Research on risk management of software development projects

Dr. Krati Dubey
Assistant *Professor*
Manipal University Jaipur
Dehmi Kalan , Off Jaipur-Ajmer Expressway, Jaipur, Rajasthan 303007, India,
er.kratidubey@gmail.com

Harsh Raj
Manipal University Jaipur
Dehmi Kalan , Off Jaipur-Ajmer Expressway, Jaipur, Rajasthan 303007, India,
harsh.219302137@muj.manipal.edu

Ankit Gupta
Manipal University Jaipur
Dehmi Kalan , Off Jaipur-Ajmer Expressway, Jaipur, Rajasthan 303007, India,
Ankit.219302079@muj.manipal.edu

Abstract

Within the domain of computer program improvement, exploring through challenges and vulnerabilities is unavoidable. All through the improvement lifecycle, dangers linger huge, requesting capable administration for extend victory. This paper dives into five key reasons to comprehensively get a handle on the nature of these dangers:
the selection of untested innovations, perplexing framework necessities, strong framework design, ideal framework execution, and organizational or non-functional complexities. Successful hazard administration includes techniques to maintain a strategic distance from, contain, relieve, and screen potential pitfalls. The central point of computer program advancement extend hazard administration ought to be on hazard diminishment, avoidance, persistent evaluation of potential issues, exact hazard definition, prioritization, and proactive determination. This paper proposes a organized approach comprising four steps:
chance recognizable proof, surveying the presentation level for each hazard, executing chance moderation procedures, and concluding the chance administration handle. Furthermore, three sorts of countermeasures—careful, normal, and flexible—are suggested for dealing with different dangers in computer program improvement ventures.

1. Introduction

In the world of software development, this method carries significant risks that permeate the entire development process.
Ambitious initiatives often seek to push the limits of current software capabilities and venture into unexplored territory in search of unexpected successes [1]. However, such activities are inherently dangerous and must be handled appropriately.
Success depends on successfully mitigating these risks, especially in an industry where rapidly built, modern software is the norm RISKS IN SOFTWARE

DEVELOPMENT PROJECTS

Risk in software development projects encompasses dynamic variables with a substantial impact on success, from project abandonment to missed deadlines. Stemming from factors like untested technologies, evolving requirements, complex architectures, performance concerns, and organizational issues, risks demand strategic handling. Integrating new tech is common but improper use, often due to knowledge gaps, can lead to failure. Custom software requirements guide development, yet finding a precise solution is challenging. Changing requirements during development poses significant risks.
Early architecture decisions may not align with all requirements, and verifying effectiveness through prototypes in later stages carries potential risks. Ensuring non-functional needs' satisfaction requires performance testing on realized solutions, making early prediction challenging.

Early software architecture decisions that are critical to development do not necessarily align with all solution requirements .Verifying the effectiveness of an architecture is typically done through software prototypes in the later stages of development, which comes with potential risks. It is important to ensure satisfactory software performance for non-functional user needs.

However, performance testing is performed on realized solutions, which makes it difficult to accurately predict performance at an early stage, leading to inherent risks.

Risks related to organizational or non-functional issues are associated with project resources and schedules.

Organizational issues can impact software solutions, highlighting the need for an efficient organization to ensure project success.

Defined schedules can pose risks due to unforeseen events that lead to delays and require effective management to balance customer and developer satisfaction.

Managing risk in software development includes various options such as avoidance, containment, mitigation, and monitoring.

Ideally, risk should be avoided completely through the use of different technologies, flexible requirements, or tailored project plans.

For unavoidable risks, containment is an option to limit the impact on specific parts of the project, and this is achieved through identification and strategic changes to the project.

Containment or monitoring is essential for risks that cannot be avoided or suppressed.

Realizing a software prototype can help reduce some risks by testing and understanding the potential impact.

Contingency plans are important for risks that cannot be avoided or mitigated and include actions to be taken if the risks occur. Unmitigated risks pose a serious threat to software development projects and should be treated with great importance.

After such risks occur, a comprehensive evaluation of the project and decisions regarding the future of the project are essential.

Risks can be identified and addressed at various stages of a software development project but given the potentially significant costs associated with emerging risks, early identification is essential for prompt action. Organizational or non-functional issues pose risks related to resources and schedules. Effective organization is crucial, and defined schedules may face risks, necessitating management balance. Managing risks involves avoidance, containment, mitigation, and monitoring. Ideal avoidance uses tech, flexible requirements, or tailored plans. Containment limits impact on specific project parts, achieved through identification and strategic changes. Monitoring is crucial for unavoidable risks.

Realizing a software prototype helps test and understand potential impacts, while contingency plans address unmitigated risks. These risks pose a serious threat, requiring comprehensive project evaluation and decisions for the future.

Identifying and addressing risks early is vital, given the potentially significant costs.

2. RISK MANAGEMENT STEPS

Compelling hazard administration of program improvement ventures includes proactively moderating and dodging dangers, ceaselessly evaluating potential issues, characterizing key dangers, and killing them. It could be a vital arrangement.

Fruitful chance management relies on a comprehensive understanding of the whole extend and requires four fundamental steps.

The primary step includes hazard distinguishing proof, which includes distinguishing all variables that may lead to extend disappointment.

These variables are related to the innovation and organizational angles utilized within the computer program advancement prepare.

Careful perceptions and evaluations are carried out to reveal potential dangers.

Vital subtle elements such as chance portrayal, likelihood of event, related costs, conceivable chance arrangements, and measures to dodge them are recorded.

The moment step in distinguishing dangers is to survey the level of presentation for each recognized chance.

Dangers are efficiently positioned concurring to their impact, and needs are decided by the seriousness of their impact.

Dangers with critical affect are given higher need because it is more cost-effective to resolve them early instead of at a afterward organize.

The following step centers on chance relief, with the point of maintaining a strategic distance from the distinguished dangers or anticipating them from happening.

Relief methodologies incorporate evasion, lessening, and transmission. Although shirking is the ideal approach, it frequently requires major reorganization of the advancement prepare and can show challenges. Decrease is the vital replanning of a project to decrease the probability of a chance happening, and migration is the coordinating of hazard to ranges where it has less affect. The ultimate step, chance evaluation, happens after the hazard relief arrange is created.

This step incorporates all activities related to prevention and relief, point by point within the crisis plan, and diagrams the steps to be taken within the occasion of a chance.

3. COUNTERMEASURES FOR EFFECTIVE RISK MANAGEMENT

To achieve effective risk management in software development projects, three distinct countermeasures can be employed based on the magnitude of risk.

The first countermeasure is the careful approach, tailored for fresh and inexperienced organizations embarking on software development projects involving new and unproven technologies.

This high-priority risk management method requires comprehensive risk analysis at multiple levels.

Risks are scrutinized and tracked on individual, team, and organizational levels, involving active participation from every project team member.

Formal risk management roles are defined and specific project members are designated with primary responsibility for risk management Considering the variety of risks addressed by prudent measures, organizations establish formal risk interpretation and classification policies to quickly differentiate and address key risks.

Project team members continuously monitor risks and related actions and prioritize development activities based on the importance of the associated risks.

The second measure is a typical approach suitable for mature and experienced software development organizations

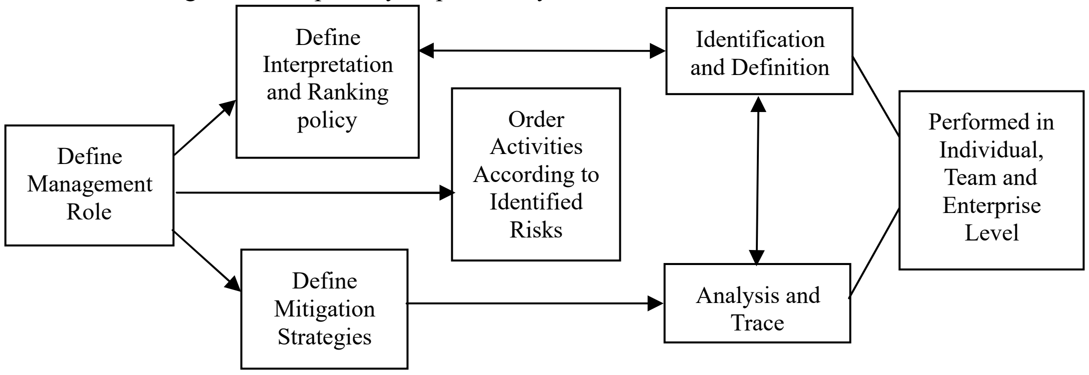
Define Management Role
Define Interpretation and Ranking policy
Order Activities According to Identified Risks
Define Mitigation Strategies
Identification and Definition
Analysis and Trace
Performed in Individual, Team and Enterprise Level

Figure 1 Activities of careful countermeasure

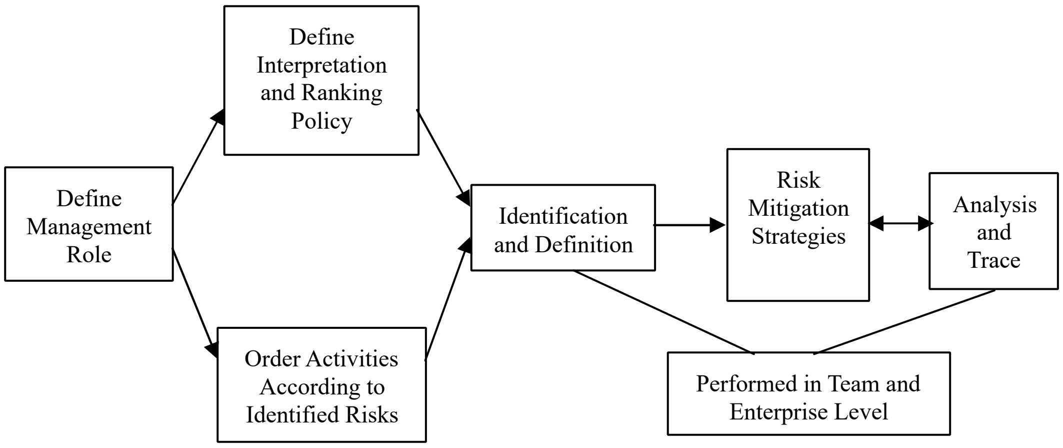
Define Management Role
Define Interpretation and Ranking Policy
Order Activities According to Identified Risks
Identification and Definition
Risk Mitigation Strategies
Analysis and Trace
Performed in Team and Enterprise Level

Figure 2 Activities of typical countermeasure

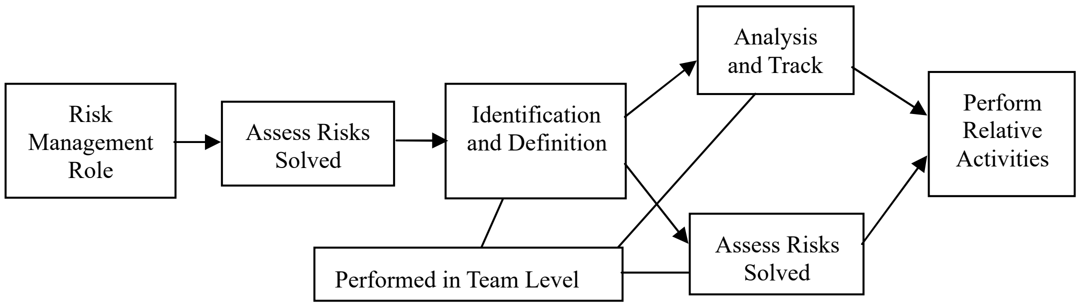

Figure 3 Activities of flexible countermeasure

This measure identifies risks at a moderate tolerable risk level and at corresponding stages as the project progresses. Activities of the careful risk management countermeasure are presented in Figure .
1.

Well-defined project risks are related to the entire project, so risk management is primarily done at the team and organizational level.

Unlike the prudent approach, general measures do not require the role of a formal risk management project.

Risks are identified by the project management team and, in some cases, team members.

Because only a limited number of risks are considered important, the project plan is stable and changes from time to time.

Risk interpretation and classification policies are less formal and rely on the project team to define and classify risks.

The third measure is a flexible approach designed for mature organizations with formally defined software development projects that leverage well-known and proven technology with an efficient organization.

This approach assumes a small number of acceptable risks and uses a relatively informal definition of risk.

Flexible measures focus on defining risk mitigation measures and contingency plans for a few selected key risk, with the aim of minimizing the workload associated with risk management. This countermeasure is based on comparing currently identified risks to previously encountered ones and risk management is practiced mostly in the organizational level. There is no risk interpretation and ranking policy and risks are identified at the beginning This action is primarily taken at the organizational level and is based on comparing currently identified risks with previously encountered risks.

There is no formal policy regarding the interpretation and classification of risks, and risks are initially identified at the beginning of a project and, in some cases, throughout the project lifecycle.

The initial project plan is developed based on the identified risks, so that if the risks ultimately materialize, the impact on the project plan will be minimal. Activities of the flexible risk management countermeasure are presented in Figure 3.

4. Conclusions

Risk is inherent in software development projects, and project success is closely related to implementing an effective risk management strategy.

Risk factors common to most projects include the use of new and untested technology, evolving system requirements, complex system architecture, system performance considerations, and organizational or non-functional aspects.

To manage and mitigate these risks, basic steps in software development project risk management are essential. It involves identifying risks, assessing the threat level of each risk, mitigating the risks, and completing the risk management process.

Moreover, three customized measures: prudent, standard, and flexible, provide an effective approach to manage software development project risks based on different scales.

In summary, successfully executing a software development project requires a proactive and strategic approach to identifying, assessing, mitigating, and containing risks.

The choice of countermeasures depends on the specific risk situation, and adaptability and careful consideration are important in risk management practice.

5. References

[1] Kwak Y H, Stoddardb J (2004). “Project risk management: lessons learned from software development environment”. Technovation, Vol. 24, No.11, pp.915-920.

[2] Dan X.Houston, Gerald Tod (2001). “Stochastic simulation of risk factor potential effects for software development risk management”. Journal of Systems and Software, Vol. 59,No.3, pp.247-258.

[3] Morcio Oliveira Barros (2004). “Supporting risks in software project management”. Journal of Systems and Software, Vol. 70, No.1, pp.21-35.

[4] Gang Xie, Jin Zhan (2006). “Risk avoidance in bidding for software projects based on life cycle management theory”. International Journal of Project Management, Vol. 12,No.4, pp.516-521.