. and S. Huang, 2009. An empirical analysis of the impact of software development problem factors on software maintainability. J. Syst. Software, 82: 981-992. DOI: 10.1016/j.jss.2008.12.036

Dash, R. and R. Dash, 2010. Risk assessment techniques for software development. Eur. J. Sci. Res., 42: 629-636.

Dom, R., B. Abidin, S. Kareem, S. Ismail and N. Daud, 2012. Determining the critical success factors of oral cancer susceptibility prediction in Malaysia using fuzzy models. Sains Malaysiana, 41: 633-640.

Dufner, D., O. Kwon and A. Doty, 1999. Improving software development project team performance: A web-based expert support system for project control. Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences, Jan. 5-8, IEEE Xplore Press, Maui, HI, USA, pp: 1-10. DOI: 10.1109/HICSS.1999.772622

Dyba, T. and T. Dingsoyr, 2008. Empirical studies of agile software development: A systematic review. Inform. Software Technol., 50: 833-859. DOI: 10.1016/j.infsof.2008.01.006

Dzung, D. and C. Briod, 2005. Shared authorization in industrial automation systems using threshold cryptography. Proceedings of the 10th IEEE Conference on Emerging Technologies and Factory Automation, Sept. 19-22, IEEE Xplore Press, Catania, pp: 867-876. DOI: 10.1109/ETFA.2005.1612764

Elzamly, A. and B. Hussin, 2011a. Estimating quality-affecting risks in software projects. Int. Manage. Rev., 7: 66-83.

Elzamly, A. and B. Hussin, 2011b. Managing software project risks with proposed regression model techniques and effect size technique. Int. Rev. Comput. Software, 6: 250-263.

Elzamly, A. and B. Hussin, 2013a. Managing software project risks (Implementation Phase) with proposed stepwise regression analysis techniques. Int. J. Inform. Technol., 1: 300-312.

Elzamly, A. and B. Hussin, 2013b. Managing software project risks (Design Phase) with proposed fuzzy regression analysis techniques with fuzzy concepts. Int. Rev. Comput. Software, 8: 2601-2613.

Elzamly, A. and B. Hussin, 2014a. An enhancement of framework software risk management methodology for successful software development. J. Theoretical Applied Inform. Technol., 62: 17-17.

Elzamly, A. and B. Hussin, 2014b. Managing software project risks (Planning Phase) with proposed fuzzy regression analysis techniques with fuzzy concepts. Int. J. Inform. Comput. Sci., 3: 31-40. DOI: 10.14355/ijics.2014.0302.02

Emam, K., 2005. The ROI from Software Quality. 1st Edn., CRC Press, ISBN-10: 1420031201, pp: 296.

Fairley, R., 2009. Managing and Leading Software Projects. 1st Edn., Wiley, Los Alamitos, ISBN-10: 0470294558, pp: 492.

Ferraro, J., 2012. Project Management for Non-Project Managers. 1st Edn., AMACOM Div American Mgmt Assn, New York, ISBN-10: 0814417361, pp: 242.

Galorath, D., 2006. The 10 step software estimation process for successful software planning, measurement and control.

Galorath, D. and M. Evans, 2006. Software Sizing, Estimation and Risk Management: When Performance is Measured Performance Improves. 1st Edn., CRC Press, ISBN-10: 0849335930, pp: 576.

Grabski, S., S. Leech and B. Lu, 2001. Risks and controls in the implementation of ERP systems. Int. J. Digital Account. Res., 1: 47-68. DOI: 10.4192/1577-8517-v1_3

Grady, J.O., 2010. System Requirements Analysis. 1st Edn., Academic Press, Amsterdam, ISBN-10: 0080457851, pp: 480.

Green, R., 2000. Documentation meets version control: An automated backup system for HTML-based help. Proceedings of the 18th Annual ACM International Conference on Computer Documentation: Technology and Teamwork, Sept. 24-27, Cambridge, MA, USA, pp: 541-548.

Greenyer, J., A. Sharifloo, M. Cordy and P. Heymans, 2012. Efficient consistency checking of scenario-based product-line specifications. Proceedings of the 20th IEEE International Requirements Engineering Conference, Sept. 24-28, IEEE Xplore Press, Chicago, IL, pp: 161-170. DOI: 10.1109/RE.2012.6345800

Horine, G., 2009. Absolute Beginner's Guide to Project Management. 2nd Edn., Pearson Education, ISBN-10: 0768687446, pp: 432.

Gu, X., G. Song and L. Xiao, 2006. Design of a fuzzy decision-making model and its application to software functional size measurement. Proceedings of the International Conference on Computational Intelligence for Modelling Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, Nov. 28-Dec. 1, IEEE Xplore Press, Sydney, NSW, pp: 199-205. DOI: 10.1109/CIMCA.2006.82

Han, W. and S. Huang, 2007. An empirical analysis of risk components and performance on software projects. J. Syst. Software, 80: 42-50. DOI: 10.1016/j.jss.2006.04.030

Hayat, F., M. Bhatt, N. Ehsan, A. Ishaque and S. Ahmed *et al*., 2010. A methodology to manage the changing requirements of a software project. Proceedings of the International Conference on Computer Information Systems and Industrial Management Applications, Oct. 8-10, IEEE Xplore Press, Krackow, pp: 319-322. DOI: 10.1109/CISIM.2010.5643642

Hill, P., 2010. Practical Software Project Estimation: A Toolkit for Estimating Software Development Effort and Duration. 1st Edn., McGraw Hill Professional, New York, ISBN-10: 0071717927, pp: 312.

Hoffer, J., J. George and J. Valacich, 2011. Modern Systems Analysis and Design. 6th Edn., Pearson Education India, ISBN-10: 8131726258, pp: 575.

Holcombe, M., 2008. Running an Agile Software Development Project 1st Edn., John Wiley and Sons, ISBN-10: 0470385871, pp: 350.

Huang, S. and W. Han, 2008. Exploring the relationship between software project duration and risk exposure: A cluster analysis. Inform. Manage., 45: 175-182. DOI: 10.1016/j.im.2008.02.001

IEEE, 2011. IEEE Guide-Adoption of the Project Management Institute (PMI ®) Standard A Guide to the Project Management Body of Knowledge (PMBOK ® Guide) 4th Edn., The Institute of Electrical and Electronics Engineers, Inc., pp: 508.

Islam, S., 2009. Software development risk management model: A goal driven approach. Proceedings of the Doctoral Symposium for ESEC/FSE on Doctoral Symposium, Aug. 24-28, Amsterdam, Netherlands, pp: 5-8. DOI: 10.1145/1595782.1595785

Jalote, P., 2002. Software Project Management in Practice 1st Edn., Addison-Wesley Professional, Boston, ISBN-10: 0201737213, pp: 262.

Jiang, G. and Y. Chen, 2004. Coordinate metrics and process model to manage software project risk. Proceedings of the IEEE International Engineering Management Conference, Oct. 21-21, IEEE Xplore Press, Singapore, pp: 865-869. DOI: 10.1109/IEMC.2004.1407505

Jianga, J., G. Kleinb, H.G. Chenc and L. Lind, 2002. Reducing user-related risks during and prior to system development. Int. J. Project Manage., 20: 507-515. DOI: 10.1016/S0263-7863(01)00049-7

Jin, J. and B. Li, 2009. Cooperative multicast scheduling with random network coding in WiMAX. Proceedings of the 17th International Workshop on Quality of Service, Jul. 13-15, IEEE Xplore Press, Charleston, SC, pp: 1-9. DOI: 10.1109/IWQoS.2009.5201384

Jones, C., 2008. Applied Software Measurement: Global Analysis of Productivity and Quality. 3rd Edn., McGraw Hill Professional, New York, ISBN-10: 0071643869, pp: 662.

Jones, C., 2009. Software Engineering Best Practices: Lessons from Successful Projects in the Top Companies, 1st Edn., McGraw Hill Professional, New York, ISBN-10: 0071621628, pp: 688.

Kamaruddin, A., 2006. Development of an early software project risk assessment application using case-based reasoning. Universiti Teknologi MARA.

Kandt, R.K., 2003. Software Requirements Engineering: Practices and Techniques. JPL.

Kanjanasanpetch, P. and B. Igel, 2003. Managing knowledge in Enterprise Resource Planning (ERP) implementation. Proceedings of the Managing Technologically Driven Organizations: The Human Side of Innovation and Change Engineering Management Conference, Nov. 2-4, IEEE Xplore Press, pp: 30-35. DOI: 10.1109/IEMC.2003.1252226

Kasurinen, J., O. Taipale and K. Smolander, 2010. Test case selection and prioritization: Risk-based or design-based? Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement Sept. 16-17, Bolzano, Italy, pp: 1-10. DOI: 10.1145/1852786.1852800

Keil, M., P. Cule, K. Lyytinen and R. Schmidt, 1998. A framework for identifying software project risks. Commun. ACM, 41: 76-83. DOI: 10.1145/287831.287843

Keil, M., L. Li, L. Mathiassen and G. Zheng, 2008. The influence of checklists and roles on software practitioner risk perception and decision-making. IEEE Comput. Society, 81: 908-919.

Keil, M., A. Tiwana and A. Bush, 2002. Reconciling user and project manager perceptions of IT project risk: A Delphi study. Inform. Syst. J., 12: 103-119.

Khanfar, K., A. Elzamly, W. Al-Ahmad, E. El-Qawasmeh and K. Alsamara *et al*., 2008. Managing software project risks with the chi-square technique. Int. Manage. Rev., 4: 18-29.

Kontio, J., 2001. Software engineering risk management: A method, improvement framework and empirical evaluation. Computer. Suomen Laatukeskus/the Center for Excellence Purotie.

Kouskouras, K. and A. Georgiou, 2007. A discrete event simulation model in the case of managing a software project. Eur. J. Operat. Res., 181: 374-389. DOI: 10.1016/j.ejor.2006.05.031

Kumar, J., A. Mandala, M. Chaitanya and G. Prasad, 2011. Fuzzy logic for software effort estimation using polynomial regression as firing interval. Int. J. Comput. Technol. Applic., 2: 1843-1847.

Kumar, R., 2002. Managing risks in IT projects: An options perspective. Inform. Manage., 40: 63-74. DOI: 10.1016/S0378-7206(01)00133-1

Ewusi-Mensah, K., 2003. Software Development Failures: Anatomy of Abandoned Projects. 1st Edn., MIT Press, Cambridge, ISBN-10: 0262050722, pp: 276.

Lan, Y. and S. Guo, 2008. Multiple stepwise regression analysis on knowledge evaluation. Proceedings of the International Conference on Management of E-Commerce and E-Government, Oct. 17-19, IEEE Xplore Press, Jiangxi, pp: 297-302. DOI: 10.1109/ICMECG.2008.65

Lee, S., 2011. Probabilistic risk assessment for security requirements: A preliminary study. Proceedings of the 5th International Conference on Secure Software Integration and Reliability Improvement, Jun. 27-29, IEEE Xplore Press, Jeju Island, pp: 11-20. DOI: 10.1109/SSIRI.2011.12

Li, M., S. Yu, N. Cao and W. Lou, 2013. Privacy-preserving distributed profile matching in proximity-based mobile social networks. IEEE Trans. Wireless Commun., 12: 2024-2033. DOI: 10.1109/TWC.2013.032513.120149

Lientz, B.P. and L. Larssen, 2006. Risk Management for IT Projects: How to Deal with Over 150 Issues and Risks. 1st Edn., Routledge, ISBN-10: 0750682310, pp: 331.

Lin, J., Q. Zhuang and C. Huang, 2012. Fuzzy statistical analysis of multiple regression with crisp and fuzzy covariates and applications in analyzing economic data of china. Comput. Econom., 39: 29-49. DOI: 10.1007/s10614-010-9223-1

Linberg, K.R., 1999. Software developer perceptions about software project failure: A case study. J. Syst. Software, 49: 177-192. DOI: 10.1016/S0164-1212(99)00094-1

Liu, D., Q. Wang and J. Xiao, 2009. The role of software process simulation modeling in software risk management: A systematic review. Proceedings of the 3rd International Symposium on Empirical Software Engineering and Measurement, Oct. 15-16, IEEE Xplore Press, Lake Buena Vista, FL, pp: 302-311. DOI: 10.1109/ESEM.2009.5315982

Lyons, T. and M. Skitmore, 2004. Project risk management in the Queensland engineering construction industry: A survey. Int. J. Project Manage., 22: 51-61. DOI: 10.1016/S0263-7863(03)00005-X

Ma, W., L. Liu, W. Feng, Y. Shan and F. Peng, 2009. Analyzing project risks within a cultural and organizational setting. Proceedings of the ICSE Workshop on Leadership and Management in Software Architecture, IEEE Computer Society Washington, DC, USA, pp: 6-14. DOI: 10.1109/LMSA.2009.5074858

Maglyas, A., 2009. Evaluating the success of software development projects in Russia, Ukraine and Belarus. Lappeenranta University of Technology.

Martin, C.L., J.L. Pasquier, C.M. Yanez and A.G. Tornes, 2005. Software development effort estimation using fuzzy logic: A case study. Proceedings of the 6th Mexican International Conference on Computer Science, Sept. 26-30, IEEE Xplore Press, pp: 113-120. DOI: 10.1109/ENC.2005.47

Marza, V. and M. Seyyedi, 2009. Fuzzy multiple regression model for estimating software development time. Int. J. Eng. Bus. Manage., 1: 31-34. DOI: 10.5772/6775

Masticola, S., 2007. A simple estimate of the cost of software project failures and the breakeven effectiveness of project risk management. Proceedings of the 1st International Workshop on the Economics of Software and Computation, May 20-26, IEEE Xplore Press, Minneapolis, MN., pp: 6-6. DOI: 10.1109/ESC.2007.1

McConnell, S., 1996. Rapid Development: Taming Wild Software Schedules. 1st Edn., Microsoft Press, Redmond, ISBN-10: 1556159005, pp: 647.

Miler, J., 2005. A method of software project risk identification and analysis. Technology. Gdansk University of Technology.

Miller, S., 2004. Best practices for software projects: Requirements scrubbing. TechWell Corp.

Miotto, O., T. Tan and V. Brusic, 2005. Supporting the curation of biological databases with reusable text mining. Genome Inform., 16: 32-44. PMID: 16901087

Munch, J. and J. Heidrich, 2004. Software project control centers: Concepts and approaches. J. Syst. Software, 70: 3-19. DOI: 10.1016/S0164-1212(02)00138-3

Na, K., J. Simpson, X. Li, T. Singh and K. Kim, 2007. Software development risk and project performance measurement: Evidence in Korea. J. Syst. Software, 80: 596-605. DOI: 10.1016/j.jss.2006.06.018

Nakatsu, R. and C. Iacovou, 2009. A comparative study of important risk factors involved in offshore and domestic outsourcing of software development projects: A two-panel Delphi study. Inform. Manage., 46: 57-68. DOI: 10.1016/j.im.2008.11.005

Oda, M., 2011. The characteristics of the use of twitter by beginners: Study of the applicability to the e-healthcare. Proceedings of the International Conference on Systems, Man and Cybernetics, Oct. 9-12, IEEE Xplore Press, Anchorage, AK., pp: 1268-1273. DOI: 10.1109/ICSMC.2011.6083834

Oracle, 2010. A standardized approach to risk improves project outcomes and profitability. Oracle White Paper.

Pandian, C.R., 2006. Applied Software Risk Management: A Guide for Software Project Managers. 1st Edn., CRC Press, ISBN-10: 0849305314, pp: 264.

Peppen, A. and M. Ploeg, 2000. Practising what we teach: Quality management of systems-engineering education. IEEE Trans. Syst. Man Cybernet., 30: 189-196. DOI: 10.1109/5326.868440

Persohn, K. and D. Brylow, 2011. Interactive real-time embedded systems education infused with applied internet telephony. Proceedings of the 35th IEEE Annual Computer Software and Applications Conference, Jul. 18-22, IEEE Xplore Press, Munich, pp: 199-204. DOI: 10.1109/COMPSAC.2011.33

Popescu, C. and M. Giuclea, 2007. A model of multiple linear regression. Publish. House Romanian Acad., 8: 1-8.

Puntambekar, A.A., 2009. Software Engineering. 1st Edn., Technical Publications, ISBN-10: 8184311834, pp: 202.

RM: GBP, 2003. Project Management Committee. Risk Management: Guidelines and Best Practices, pp: 19.

Rodriguez-Repisoa, L., R. Setchib and J. Salmeronc, 2007. Modelling IT projects success: Emerging methodologies reviewed. Technovation, 27: 582-594. DOI: 10.1016/j.technovation.2006.12.006

Ropponen, J. and K. Lyytinen, 2000. Components of software development risk: How to address them? A project manager survey. IEEE Trans. Software Eng., 26: 98-112. DOI: 10.1109/32.841112

Rudd, C., 2010. ITIL V3 Planning to Implement Service Management. 1st Edn., The Stationery Office, London, ISBN-10: 0113311095, pp: 320.

Sarfraz, F., 2009. Managing for a successful project closure. Proceedings of the Portland International Conference on Management of Engineering and Technology, Aug. 2-6, IEEE Xplore Press, Portland, OR, pp: 1392-1395. DOI: 10.1109/PICMET.2009.5262001

Savolainen, P., J. Ahonen and I. Richardson, 2012. Software development project success and failure from the supplier's perspective: A systematic literature review. Int. J. Project Manage., 30: 458-469. DOI: 0.1016/j.ijproman.2011.07.002

Schmidt, R., K. Lyytinen, M. Keil and P. Cule, 2001. Identifying software project risks: An international Delphi study. J. Manage. Inform. Syst., 17: 5-36.

Schulmeyer, G., 2008. Handbook of Software Quality Assurance. 4th., Edn., Artech House, Incorporated, Boston, ISBN-10: 1596931868, pp: 464.

Schwalbe, K., 2010. Information Technology Project Management, Revised. 6th., Edn., Cengage Learning, Boston, MA., ISBN-10: 1111221758, pp: 704.

Selby, R.W., 2007. Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research. 1st Edn., John Wiley and Sons, Hoboken, N.J., ISBN-10: 047014873X, pp: 818.

Sentas, P., L. Angelis and I. Stamelos, 2003. Multinomial logistic regression applied on software productivity prediction. Proceedings of the 9th Panhellenic Conference in Informatics, (PCI' 03), pp: 1-12.

Shand, R.M., 1994. User manuals as project management tools. II. Practical applications. IEEE Trans. Profess. Commun., 31: 123-142. DOI: 10.1109/47.317478

Smith, N., T. Merna and P. Jobling, 2006. Managing Risk in Construction Projects. 2nd Edn., Blackwell Publishing, pp: 244.

Sodhi, J. and P. Sodhi, 2001. IT Project Management Handbook. 1st Edn., Management Concepts, ISBN-10: 1567260985, pp: 350.

SPM: MT, 2004. SE Project 2003/2004 group E. Software project management: methodologies and techniques.

Strawbridge, C., 2005. Project management in large collaborations: SNS lessons learned for ITER. Proceedings of the 21st IEEE/NPS Symposium on Fusion Engineering, (SFE' 05), IEEE Xplore Press, Knoxville, TN., pp: 1-5. DOI: 10.1109/FUSION.2005.252858

Sumner, M., 2000. Risk factors in enterprise-wide/ERP projects. J. Inform. Technol., 15: 317-327. DOI: 10.1080/02683960010009079

Surie, D., 2004. Evaluation and integration of risk management in CMMI and ISO/IEC 15504. Proceedings of the 8th Student Conference in Computing Science, (CCS' 04), Umeå, Sweden, pp: 14-14.

Stensrud, E., 2003. A further empirical investigation of the relationship between MRE and project size.

Taylor, J., 2004. Managing Information Technology Projects: Applying Project Management Strategies to Software, Hardware and Integration Initiatives, 1st Edn., American Management Association, New York, ISBN-10: 0814408117, pp: 274.

Tayntor, C.B., 2006. Successful Packaged Software Implementation. 1st Edn., CRC Press, ISBN-10: 0849334101, pp: 336.

Tomczyk, C., 2005. Project Manager's Spotlight on Planning. 1st Edn., Wiley, San Francisco, ISBN-10: 0782144136, pp: 208.

Tsui, F.F., 2004. Managing Software Projects. 1st Edn., Jones and Bartlett Learning, Sudbury, ISBN-10: 0763725463, pp: 337.

Veracode, 2008. IT risk management guide to software risk assessments and audits. Information Systems Veracode.

Verner, J., K. Cox and S.J. Bleistein, 2006. Predicting good requirements for in-house development projects. Proceedings of the ACM/IEEE International Symposium on Empirical Software Engineering, Sept. 21-22, Rio de Janeiro, Brazil, pp: 154-163. DOI: 10.1145/1159733.1159758

Wallace, L. and M. Keil, 2004. Software project risks and their effect on outcomes. Commun. ACM, 47: 68-73. DOI: 10.1145/975817.975819

Webern, P., G. Medina-Oliva, C. Simon and B. Iung, 2010. Overview on Bayesian networks applications for dependability, risk analysis and maintenance areas. Eng. Applic. Artificial Intell., 42: 115-125. DOI: 10.1016/j.engappai.2010.06.002

Westfall, L., 2001. Software Risk Management. 1st Edn., Westfall Team, pp: 8.

Westfall, L., 2006. Software Requirements Engineering: What, Why, Who, When and How. 1st Edn., pp: 14.

Yassin, A., 2010. Organizational Information Markets: Conceptual Foundation and an Approach for Software Project Risk Management. University of South Florida.

Zhou, N., J. Pierre and D. Trudnowski, 2012. A stepwise regression method for estimating dominant electromechanical modes. IEEE Trans. Power Syst., 27: 1051-1059. DOI: 10.1109/TPWRS.2011.2172004