IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON SOFTWARE STANDARDS AND THEIR IMPACT IN REDUCING SOFTWARE FAILURES

Received December 14, 2017, accepted February 4, 2018, date of publication February 14, 2018, date of current version March 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2805862

A Critical Analysis of Software Risk Management Techniques in Large Scale Systems

MARUF PASHA[1], GHAZIA QAISER[1], AND UROOJ PASHA[2]
[1]Department of Information Technology, Bahauddin Zakariya University, Multan 60000, Pakistan
[2]Institute of Management Sciences, Bahauddin Zakariya University, Multan 60000, Pakistan

Corresponding author: Maruf Pasha (maruf.pasha@bzu.edu.pk)

ABSTRACT Researchers in the software industry have focused on risk management systems for a long time. Software risk management is a software engineering practice that contains risk identification, risk estimation, mitigation, and monitoring. It delivers a disciplined environment for efficient decision-making to assess the problems in software development. Measuring risks in a large-scale system is comparatively difficult because of its complex nature. Large-scale systems are challenging since many risks can arise during system development. Risk factors in large-scale systems are relatively different from small systems, especially with respect to the independent components. This paper describes a difference between large-scale and small-scale systems along with an exhaustive list of risk factors. The tools from the literature are further divided into sub categories according to those that are best suited. We present a detailed comparative analysis for different software-related risk management models with some commonly identified features and further categorize them into classes based on the severity of their risks.

INDEX TERMS Risk management, large scale system, risk factors, risk management tools.

I. INTRODUCTION

Risk is a problem that can cause extensive losses and threats for various organizational procedures. In Computer Science-related fields, risks can come from networks, the Internet, malicious codes or users, loopholes and from physical security [1]. There can be many risks in creating high quality software systems. These cannot be completely eliminated but project managers can reduce these risks and their impact on products by calculating these risks on IT resources [2]. In the current era, software systems have become more complex and are known to be large-scale systems. Increases in project size and complexity result in increases of various risks. The software industry is one of the largest industries in the world. According to [1], every year an average of $350 billion of off-the-shelf software are sold. For large-scale systems, risk management is an investment that can be valuable in the future because of high quality and reliability demands [3]. The main goal for risk management system is to identify and control all possible risks before they occur during software development. Typically, risk can be classified as systematic risks and un-systematic risks [1]. Systematic risks involve the risks caused by external factors, including hacking, viruses, natural disasters and power loss. An example systematic risk is a vulnerable browser, in which any kind of loophole may lead to security breaches and can harm the resources of that organization.

Additionally, un-systematic risks cause unique risks for the firm, including the misuse of confidential data, application error, inside attacks, data loss, equipment malfunctions and human interactions.

Moreover, systematic risks are known as generic risks and unsystematic risks are known as specific risks. Figure 1 describes the classification of risks in software development. In software development, a risk management team continually assesses and monitors various risks and determines their negative impacts on software development. Risk management systems provide a dynamic way for decision-making by prioritizing and ranking the risks. Normally, a risk management system is based on the identification and assessment of risks [1], [4]. Like many other businesses, software development risk cannot be eliminated, but risk managers can reduce the impact of these risks by using appropriate risk management tools and techniques.

A. RISK MANAGEMENT TOOLS

The risk management process requires tools that can identify and be applied to perceived risks. Through analyzing the existing literature of risk management in software

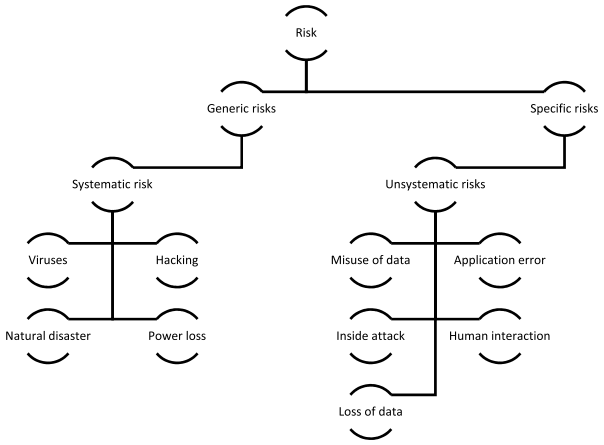

FIGURE 1. Types of risks in software development.

development, we have listed and categorized some risk management tools that are considered to be helpful for mitigating risks [3]. Table 1 presents a summarized view of the tools that are applicable in software development process for risk mitigation.

II. BACKGROUND

The risk management process was introduced in software development as an explicit process in 1980 [21]. Barry Boehm is known as the father of risk management in software engineering. He proposed the risk-driven spiral model and highlighted the concepts of managing risks in software development. In recent trends, an increasing demand for software projects can cause more risks. With more advancements in programming languages, some software issues become more complex and, in those projects, it is difficult to manage risks. As a result, intense research is focused on this risk management process. The research aims to improve software management techniques and advance this field [5].

The top 10 software risks from different studies and their risk factors in the existing literature were reviewed and compiled [6]. The author's results show that planning and controlling are the areas in which some risks exist in software development processes. Reference [1] presents the increasing role of risk management in software development to support and avoid risks. Authors describe a data model for risk identification and risk assessment processes. Reference [7] explores the RM (Risk Management) and KM (Knowledge Management) fields. The authors propose a conceptual framework, KBRM (Knowledge Based Risk Management), that employs KM with RM for improving the effectiveness in IT projects. This study finds some necessary elements for the KBRM framework and recommends some tools for improving the risk management process. Reference [8] presents a risk management process designed for only small businesses. This study is based on extensive research of different small business software development. This research provides a list of risk management tools and recommends some techniques for risk mitigations. Reference [9] describes the relationship between project success and risk management. This method also combines the soft and hard aspects of risk management systems with project success. This work involved interviews with risk and project managers to analyze the hard and soft aspects of risk management systems for project success. Reference [10] presents experiences from different real life industrial cases for measuring risks through qualitative or quantitative data. This research describes the dimensions and causes of the risks, and practically applies the theoretical approaches to industries with project managers. Reference [3] describes a list of techniques that were reviewed from existing literature and proposed some guidelines for using these techniques in real scenarios. This study helps risk management and knowledge management to be integrated and work more efficiently. Reference [11] presents a software application framework for RRA (Rapid Risk Assessment) in integrated supply chain risk management. The proposed method combines qualitative and quantitative data to access and prioritize risks. Qualitative data is based on surveys and quantitative data is based on fuzzy logic and probability theory. Reference [12] presents a study on the risk management system and reviews some tools for the risk mitigation. These studies also present a review of risk management models and describe the techniques for choosing the best technique among them for risk measurement.

A. SOFTWARE RISK MANAGEMENT IN LARGE SCALE SYSTEMS

The risks associated with software development can come from many sources such as human error, software failure, hardware failure or organizational error [13]. In general, there are many techniques and methods for risk management in software development. Software development projects have an increased rate of failure and, like other businesses, software development involves technical and expensive resources. Moreover, if the size and complexity EW increases, risk mitigation in software development becomes more problematic [4].

A major problem with large-scale system development is its unmanageability and uncontrollability. According to [14], the main issues are not having the proper requirements and frequent changes in the requirements. Large-scale systems are made from networks of humans and technical resources that include machines, computers and software. The differences between large-scale and small-scale systems are the interfaces, interdependencies and attention paid to the elements. When it comes to risk mitigation in large-scale systems, researchers have focused on its interdependence, risk migration and long incubation periods. In general, large-scale systems are more difficult to handle since they are more complex. Therefore, the large system is divided into subcomponents and subfactors in order to perform well. By dividing the system into smaller groups, risk mitigation is possible [15].

TABLE 1. Tools for risk management.

Category	Tools	Reference
Risk identification	Brainstorming	[3] [22]
	Checklists	[3] [22]
	Periodic risk reports	[22] [23]
	Risk documentations	[22] [23] [4]
Risk analysis		
	Probability of risk assessment	[22] [3]
	Impact of risk assessment	[22] [3]
	Time frame for risk assessment	[22]
	Risk categorization and classification	[22] [3]
	Risk prioritization	[22] [3]
	Graphical representation of risks	[22] [3]
Risk planning		
	Responsibility assignments	[4] [22] [23]
	Team planning for risk mitigation	[22] [23] [4]
	Time limited actions	[22] [23]
	Cost benefit assessment during planning	[22] [23] [4]
	Causes and effects analysis of risk	[24] [22] [23] [3] [4]
	Human reliability assessment	[3] [22]
	Technical tools analysis	[23] [22] [3] [4]
	Project re-planning for risks management	[22] [4]
	Communication plans with users and stakeholders	[4] [23] [22]
	Reusable code	[4] [22] [23]
Risk tracking		
	Revise risk assessment	[22] [4]
	Periodic reviews of documents	[22] [23] [4]
	Periodic reports of risk status	[22] [23]
	Periodic risk mitigation reports and plans	[4] [22] [23]
	Periodic reporting of recent trends	[22] [23]
	Critical reporting of risks to senior management	[22] [24] [23]
	Analysis of recent trends and expectations	[22] [24] [23] [3] [4]
	Project re-planning	[4] [22] [23]

TABLE 1. *(Continued.) Tools for risk management.*

Risk control	Process of closing risks	[22]
	Possible plans for risks management failure	[22]
	Cost-benefit analysis during risk control	[22] [23] [3] [4]
	Cause-and-effect analysis during risk control	[22] [3] [23]
	Reusable test plans	[23] [4]
	Design recovery plan	[21]
Risk monitoring	Prototyping	[22] [23] [4]
	Benchmarking	[22] [21]
	Simulations	[22] [21]
	Requirement management	[22] [4] [23]
	Contractor and sub-contractor management	[22] [23]
	Configuration control	[22] [23]
	Training programs	[22] [24] [4]
	Quality control	[22] [23] [4]
	Quality management	[22] [23] [4]
	Customer satisfaction feedback	[22] [21]
	Team building sessions	[22] [23] [24] [4]
	Training of users with new technology	[22] [23] [24] [4]

B. RISK ASSESSMENT FRAMEWORK

Large-scale systems are complex. They require careful processing, planning and execution for a successful final product. Risk in software development is a crucial issue, and software development suffers from lack of proper risk mitigation [16]. In figure 2, a risk assessment framework is described. Risk mitigation starts with the risk identification process in which risk managers identify all possible risk factors after the identification process. Then, risks will be categorized and prioritized. Managers and senior management calculate the impacts of the risks on software development. With the help of this risk estimation, more influential risks will be mitigated first. This process is beneficial to minimize large organizational losses. After that, the high impact risks will be analyzed. Risk managers will plan for the mitigation of risk and apply techniques to counteract it. After enacting these countermeasures, risk managers monitor and report the process. If they find more loopholes, this process will start again with the risk identification process. In the phase of monitoring and reporting, the team members should control for and track the risk. In the final step, the risk feedback is evaluated by the stakeholders of the software project. They update and share their feedback to the team members and will track this process.

C. RISK FACTORS IN SOFTWARE DEVELOPMENT

This study describes an exhaustive list of risk factors that are identified from the existing literature. After identifying these factors, they are categorized according to some different phases, such as the user level phase, requirement gathering, planning, analysis, design, implementation and maintenance. Table 2 illustrates the risk factors that can be in small-scale or large-scale systems.

D. SMALL-SCALE VS LARGE-SCALE SOFTWARE

There can be small, medium or large enterprises of software development that can be affected by the risks. Identifying the problem and its source is the first step for the risk assessment [17]. For large-scale systems, risk assessment is itself a challenge [18], because, according to [32], large-scale

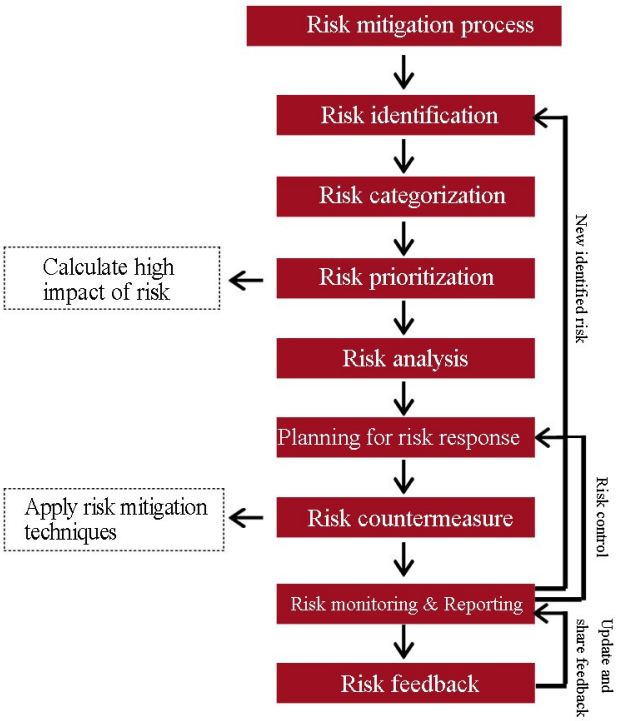

FIGURE 2. Risk assessment framework.

FIGURE 3. Impact of risk.

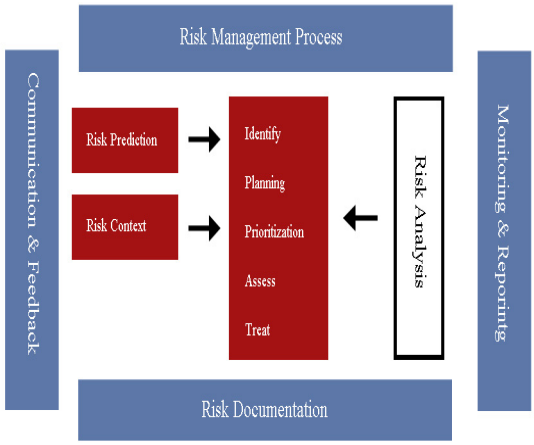

FIGURE 4. Risk management process.

software systems are like a puzzle. Risk management in large-scale systems is difficult since components in large-scale systems are more complex and independent [19]. Usually, many software projects deliver the proper functionality and the performance that was promised by the developers, but some software projects fail to fulfill the requirements of users in given time and budget [20]. Risk can occur in any project and can harm the final product and its functionality. For reducing or avoiding risks, risk managers should take appropriate countermeasures. If there are no proper risk management techniques, the failure of a large-scale system can affect the stakeholders of that product. Failure of the final product can waste the budget and time of its customers, employees and organization. The failure can reduce the profits of the business. Figure 3 describes who will be affected when risk occurs.

III. RISK MANAGEMENT PROCESS

Risk management processes have always been a crucial part of software development. For complex and large-scale systems, it is beneficial to implement a risk management process to reduce the possible risks [21]. Project managers can usually predict and identify risks, but this is not enough. According to [21], in large-scale systems, a formal risk management process is necessary for the effectiveness of the project. For developing a risk management process in a systematic way, it is crucial to have standardized and well-developed methods for each step. Figure 4 describes a risk management process. The risk management process starts with the risk prediction. The risk managers and project managers define the context of the risk. In this step, risk management tools like checklists, brainstorming and requirement analysis can be used for predicting and perceiving the risks in a system. After the identification of risks, the mitigation planning will start. Managers calculate, estimate, and prioritize the impacts of risks. Risk prioritization is a crucial step that addresses the sequence of risks in the mitigation process. Risk analysis is the step that will keep analyzing the whole of these processes in order to review the risks. After this process, with the proper planning, the risk will be assessed and treated. Risk monitoring and reporting will keep track of identified risks and its measuring process. It helps risk managers avoid future risks in their projects. Communication and feedback is useful for all stakeholders of the project, and sharing their feedback will help project managers avoid risks and make an effective final product. In the risk documentation step, the whole process will be written in a document form to help the organization.

TABLE 2. Risk factors.

Phase	No	Risk factors	Ranking for large scale	Ranking for small scale
User level [24] [25] [26] [21] [20] [27]	1	Irresponsible user	**Medium**	**Low**
	2	Low involvement of user	**Low**	**Low**
	3	Not ready to accept new technology	**Low**	**High**
	4	Unable to convey his idea to designer	**Medium**	**Low**
	5	Conflicts among different users	**High**	**Low**
Requirement gathering [17] [25] [24] [26] [6] **[21] [20] [16]**	1	Misunderstanding in requirement gathering	**High**	**Medium**
	2	Requirements are not deliverable	**High**	**Low**
	3	Incomplete requirements	**High**	**Low**
Planning [26] [25] [23] [24] [6] **[28] [21] [20] [16]**				
	1	Unrealistic schedule and budget	**Medium**	**Low**
	2	Unrealistic goals	**Medium**	**Low**
	3	Inappropriate staff	**Low**	**Medium**
	4	Lack of committed leadership	**Low**	**High**
	5	Poor planning	**High**	**Low**
	6	Lack of appropriate software project management methodology	**Medium**	**Low**
	7	Change in management during development of project	**High**	**Medium**
	8	Less communication between the team of software project management team	**Medium**	**Low**
	9	Absence of historical data	**High**	**Low**
Analysis [25] [23] [24] [26] [6] **[21] [20] [27]**	1	Unclear, wrong and rapid change in requirements	**High**	**Low**
	2	Missing complete requirement analysis	**Medium**	**High**
	3	Gold plating	**High**	**Medium**
	4	Absence of skills in IT manager	**Low**	**Low**
	5	Requirement are not identified	**Medium**	**Low**
	6	Lack of knowledge about programming tools and technologies	**Low**	**High**
	7	Lack of confidentiality and accuracy of planned software	**High**	**Medium**
	8	Major change in requirements after planning phase	Medium	Low
	9	Change in the specification in software project	Low	Low
	10	Inappropriate analysis of values for measuring	High	Medium

TABLE 2. *(Continued.)* **Risk factors.**

Design [25] [23] [26] [24] [6] [20] [28] [16] [27]	1	New technology about domain	**High**	**Low**
	2	Developing wrong software functions	**Medium**	**High**
	3	Developing wrong UI	**Medium**	**High**
	4	Insufficient functions to ensure the security of database	**High**	**High**
	5	Lack of integrity in design	**Medium**	**Medium**
	6	Lack of quality software project	**High**	**Medium**
	7	Absence of quality architectural design and documents	**High**	**Low**
	8	Fail to design user requirements	**Low**	**Low**
	9	Lack of integration between team of software project development e.g., stakeholders	**High**	**Low**
	10	Improper alignment of software project with local practices	**Low**	**Medium**
	11	Developer has no knowledge of business	**Low**	**Low**
Implementation [25] [23] [24] [26] [6] [21] [20] [27]				
	1	Personal shortfalls	**Low**	**High**
	2	Fails to apply phased delivery approach	**Medium**	**Low**
	3	Lack of proper attention in developing and implementation	**High**	**Low**
	4	Inappropriate team members	**Medium**	**Low**
	5	Inappropriate code comments	**Low**	**Low**
	6	Inappropriate test case	**High**	**Medium**
	7	Shortfall in real time performance	**High**	**Medium**
	8	Design of test case and unit level testing turns out difficult	**Medium**	**Low**
	9	Lack of programming standards	**Low**	**Low**
	10	Security	High	Medium
Maintenance [25] [23] [9] [29] [24] [26] [21] [20] [28] [16] [27]	1	Lack of skills	Low	Medium
	2	Improper change management	High	Medium
	3	Politics with negative impact on software project	High	Low
	4	Shortage of IT resources and proper facilities	Medium	Medium
	5	Loss of technical resource	Low	Low
	6	Lack of commitment and involvement of top management	Medium	Low
	7	Shortfalls in external components	High	Low
	8	Legacy software project	Low	Medium
	9	Mismatch in achievement and contracting process	**Low**	**Medium**
	10	Incomplete user documents	**High**	**Medium**
	11	Harmful actions	**High**	**Low**

TABLE 2. *(Continued.)* **Risk factors.**

	12	Demographically not accepted	**Low**	**High**
	13	New product in market (competition)	**High**	**Medium**
	14	Project value	**High**	**Low to Medium**
	15	Complex for users/Not user friendly	**High**	**Medium**
	16	Absence of historical data	**Medium**	**Low**
	17	Harmful competitive action	**High**	**Low**

TABLE 3. **A comparative study.**

Model / Technique	Risk Identification	Risk Categorization	Decision Making Tool	SEI Standard	Team Involvement	Continuous Risk Management	Risk Automation	Flexible	Risk Estimation	Risk Documentation	Systematic Risk Analysis	Risk tracking And Monitoring	Qualitative Analysis	Quantitative Risk Analysis	Identify sources Of risks	Enhanced UI	Risk Repository
SRE [12][30][37] Rabbi et al, Chawan et al, Carr et al,	✔	✔	✔	✔	✔						✔	✔					
TRM [12] [30][38] Rabbi et al, Chawan et al, Higuera et al	✔			✔	✔	✔						✔					
Softrisk model [12][36] Rabbi et al, Keshlaf et al,		✔				✔	✔	✔	✔	✔		✔					
Riskit framework [12] [21] Rabbi et al, Wallmüller et al,	✔	✔			✔	✔				✔	✔	✔	✔			✔	
Project risk Management process [30] Chawan et al,	✔	✔							✔			✔					
Risk management process [30][35] Chawan et al, Kontio et al.,	✔	✔							✔			✔	✔				
CMM based Model [12] [31] [34] Rabbi et al, Kaur et al, Ruzhi et al,	✔	✔		✔		✔			✔			✔					✔
ARMOR [12] [33] Rabbi et al, Lu et al,	✔	✔	✔						✔					✔	✔	✔	✔

IV. MODEL COMPARISON

In this section, the present authors extract some features from the literature and those features are used to analyze the risk management model and framework. This analysis is based on extracted common features among risk management models and frameworks. The comparative analysis can help others select the best model or technique for risk management. Table 3 shows a comparative analysis table.

A. DISCUSSION

Software risk management is a developing discipline whose crucial objective is to identify, assess, and reduce the perceived risks. Software risk management models are used to identify and control the risks. Different models have different features. In our study, we take some common features from the literature and conduct a comparative analysis of these models.

The Software Risk Evaluation (SRE) is a risk management process technique that provides a more comprehensive outline. This technique is considered good for risk identification and the analysis of the risks. The SRE is not only helpful for the risk management process but is also used as a decision-making tool. It provides a clear and understandable view of risks by the categorization of risks. It works on SEI (Software Engineering Institute) standards and addresses its basic elements like risk identification, risk analysis, planning and communication. It systematically tracks risks by taking snapshots of risks. The SRE composes a shared view and constructs a common framework for all team members in the risk mitigation process. This technique involves its team members but not all stakeholders. Lack of stakeholder's involvement makes the process vaguer and can lead to failure.

TRM (Team Risk Management) presents operational activities and an organizational structure for the risk management process. It manages risks in the whole software development process with the involvement of all individuals and stakeholders, which makes the decision-making process easier and more efficient. It also works according to the SEI standards and brings all participants (including investors, managers, developers, and customers) within the organization. TRM frequently and cooperatively ensures the continuous risk management process throughout software development since risks can exist in any phase of software development. In this risk management technique, the managers set the scheduled activities and execute them in a continuous cycle. Through TRM, the risks can be regularly identified and analyzed. It keeps track of risks with regular reviews and continuous monitoring of the risk mitigation process.

The Softrisk management technique [12] almost overcomes the shortcomings of previous and traditional techniques. It has become an emerging technique for software risk management. This technique ensures that the risk automation is suitable for any kind or size of project. Softrisk is constructed on the idea of documentation and concentrates on extreme risks by prioritizing the causes of big losses. The Softrisk technique identifies both the general risks that can occur on any project and specific risks that can occur in software development. This risk mitigation technique documentation is an important step for tracking current risks and monitoring future risks. Softrisk uses visual graphs for the identification of risks and signifies the most dangerous risks using the color red. In the last stage of the Softrisk model, re-estimation, re-prioritization, re-assessment and re-documentation will be performed for continuous software risk management.

The Riskit model has been theoretically developed and applied to risk management. This model is very famous because it has been applied in various industries in both the US and Europe. This technique is useful because of its systematic risk analysis, as it is based on qualitative risk analysis. This risk management tool uses a graphical user interface. It ranks the risks based on historical data and it uses graphical representation to document this technique. This technique also involves all the stakeholders of the project, which can reduce the causes of failure. This technique identifies risks and ensures continuous risk management on the basis of risk monitoring and tracking.

The project risk management framework is based on the two major activities of risk assessment and risk control. The other steps of this framework work under these primary categories. Project risk management is a framework for risk identification and specifies the specific risks that can cause the failure of the final product. In this framework, risk analysis and risk prioritization categorize and rank the risks, which helps identify the severity of the risks. This technique monitors and tracks the risks. It is considered to be effective for the risk estimation.

The risk management process starts with risk identification and lists all possible risks that can occur in the final product. After the identification of the risks, analysis and prioritization will be performed on the perceived and identified lists. A risk management plan is created to reduce the impacts of risks in the project. This framework also plans for what to do if the risk turns into a problem. This approach uses quantitative analyses for risk monitoring, and tracking is also implemented for to update current and future risks.

The **Capability Maturity Model (CMM)-**based model follows the CMM framework for risk mitigation. In this model, a database is used to identify and measure risks. This model combines risk assessment and risk control. Risk assessment is the first step for risk identification and risk prioritization. In the second step of risk control, risk mitigation plans are created, and risk monitoring will be performed that will update current risks and track future risks. This is a well-structured risk management framework that will keep a repository for identifying and controlling risk. This step plays a crucial rule for decision-making in risk management.

The analyzer for reducing module operational risks (ARMOR) is a risk analyzing technique that will automatically identify operational risk in software modules. It also creates a database and assesses the risks on the bases of the risk repository. This risk management framework is based on statistical quantities for the risk mitigation process. This tool identifies the project risks and highlights the sources of risks in the project. This tool is considered to be effective for risk estimation and it also helps as a decision-making tool. A detailed comparative study of methodologies is presented in table 3.

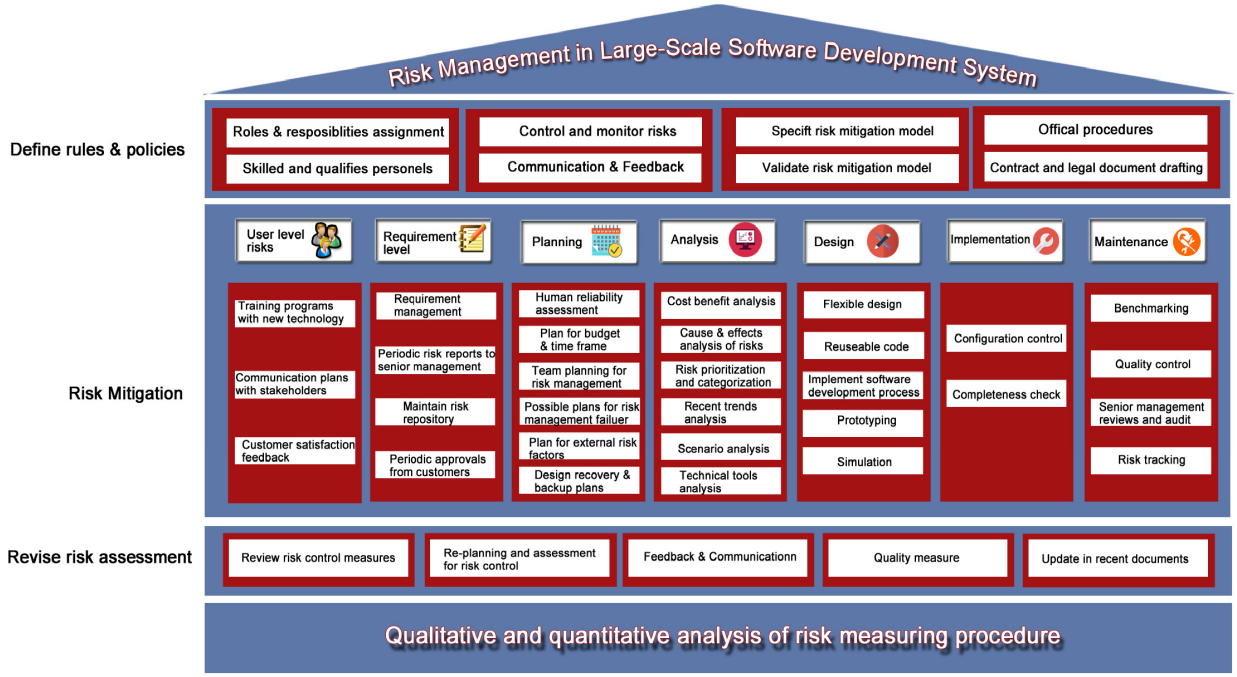

FIGURE 5. Proposed methodology for risk mitigation in large-scale systems.

V. PROPOSED METHODOLOGY FOR RISK MITIGATION IN LARGE-SCALE SYSTEMS

In recent trends, the scale of software development systems is increasing in a determined way [39]. Complexity, bandwidth, lines of code, memory, communication, investment, dataset, dependency and many other factors of a system are increasing day by day in a system, and risk mitigation in these types of systems is quite difficult [15]. In this methodology, some steps are suggested for risk mitigation in a large-scale software development system. The proposed methodology categorizes risk factors such as user level risks, requirement level, planning, analysis, design, implementation and maintenance, which were previously discussed. Figure 5 defines the risk mitigation in large-scale systems.

A. DEFINE RULES AND POLICIES

In the initial stage of this methodology, the organization defines some rules and polices. In this step, the rules and policies will be integrated, and the official work will be done here.

a. **Roles and responsibilities assignments:** Roles will be assigned to qualified and skilled persons. The top management will handle this process and assign an experienced manager to the project that will be able to look after the whole software development process. In large-scale systems, many employees are involved, and handling them is quite difficult and challenging. Therefore, in the early step, the roles and work will be assigned to the proper persons.

b. **Control and monitor risks:** In this step, the setup that will monitor and communicate with all team members in the risk mitigation process will be configured. Communication plans with top management and their feedback will be planned here.

c. **Specific risk mitigation model:** The model for the risk mitigation process will be specified. The top management will select the risk management model according to the needs of the project. After selecting the model, the model will be evaluated according to the project. The managers will select the risk mitigation model that will allow for the risk mitigation procedures to be applied on that particular project. The model is validated before it is applied to their project. The validation process will be performed by the top management of the organization.

d. **Official procedures:** In this step, all official work will be done. Contracts and all legal documents will be drafted and signed by the investors and project partners. In large-scale systems, this is an important step before proceeding to the risk mitigation step.

B. RISK MITIGATION

In this second step, the previously discussed risk mitigation process is categorized according to the user level, requirement level, planning, analysis, design, implementation and maintenance level in the software development.

a. **User level risks:** User level risks are the risks that will occur on the user side, such as low user involvement and not having proper knowledge of newly developed technology. For the prevention of these types of risks, the organization will train users on the new technology. Communication plans will be established with all stakeholders that will aid the organization in

communicating with users. In the final step, customer satisfaction feedback will be acquired. This step will be repeated in every step of development.

b. **Requirement level:** Sometimes users or customers are not able to describe their needs to the developer or the frequent change in requirements can be risky for the organization. Requirement management will be helpful for the organization to handle these types of risks. Periodic risk reports will be generated that will be delivered to the top management. The top management will look after the requirements and all changes performed by the users. The organization will maintain a risk repository that will maintain all possible risks, which will helpful for the organization to avoid future risks. The organization will assess periodic feedback from customers. This step can be helpful from both the customer and organizational points of view.

c. **Planning:** In this process, the planning will be done for all possible risks. A human reliability assessment will be performed by the managers and top management to detect human errors. Employees skills will be examined and their reliability will be checked. The budget and time frame will be planned for software development. Proper resources will be allocated to the team members of the project. Whole teams in the project will plan for risk management. The possible plans for risk mitigation failure will be designed so that if the risk management plan fails, the whole team can determine the next steps. The whole team and top management of that organization will have planned for all possible external risks factors. Finally, a backup and recovery plan will be designed in case any data failures occur.

d. **Analysis:** In this step, the whole planning will be analyzed by the team members. In the initial step, the cost-benefit analysis will be completed. The decision makers will assess if the risk mitigation process will be costlier than the benefits of the project. The causes and effects of the risks will be analyzed. After identifying all possible risks in the software development process, the risk will be categorized and prioritized according to its severity. Recent software development trends will be analyzed and applied to the project. Analysts will assess the whole software development life cycle and identify the overall possible risks. The technical tools and new technology will be analyzed in order to further avoid future risks. The reliability of technical tools is essential in large-scale systems, since many resources have been invested in the project.

e. **Design:** In the design phase, the designers and developers start developing project. The design should be flexible for any kind of minor change. It will be beneficial if the designer implements reusable code that will save time and costs. The project managers will select a proper standardized software development process according to the need of the project. The software of final product will be testing before it's finally implemented. This will be helpful in identifying all possible bugs in the project before its implementation.

f. **Implementation:** After designing and testing the whole project, the software will be implemented. The configuration of a large-scale software system is really challenging. The top management will track and monitor its configuration. After the configuration of the whole software project, the completeness check will be applied to all essential libraries and functions.

g. **Maintenance:** After implementing the system, the maintenance of the whole system is required. The organization will make a benchmark for this system. Top management will take care of the quality control of system. They will also review on possible risks. Risk tracking will be done in the final.

C. REVISE RISK ASSESSMENT

Revise risk assessment will be applied after risk mitigation, in the case any risk reoccurs. This establishes a continuous reassessment of the risk management process.

a. **Review risk control measures:** In this step, the reviews will be generated on the risk management process. Continuous reassessments will make this methodology a continuous risk management process. Revisions in the process make the system more quantifiable for risk management.

b. **Re-planning and assessment:** If there is a risk of reoccurrence, the step of re-planning will be implemented on project.

c. **Feedback and communication:** In this stage, the feedback and communication will be planned with all stakeholders and top management. This step will require the reapplication of any changes after the re-planning process.

d. **Quality measure:** Measuring the quality of final product is essential for organizational and customer satisfaction.

e. **Update in recent documentation:** The process of documentation will be last performed. Recent changes and trends will also be updated in the most recent documents.

D. QUALITATIVE AND QUANTITATIVE ANALYSIS OF RISK MEASURING PROCEDURE

In the last step, qualitative and quantitative analysis will be applied on the whole risk management process. For qualitative analyses, graphical charts and presentations will be designed. For quantitative analyses, surveys and interviews will be performed by the top management.

VI. CONCLUSION

The most crucial thing in software development is the assessment and control of risk factors. Many risk management techniques and supporting tools have been proposed during recent decades. The aggressive competition among organizations has forced the use these techniques to improve product quality and risk assessment. Consequently, this study

aims to provide a critical analysis of the different risk management models with some features extracted from the literature. This analysis helps others choose the best-suited model or framework for their organization. This research also contributes an exhaustive list of risk factors and their supporting tools for risk management. A comparative analysis table is illustrated that will help to analyze the identified risk management models from the existing literature. Some common features and factors are perceived and, with them, the models are analyzed. From this comparison table, we conclude that every model that has different specifications may be applicable in different environments.

REFERENCES

[1] M. Chowdhury, A. Al, and S. Arefeen, "Software risk management: Importance and practices," in *Proc. IJCIT ISSN*, 2011, pp. 2078–5828.

[2] Y. Amihud and B. Lev, "Risk reduction as a managerial motive for conglomerate mergers," *Bell J. Econ.*, vol. 12, no. 12, pp. 605–617, 1981.

[3] A. Cagliano, S. Grimaldi, and C. Rafele, "Choosing project risk management techniques. A theoretical framework," *J. Risk Res.*, vol. 18, no. 2, pp. 232–248, 2015.

[4] A. Elzamly, B. Hussin, and N. M. Salleh, "Top fifty software risk factors and the best thirty risk management techniques in software development lifecycle for successful software projects," *Int. J. Hybrid Inf. Technol.*, vol. 9, no. 6, pp. 11–32, 2016.

[5] A. Kumar, O. Samuel, X. Li, M. Abdel-Basset, and H. Wang, "Towards an efficient risk assessment in software projects–Fuzzy reinforcement paradigm," *Comput. Elect. Eng.*, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.compeleceng.2017.07.022

[6] T. Arnuphaptrairong, "Top ten lists of software project risks: Evidence from the literature survey," in *Proc. Int. Multiconf. Eng. Comput. Sci.*, vol. 1. 2011, pp. 1–6.

[7] S. Alhawari, L. Karadsheh, A. N. Talet, and E. Mansour, "Knowledge-based risk management framework for information technology project," *Int. J. Inf. Manage.*, vol. 32, no. 1, pp. 50–65, Feb. 2012.

[8] S. Marcelino-Sádaba, A. Pérez-Ezcurdia, A. M. E. Lazcano, and P. Villanueva, "Project risk management methodology for small firms," *Int. J. Project Manage.*, vol. 32, no. 2, pp. 327–340, Feb. 2014.

[9] M. M. de Carvalho and R. R. Junior, "Impact of risk management on project performance: The importance of soft skills," *Int. J. Prod. Res.*, vol. 53, no. 2, pp. 321–340, 2015.

[10] J. Knodel, M. Naab, E. Bouwers, and J. Visser, "Software risk management in practice: Shed light on your software product," in *Proc. IEEE 22nd Int. Conf. Softw. Anal., Evol. Reeng. (SANER)*, Mar. 2015, pp. 592–594.

[11] F. Aqlan, "A software application for rapid risk assessment in integrated supply chains," *Expert Syst. Appl.*, vol. 43, pp. 109–116, Jan. 2016.

[12] M. F. Rabbi and K. O. B. Mannan, "A short review for selecting the best tools and techniques to perform software risk management," *Eur. J. Adv. Eng. Technol.*, vol. 3, no. 6, pp. 1–7, 2016.

[13] C. Chittister and Y. Y. Haimes, "Risk associated with software development: A holistic framework for assessment and management," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, no. 3, pp. 710–723, May 1993.

[14] T. Tamai and A. Itou, "Requirements and design change in large-scale software development: Analysis from the viewpoint of process backtracking,"in *Proc. 15th Int. Conf. Softw. Eng.*, May 1993, pp. 167–176.

[15] M. Grabowski and K. Roberts, "Risk mitigation in large-scale systems: Lessons from high reliability organizations," *California Manage. Rev.*, vol. 39, no. 4, pp. 152–161, 1997.

[16] S.-M. Huang, "Assessing risk in ERP projects: Identify and prioritize the factors," *Ind. Manage. Data Syst.*, vol. 104, no. 8, pp. 681–688, 2004.

[17] M. F. Rabbi and K. O. B. Mannan, "A review of software risk management for selection of best tools and techniques," in *Proc. 9th ACIS Int. Conf. IEEE Softw. Eng., Artif. Intell., Netw., Parallel/Distrib. Comput. (SNPD)*, Aug. 2008, pp. 773–778.

[18] M. Grabowski, J. R. W. Merrick, J. R. Harrold, T. A. Massuchi, and J. D. van Dorp, "Risk modeling in distributed, large-scale systems," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 30, no. 6, pp. 651–660, Nov. 2000.

[19] L. Northrop, "Ultra-large-scale systems: The software challenge of the future," Softw. Eng. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep., 2006.

[20] L. Wallace and M. Keil, "Software project risks and their effect on outcomes," *Commun. ACM*, vol. 47, no. 4, pp. 68–73, Apr. 2004.

[21] E. Wallmüller, M. Wieczorek, U. Naujoks, and B. Bartlett, "Risk management for IT and software projects," in *Business Continuity*. Berlin, Germany: Springer, 2002, pp. 165–178.

[22] T. Raz and E. Michael, "Use and benefits of tools for project risk management," *Int. J. Project Manage.*, vol. 19, no. 1, pp. 9–17, Jan. 2001.

[23] A. Elzamly and B. Hussin, "Classification and identification of risk management techniques for mitigating risks with factor analysis technique in software risk management," *Rev. Comput. Eng. Res.*, vol. 2, no. 2, pp. 22–38, 2015.

[24] M. Sumner, "Risk factors in enterprise-wide/ERP projects," *J. Inf. Technol.*, vol. 15, no. 4, pp. 317–327, Dec. 2000.

[25] B. W. Boehm, "Software risk management: Principles and practices," *IEEE Softw.*, vol. 8, no. 1, pp. 32–41, Jan. 1991.

[26] R. Schmidt, K. Lyytinen, M. Keil, and P. Cule, "Identifying software project risks: An international Delphi study," *J. Manage. Inf. Syst.*, vol. 17, no. 4, pp. 5–36, 2001.

[27] H. Barki, S. Rivard, and J. Talbot, "Toward an assessment of software development risk," *J. Manage. Inf. Syst.*, vol. 10, no. 2, pp. 203–225, 1993.

[28] S. Coyle and K. Conboy, "A case study of risk management in agile systems development," in *Proc. 17th Eur. Conf. Inf. Syst.*, Verona, Italy, 2009, pp. 2567–2578.

[29] A. Elzamly and B. Hussin, "Modelling and evaluating software project risks with quantitative analysis techniques in planning software development," *J. Comput. Inf. Technol.*, vol. 23, no. 2, pp. 123–139, 2015.

[30] P. M. Chawan, J. Patil, and R. Naik, "Software risk management," *Int. J. Comput. Sci. Mobile Comput.*, vol. 2, no. 5, pp. 60–66, May 2013.

[31] K. Kaur, A. Kaur, and R. Kaur, "Study of different risk management model and risk knowledge acquisition with WEKA," *Int. J. Eng. Res. General Sci.*, vol. 2, no. 4, pp. 26–35, Jun./Jul. 2014.

[32] R. N. Charette, "Large-scale project management is risk management," *IEEE Softw.*, vol. 13, no. 4, pp. 110–117, Jul. 1996.

[33] M. R. Lu, J. S. Yu, E. Keramidas, and S. R. Dalal, "ARMOR: Analyzer for reducing module operational risk," in *25th Int. Symp. IEEE Fault-Tolerant Comput. (FTCS) Dig. Papers*, Jun. 1995, pp. 137–142.

[34] X. Ruzhi, Q. Leqiu, and J. Xinhai, "CMM-based software risk control optimization," in *Proc. IEEE Int. Conf. Inf. Reuse Integr. (IRI)*, Oct. 2003, pp. 499–503.

[35] J. Kontio and V. R. Basili, "Empirical evaluation of a risk management method," in *Proc. SEI Conf. Risk Manage.*, 1997, pp. 1–8.

[36] A. A. Keshlaf and K. Hashim, "A model and prototype tool to manage software risks," in *Proc. 1st Asia–Pacific Conf. IEEE Quality Softw.*, Oct. 2000, pp. 297–305.

[37] M. J. Carr, S. L. Konda, I. Monarch, F. C. Ulrich, and C. F. Walker, "Taxonomy-based risk identification," Softw. Eng. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU/SEI-93-TR-06, 1993.

[38] R. P. Higuera, D. Gluch, A. J. Dorofee, R. L. Murphy, J. A. Walker, and R. C. Williams, "An introduction to team risk management," Softw. Eng. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, Special Rep. CMU/SEI-94-SR-1 version 1.0, 1994.

[39] R. P. Gabriel, L. Northrop, D. C. Schmidt, and K. Sullivan, "Ultra-large-scale systems," in *Proc. Companion 21st ACM SIGPLAN Symp. Object-Oriented Program. Syst., Lang., Appl.*, 2006, pp. 632–634.

MARUF PASHA received the M.S. degree in IT from NUST, Pakistan, and the Ph.D. degree from the University of South Brittany, France. He is currently serving as the Head of the Department of Information Technology, Bahauddin Zakariya University, Multan. He has published a number of articles in national and international conferences and journals. His research interests include semantic Web, big data, and IoT systems.

GHAZIA QAISER received the B.S. degree in information technology from the Punjab College, Multan, Pakistan, in 2014, and the master's degree in information technology from Bahauddin Zakariya University, Multan, in 2016. She is currently pursuing the M.S. degree in information technology from Bahauddin Zakariya University.

UROOJ PASHA received the master's degree in industrial logistic and computer sciences, and the Ph.D. degree from the Molde University College, Norway, in 2015. She is currently an Assistant Professor with the Institute of Management Sciences, Bahauddin Zakariya University, Multan. She has published a number of articles in international journals.