International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 21 (2018) pp. 15397-15403
© Research India Publications. http://www.ripublication.com

Introducing a Sub Process and Stage Oriented Risk Analysis Model for Software Projects and applying it with a Case Study

Rajendrani Mukherjee
Assistant Professor, Department of Information Technology, Calcutta Institute of Engineering and Management, 24/1A, Chandi Ghosh Road, Kolkata-700040, West Bengal, India.

Aurghyadip Kundu
Calcutta Institute of Engineering and Management, 24/1A, Chandi Ghosh Road, Kolkata-700040, West Bengal, India.

Rintesh Ghosh
Calcutta Institute of Engineering and Management, 24/1A, Chandi Ghosh Road, Kolkata-700040, West Bengal, India.

Abstract

Each year organizations and government spend huge amount of money on IT hardware, software and services. Software Project Risk Analysis and Management is essential to prevent software disasters and save money as well as time. The sooner the risk is detected and mitigated, the easier it is to ensure the quality and accuracy of product delivery. So of late, risk analysis and management is widely researched. The need and demand to develop a holistic model that will cater all possible modern age software development scenarios are huge. In this paper, we have developed a new systematic risk management model with detail steps and practical working principle. The new model aims to overcome the limitations of the existing risk models which were proposed by Barry Boehm, Robert Charette etc. The key feature of this model is that it gives more flexibility and robustness as it is sub process oriented and promotes stage wise risk management for each stage of SDLC (Software Development Life Cycle). The incorporation of stage wise risk mitigation of this new model ensures the fact that risk management can be made a part of SDLC itself. The proposed model lists different risk mitigation strategies based on the need of that particular stage and clubs those strategies into sub processes which are attached to each stage. The model also introduces the concept of parallel processing among different sub processes of each stage to save time. We have also applied this new model for a case study to judge its implementation potential

Keywords: Model, risk management, sub process, software project management, priority, outsourcing.

INTRODUCTION

Risk is an entity that a software project cannot ignore. Approximately, 5 to 15 percent of the IT projects are abandoned midway because of technical or non-technical limitations [5]. There was a time when recognition of risk was considered a defeat. However facts have proved effective strategies of risk analysis helps to combat threats before they actually cause catastrophic project disaster [4].

During 1989, Barry Boehm first proposed a concrete framework to manage risk [2]. Boehm coined the term Risk Exposure. In this framework, two primary steps have been indicated for risk analysis, namely, risk assessment and risk control. A spiral model for software development is proposed in [3]. In late 80’s Robert N. Charette was also involved in applying risk analysis principles into large scale projects [4, 5]. According to him, very large scale projects do not fit into “normal science” and are not always solvable puzzles. Several software-intensive DoD programs are discussed in [17] which used SEI-designed (Software Engineering Institute) CRM (Continuous Risk analysis) protocol [6]. A risk analysis method, Riskit, [9] has been introduced by J Kontio of Nokia Telecommunications during 1997. Another risk analysis model, the Chaos model, has been introduced in [14]. The chaos strategy is based on the chaos model and the chaos life cycle. The thumb rule of chaos strategy is to resolve the most important issue first. Though important is a very relative term, according to Raccoon [13] it is defined as big, robust and urgent. A conceptual model on software project risk analysis, the Earth-Moon model, is introduced in [16]. This model proposed two types of lifecycles in the software project risk analysis and established the relation between them. The first lifecycle relates to software project’s birth-growth-maturity-death process while the second life cycle relates to risk analysis life cycle.

Existing risk analysis methods are often inadequate for this cyber age and have severe theoretical and practical limitations. Among these existing frameworks, very few approaches[6,17] used methods and tools for risk analysis while a good number of existing frameworks did not converse about tools and methods[4,5]. In some risk analysis frameworks, there is no distinction between continuous risk analysis and non-continuous risk analysis [2,3]. Similarly, stakeholder’s perspectives are considered in some risk analysis frameworks (Riskit model), but in most models it is not even mentioned.

In this context, designing of an explicit and systematic risk analysis method overcoming the previous limitations is of utmost importance. This paper proposes and illustrates a new risk framework which is actually sub process oriented and at the same time takes into consideration of stage wise risk mitigation techniques. So we named this model as SPSO (Sub Process and Stage Oriented) Risk analysis Model. As different stages of software development have different need, the proposed techniques are designed to cater the changing need of each stage of SDLC. Moreover, the techniques are linked with sub processes and eventually sub processes are attached to each stage. The model also leverages parallel processing concept by simultaneously firing the sub processes within one stage to save time. Sub processes from different stages of SDLC are not allowed to execute in parallel to avoid residual risk overflow. Our paper describes our proposed model with key features, diagram and tabular representation. A flowchart is prepared to understand the sequencing of the Risk Treatment Process. Finally, we have taken a case study to check the real life application potential of our proposed work. The effort is concluded with future indication.

RELATED WORK

During 1989-1991, Boehm proposed a framework [2] that actually helped in applying risk management principles into practices. In this framework, two primary steps have been indicated for risk management, namely, risk assessment and risk control. Risk assessment involves risk identification (itemize project specific risk items), risk analysis and risk prioritization while risk control involves risk management planning (risk avoidance, risk transfer, risk reduction etc.), risk resolution and risk monitoring. Boehm introduced the term Risk Exposure (RE) which is defined as RE = P (UO) * L (UO), where P (UO) is the probability of a risk happening and L (UO) is the loss to the affected party. In general, software projects often concentrated on a high P (UO) or high L (UO), but it is the combination of both that actually determines the risk exposure value and the subsequent ranking. During the late 80s, Robert N Charette has laid a foundation for risk management discipline focusing on proactive risk management. It is logically Charette's risk management paradigm rather than a model because it talks about an entire new platform. According to this framework [4], very large scale projects are not fitted into normal science and 50 – 65% of these mega projects are cancelled at midway. In 1996, the Software Engineering Institute of Carnegie Mellon University released a guidebook for integrating risk management principles into projects using different methods and tools. The SEI model [6] focused activities are software risk evaluation, continuous risk management, and team risk management. According to this model, risks are to be assessed throughout the entire life cycle of the project and new risks are to be identified on weekly or monthly basis. The chaos model [13] was proposed by L Raccoon in1995. This model describes software development as a chaotic activity rather than a sequential activity and in this model parallel activities are designed between software development and a two person game of strategy like Chess or Go. The rule of chess is equivalent to what we can do in a project. The moves of a game are equivalent to software code writing, attending meetings etc. As in the opening game, players decide a strategy for future moves similarly in an opening phase of software development, developers decide a framework to build the software. The Riskit model [9] was designed by J Kontio of Nokia Telecommunications during 1997 – 1999 in which the stakeholders' perspective is taken into consideration. It is recognized in this model that without analyzing stakeholders' perspective it is very difficult to understand project goals properly and hence prioritizing expectations become difficult. The Riskit analysis graph actually behaves like a traceability matrix which traces back goals to risk factors. The Earth-Moon model [16] is focused on the necessary integration of clubbing risk management practices into software development life cycle repeatedly as stressed by Boehm and Charette. The notable feature of this model is that the characteristics of software projects are taken into consideration and a relationship is built between two types of lifecycles of software projects. The first lifecycle relates to software project's birth-growth-maturity-death process while the second life cycle relates to risk management life cycle. The Earth Moon model differs from other models by focusing on stage wise risk management where risk management strategies may be different for different stages of software project. When the system planning stage is executed, risk identification, risk analysis, risk planning, risk tracking, risk control is restricted to the system planning stage only. This risk management lifecycle will restart when the next stage of software project starts, that is, the system analysis starts. In Earth-moon model, the software project life cycle is similar to Earth's rotation and the risk management life cycle is similar to Moon's rotary motion around the revolving Earth.

Introduction of Proposed Sub Process and Stage Oriented Risk Analysis Model

This section introduces our proposed risk management model that emphasizes on stage-wise risk management. The incorporation of stage wise risk alleviation confirms the fact that risk management is a part of SDLC itself rather than an ad hoc activity. The model identifies risk in each stage of software development life cycle such as system planning, system analysis, system design, system implementation, and system maintenance. Each stage has its own risk identification, risk analysis, risk management planning, risk tracking and control mechanism. We term the three activities - risk management planning, risk tracking and risk control as Risk Treatment. As the size of the software project increases, the necessity to perform stage wise risk management increases.

After identification and ranking has been done, each risk is associated with a sub process and each sub process has its own RT mechanism. For example, the system planning stage might have a risk information sheet for individual risk tracking, whereas the system implementation stage might have a risk spreadsheet to summarize a number of risks. The sub processes for one particular stage constitute a final parent process which is in turn associated with each stage of the software project development life cycle. The working principle of the model is depicted below.

- ✓ Risks are identified at all stages of software project development life cycle.

- ✓ The potential risks are identified at each stage from all possible sources and tagged with unique identification id such as R1, R2, and R3 etc.
- ✓ Each identified risk is then quantified by multiplying P with S where P indicates the probability and S specifies the Severity. Severity is ranging from 1 to 4. In our scale, 1 stands for low, 2 stands for medium low, 3 stands for medium high and 4 stands for high.
- ✓ The assessed risks are sorted according to their risk value.
- ✓ Our model stresses on the fact that this risk identification and risk assessment should be repeated every month to ensure continuous risk management. New risk factors might crop up midway or a farfetched risk might become highly probable because of changed scenario.
- ✓ Each ranked risk is then linked with a separate sub process for risk handling. Each sub process mainly involves the three different activities including the plan to nullify the risk, ways to track it and control it, collectively termed as Risk Treatment.
- ✓ Risk Treatment mechanism for each sub process is different, mainly based on the need of the stage.
- ✓ The sub processes within one particular parent process can execute in parallel to save time. However, sub processes from different stages cannot get fired simultaneously since the condition of starting the next stage is that all risks of the previous stage are handled properly. This restriction is imposed to minimize the residual risk.

For a single stage of software development life cycle, the model is diagrammatically represented in Figure 1.

A flowchart is presented in Figure 2.

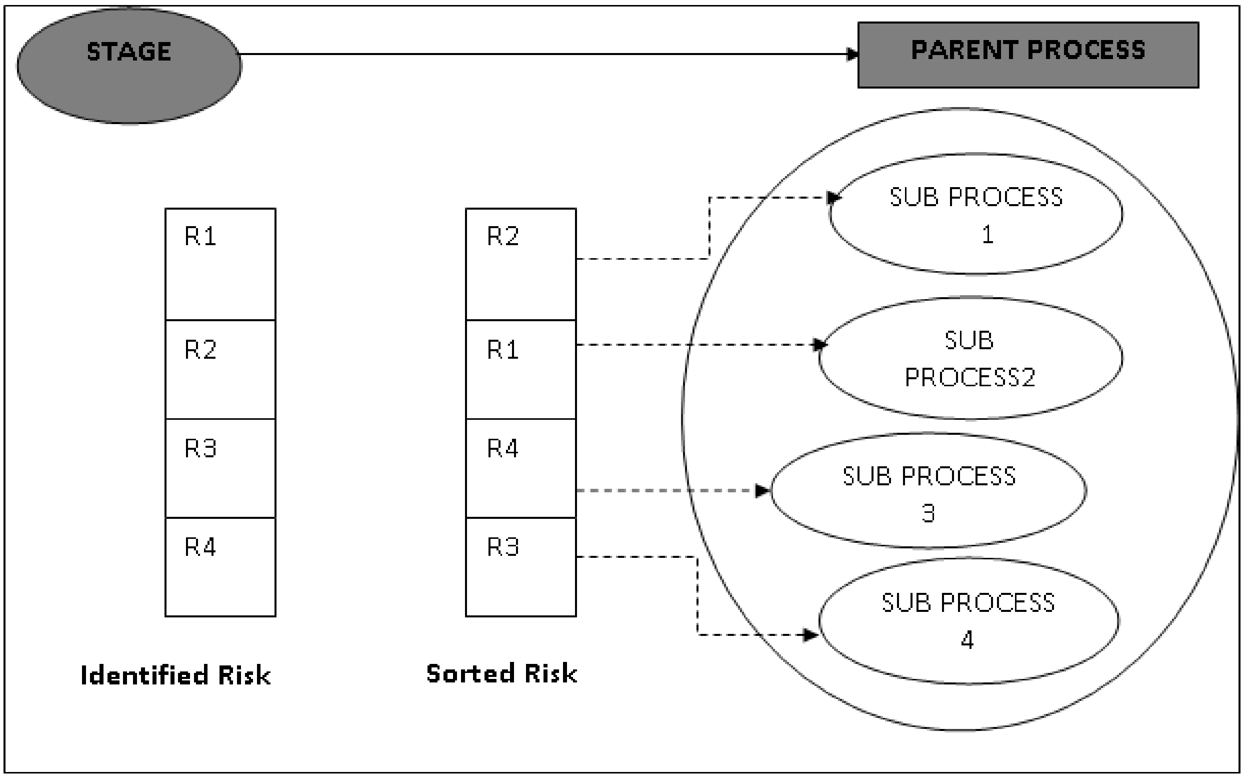

Figure 1: Block Diagram of SPSO Risk Management Model

Detailing of Proposed Sub Process and Stage Oriented Risk Analysis Model

The risk management life cycle (RMLC) of our proposed model has five steps – risk identification, risk analysis, risk management planning, risk tracking and risk control. In this section, we would like to detail each of them with specific techniques and tools which can be used to mitigate risks.

Risk Identification (RI)

RI is one of the most important jobs within the RMLC (Risk Management Life Cycle). However, very often it is casually done. It is wise to identify the sources of software project risk and prepare a plan to handle them when encountered rather than doing a post-mortem of a runaway project. In 1989, Boehm prepared a checklist to give special attention to top 10 primary sources of risk items [2]. In 2011, Stefanie Betz of Karlsruhe Institute of Technology added few more risk factors [1] from global software development perspective. IT off shoring and outsourcing of projects, current socio economic condition of global market, political movements also play pivotal role in building this risk factor list.

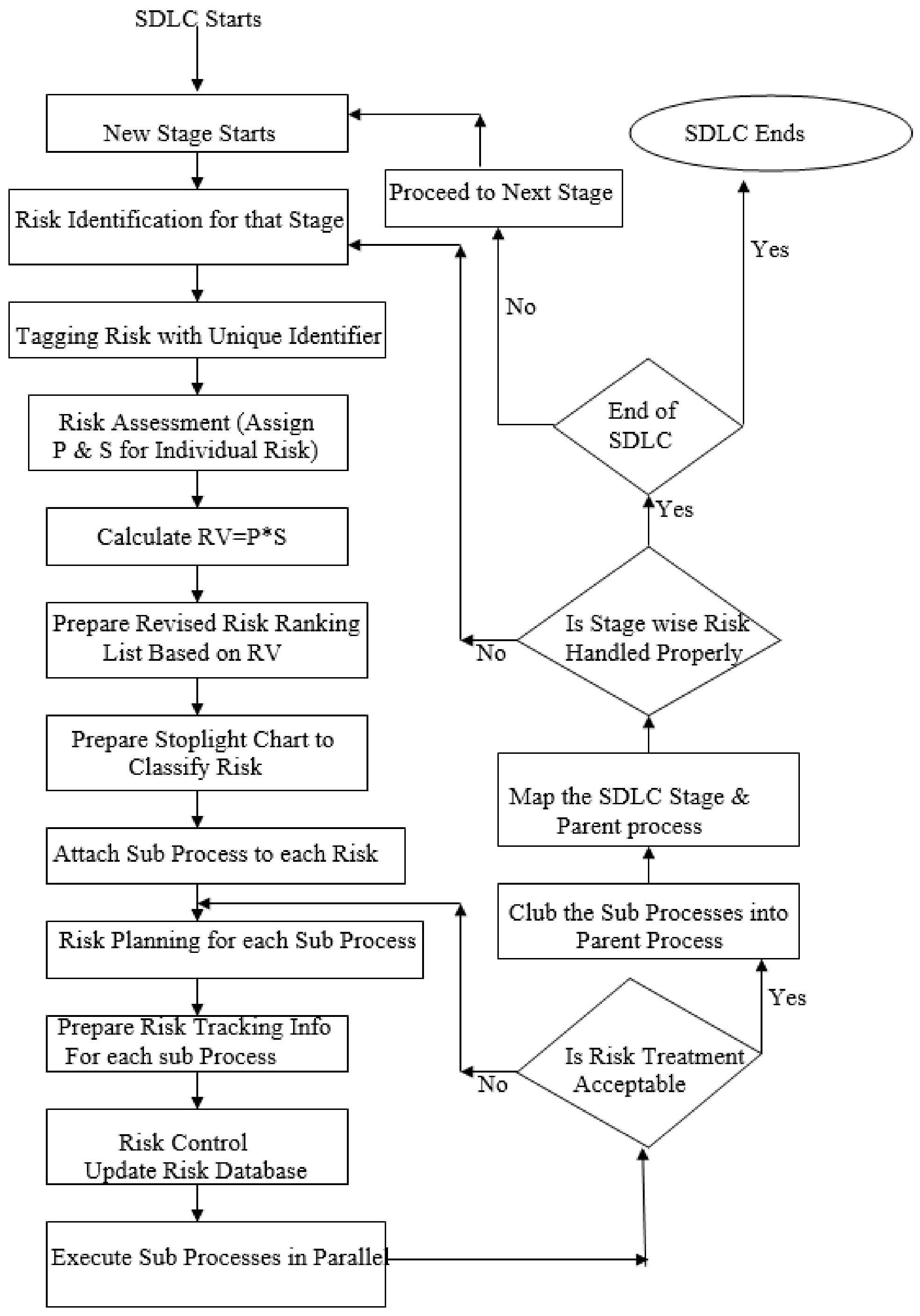
SDLC Starts
New Stage Starts
Risk Identification for that Stage
Tagging Risk with Unique Identifier
Risk Assessment (Assign P & S for Individual Risk)
Calculate RV=P*S
Prepare Revised Risk Ranking List Based on RV
Prepare Stoplight Chart to Classify Risk
Attach Sub Process to each Risk
Risk Planning for each Sub Process
Prepare Risk Tracking Info For each sub Process
Risk Control Update Risk Database
Execute Sub Processes in Parallel
Is Risk Treatment Acceptable
No
Yes
Club the Sub Processes into Parent Process
Map the SDLC Stage & Parent process
Is Stage wise Risk Handled Properly
No
Yes
End of SDLC
No
Yes
Proceed to Next Stage
SDLC Ends

Figure 2: Flowchart of SPSO Risk Model

We would like to propose two new sources of risk as they seem more crucial in current scenario. The new factors that we would like to introduce are –

- Conflict between different stakeholders (vendor, supplier, project manager, business analyst, buyer etc.). As various stakeholders have various interests and very often they are culturally diverse, sometimes conflict is inevitable. Stakeholder's participation or withdrawal is a major threat to software project.
- Unintentional Communication Problem among virtual team members. This problem might crop up from diverse time zone issue, different language proficiency of team members globally dispersed around or from the fact the concerned team member was not present at site and there was a lack in virtual knowledge transition (KT).

We would like to tabulate a comprehensive risk source list. Risk managers and team members may consult this Risk Factor

List, so that they do not miss any probable cause of project failure. *Our new model also stresses on the fact that risk source identification should be project specific rather than being generic.*

Risk Factors List:

- Inaccurate duration estimate/ unrealistic schedule
- Budgetary Constraint
- Incorrect, ambiguous , frequently changing requirements
- Conflict between different stakeholders
- Inadequate technical knowledge and developing wrong code
- Loss of key team member and lack of backup personnel
- Socio Cultural Political Factors
- Delay in global collaboration
- Unintentional Communication Problem among virtual team members

Risk Analysis (RA)

After Risk Identification, the next phase is Risk Analysis. The main purpose of Risk Analysis phase is to rank the risk so that we can set priority. Each risk is assessed by probability and severity. The main improvement in this SPSO model is that the proposed risk ranking mechanism is more practical and feasible from industrial context.

Boehm and others [2, 3] stressed on finding the real values of probability and severity. However, the real values of probability and severity are unknown or even knowable. Early detailed quantification of impact and probability is ineffective. For example, a group of 3 key team members can spend one or two days for making exact and detailed quantitative assessment of one identified risk. In reality, as the project steps forward, the worst case impact of this risk may be 4 days' work by one resource even if that occurred with a 20 percent probability. In other example, if a project manager decides to develop plans for handling 30 moderate risks, the effort to exactly quantify them becomes greater than the impact of those risks if they actually faced in reality.

So in this SPSO model, we focus on "quick and empirical" estimation of Probability and Severity, rather than accurate quantification of probability and impact. We urge the project manager or risk manager to gather his experienced resources and tag each risk with probability and severity rating.

In our model, the scales of severity is four graded (1-4). The Risk Value (RV) is calculated by multiplying Probability (P) and Severity(S). The risk list will be sorted based on the calculated RV value (P * S).We would like to build a stoplight chart to get better visualization. This chart will help to get a single shot picture of which risk needs immediate attention and which can wait.

Table 1: Stoplight Charting of severity for SPSO Risk Model

Urgency	Stoplight Color	Action Needed
High	Red	Immediate
Medium High	Blue	Near Immediate
Medium Low	Yellow	Gradual
Low	Green	Wait and Watch

According to this SPSO model each stage of the SDLC will have its own sorted risk list. A stoplight chart (Table 1) will be prepared for each stage to get better clarity. The benefit of this labelling for each stage is that it makes resource allotment easier. If a risk is flagged as red, an experienced key team member can take care of it. Also a red flagged risk will need frequent monitoring. Meanwhile, a new team member can be assigned to handle a yellow flagged risk.

Risk Management Planning

As the risks are identified and sorted, now we need to treat it by proper planning, tracking and control mechanism. We club these three activities of planning, tracking and control as Risk Treatment (RT) and link it to a sub process. Each sub process has its own risk treatment mechanism. The sub processes of one particular stage constitute a parent process. Thus traceability between SDLC stage (SS) and Parent Process (PP) is established.

Even though each risk will have its own plan to treat it, the risk planning phase of SPSO model focuses on following strategies-

1) The risk manager or project manager should stress on identifying new risks and revising the ranked risk list on a monthly basis. If required, a reminder may be set.

2) The weekly status report meeting should have a 5 or 10 minutes slot for discussing risks and their probable solutions. Project members can discuss risk items offline and can circulate a read ahead before the meeting. This will save time and create a "think risk" culture.

3) There will be clear risk mitigation strategies for each risk. A dedicated resource will be assigned to handle each risk keeping a tentative timeline in mind. For example, a software project operating in financial domain will have special arrangement for financial transaction security. As a measure, the project may use encryption mechanisms while sending transaction log files. It may also ban its employees from accessing internet while working on its client portal. Also the password management system will have a feature of resetting passwords on regular intervals.

4) The risk database will be updated on a weekly or monthly basis.

The Risk Planning phase of this SPSO model stresses on these four points. Each sub process will follow these baselines of risk planning phase. A lack of common risk planning may cause failure. The tool that can be used as part of risk planning phase is Gantt chart. It will help to get a pictorial view of allocated tasks, assigned resource and ending timeline.

Risk Tracking

The essence of Risk Tracking is monitoring and documentation. Complications may arise if one team member uses verbal reports; another team member uses no documentation. As the project size increases and the number of risk items increases it will be very difficult to remember which risks are still open and what actions have been taken to mitigate them.

The SPSO risk model focuses on building Risk Tracking Spreadsheet with all the identified risks with nine pieces of primary information (risk id, risk description, probability (P), severity (S) , risk value (P*S), stoplight flag, resource assigned, last reviewed date and mitigation status).

The main advantage of this spreadsheet is that it lists all the risks of an entire software project and helps to get a holistic view. It does not elaborate on one particular risk and the mitigation strategies, but it helps to get a single shot view of all the risks. This chart is very useful for project managers as well as senior developers as it portrays resource allocated per risk, mitigation status (WIP/Completed/On Hold etc.).

The SPSO model also utilizes another risk tracker which is termed as Risk Info Sheet. Contrary to Risk Tracking Spreadsheet, a risk info sheet gives detail information about one particular risk. Also, the tracking spreadsheet is mainly helpful to managers, whereas a risk info sheet is useful for developers to gain technical knowledge about how to encounter different risks in detail.

Risk Control

The last step in risk management life cycle or risk treatment process in SPSO model is risk control.

The SPSO model focuses on building a risk database or risk register. The database would have tables to store risk related data for all software projects for last 5 to 10 years. This will immensely help the new members of the team to get an idea of what can go wrong and how to handle them.

The risk register table will have select permission for all team members however update or insert permission can be granted to senior developers or project managers only.

Case Study Implementation

We have applied the Sub Process and Stage Oriented (SPSO) Risk model for ongoing projects in four different tutorial agencies in Kolkata, India and have analyzed it as a case study. This will help to judge the implementation potential of this SPSO Risk model in real scenario. We have presented eight possible risk source (Table 2) to the project managers of these tutorial centers and have taken their inputs on the severity of the risks and the probability of the risks happening. This has helped in calculating the average value of probability and severity of each risk item. Risk scenarios came into existence in all stages of SDLC and were solved with SPSO approach.

Table 2: Case Study Implementation for SPSO Risk Model

Stages of Software Project Life Cycle	Identified Risk for each Stage	Risk Value (Probability*Severity)	Sub Processes Attached to Mitigate Risk	Parent Process
System Planning	The number of students giving the exam is higher than the calculated capacity of the server.	3.75*0.22 = 0.825	Number of servers increased or students are grouped into batches	PP1
	Delay in result generation	1.25*0.22 = 0.27	Proper time frame should be charted.	
System Analysis	Exam countdown timer is not synchronizing properly with server time.	2.75*0.12 = 0.33	Timer should be checked regularly by conducting mock exams.	PP2
System Design	Ambiguous User Interface	2*0.10 = 0.20	Requirement specification should be clear.	PP3
System Implementation	The server runs into a frozen state in the middle of the examination.	4*0.23 = 0.92	Trial run should be done before actual exam and there will be back up.	PP4
	Internet gets disconnected or slower.	4*0.25 = 1	Change service provider.	
System Maintenance	Answers are not registering properly.	4*0.38 = 1.52	Answer database should be checked and committed on regular basis.	PP5
	Browser cache is conflicting with the next set of examination.	1*0.10 = 0.10	Cache memory should be periodically refreshed.	

Thus by use of SPSO model each relevant risk is identified and analyzed for that particular stage. Instead of preparing a risk list, "what stage has what risk" approach is much wiser. Probability and Severity values are assigned based on the conducted survey with the tutorial agencies. This stage wise risk segregation gives better modularity and makes risk analysis easier. After risks are quantified, sub processes are attached to each risk. Each sub process is different and has its own risk mitigation strategy as mentioned in Table 2. Sub processes for one particular stage can execute in parallel and are clubbed into a parent process.

CONCLUSION

In this paper, a sub process and stage oriented risk management model has been developed for software project organization. As risk analysis is a collective daily effort, every one of the project should be involved. Well defined techniques for each stage of this risk mitigation framework are also established. The working principle of the proposed risk monitoring model is illustrated with the help of a case study.

However, no specific tool has been developed for the implementation of the proposed risk supervision model. In future work, we plan to design a comprehensive tool for handling risks at each stage of software project development life cycle.

REFERENCES

[1] S. Betz, "Risk Management in Global Software Development Process Planning", 37th EUROMICRO Conference on Software Engineering and Advanced Applications, 2011.

[2] B.W. Boehm, "Software Risk Management: Principles and Practices", Proceedings of the IEEE Journal of Software, Vol. 8, No.1, January 1991, pp. 32- 41.

[3] B.W. Boehm, "A Spiral Model of Software Development and Enhancement", Proceedings of the IEEE Journal of Computer, May 1988, pp. 61 – 72.

[4] R.N. Charette, "Large-scale Project Management is Risk Management", Proceedings of the IEEE Journal of Software, Vol.13, No.4, April 2002, pp. 110 - 117.

[5] R.N. Charette, "Why Software Fails", Proceedings of the IEEE Spectrum, September 2005, pp. 42 - 49.

[6] A. Dorofee et al., "Continuous Risk Management Guidebook", SEI, Carnegie Mellon University, Pittsburgh, 1996.

[7] C. Jones, "Minimizing the risks of software development", Cutter IT Journal 11, 1998, pp. 13-21.

[8] Y.H Kwak, Stoddard J, "Project risk management: lessons learned from software development Environment", Elsevier Science Ltd., Technovation 24, 2004, pp. 915-920.

[9] J. Kontio, "Risk Management in Software Development: A Technology Overview and the Riskit Method", Proceedings of the International Conference of Software Engineering (ICSE99), Los Angeles CA, 1999, pp. 679-680.

[10] J. Kontio, V.R. Basili, "Empirical Evaluation of a Risk Management Method", Proceedings of the SEI Conference on Risk Management, Atlantic City, NJ, 1997, pp. 1- 8.

[11] J. Kontio, G. Getto, D. Landes, "Experiences in Improving Risk Management Processes using the Concepts of the Riskit Method", Proceedings of the Sixth International Symposium on the Foundations of Software Engineering, Florida, USA, November 1998, pp. 163 – 174.

[12] C. Lindholm, M. Host, "Introducing Usability Testing in the Risk Management Process in Software Development", IEEE, SEHC 2013 San Francisco, CA, USA.

[13] L. Raccon., "The Chaos Model and the Chaos Life Cycle", Notes on Software Engineering, Vol.20, No. 1, January 1995, pp. 55 – 66.

[14] L. Raccon., "The Chaos Strategy", Notes on Software Engineering, Vol.20, No. 5, December 1995, pp. 40 – 47.

[15] Linda Wallace, Mark Keil, Arun Rai, "Understanding software project risk: a cluster analysis", Elsevier, Information & Management 42, March 2014, pp. 115-125.

[16] Yu-Heng Wang, JiaJian ,Ying , Qu "The Earth-Moon Model on Software ProjectRisk Management", Proceedings of the Ninth International Conference on Machine Learning and Cybernetics, Qingdao, July 2010, pp. 1999 - 2003.

[17] R. C. Williams, J. A. Walker, A. J. Dorofee, "Putting Risk Management into Practice", Proceedings of the IEEE Journal of Software, June 1997, pp. 75 – 82.