An Investigative Survey on Technological Risks in Software Projects

Emanuel Dantas
emanuel.filho@ifpb.edu.br
Federal Institute of Paraíba - IFPB

Ademar Sousa
ademar.sousa@virtus.ufcg.edu.br
Federal University of Campina Grande - UFCG

Danyllo Albuquerque
danyllo.albuquerque@ifpb.edu.br
Federal Institute of Paraíba - IFPB

Luís Henrique Lima
lima.luis@academico.ifpb.edu.br
Federal Institute of Paraíba - IFPB

ABSTRACT

A project risk is an uncertain event or condition that, if achieved, affects at least one of the project's objectives. Technological risks have different characteristics from ordinary risks. These risks are typically more uncertain and subjective. Unfortunately, the literature is scarce regarding the presentation of technological risks. To fill this gap, we conducted a Survey with specialists in the field to assess the main risks and their impacts on software projects. As a result, three risks were cited by more than 70% of experts:"Failed to perform asynchronous tasks", "Integration with plugins failed", and "Loss of connection in HTTP request". Our study presents an emerging study, as future work a case study is planned to collect more information from the industry.

CCS CONCEPTS

• **Software and its engineering → Risk management**.

KEYWORDS

Risk Management, Technological Risk, Risk Identification

ACM Reference Format:
Emanuel Dantas, Ademar Sousa, Danyllo Albuquerque, and Luís Henrique Lima. 2021. An Investigative Survey on Technological Risks in Software Projects. In ,. ACM, New York, NY, USA, 4 pages. https://doi.org/10.5753/ise.XXXX

1 INTRODUCTION

Risks are inherent in software development [24–26], and managing them is crucial for the success of software projects [14, 18, 26, 39]. Risk management is a popular topic, being present in several industry guidelines or standards [15, 17, 29, 32, 37]. The Project Management Body of Knowledge (PMBOK) defines seven processes for risk management [32], with activities from planning to risk monitoring. Additionally, the ISO 31000 standard presents principles and recommendations in five activities that address risks [17]. Good practices and processes for managing risks can also be found in CMMI [29], PRINCE2 [15], and the Scrum Guide [37].

Identifying project risks is one of the most challenging activities in risk management [6, 21, 26]. In industry, different methods are used for this purpose. Examples include graphic methods (e.g., cause and effect diagrams [12, 34]), SWOT matrix [35], and systematic methods (e.g., probability and impact matrix [40], and checklists [38]). Furthermore, companies have applied intelligent techniques [30], including Artificial Intelligence and Data Analytics, to support the identification of project risks [10, 13, 27, 31].

The literature contains several approaches for managing software project risks focusing on distinct factors. Most risks are related to factors such as requirements (e.g., duplication, clarity, ambiguity of requirements), team formation (e.g., turnover, motivation, skills), and aspects of the development process (e.g., management knowledge, quality process, delivery information) [3, 8, 9]. However, there is a lack of solutions for managing technological risk factors. Managing technological risks is essential to the success of software projects. Technological risks have characteristics distinct from regular ones [7, 16, 22]. Usually, these risks are more uncertain and subjective [1, 4, 19]. Given the high volatility, accurately predicting the impacts of technological risks is difficult [20, 28].

Dantas et al. [5] used Grounded Theory to identify the main technological risks in software projects. The authors interviewed experts from organizations that execute software projects and created a catalog of twenty-two technological risks. However, no secondary studies have investigated the occurrence and impact of these technological risks.

In this sense, we aim to expand the analysis by [5] in this work. We used the catalog of technological risks in an online questionnaire and asked participants to rate the frequency of these risks. We received responses from 52 professionals from eighteen software development companies. The remainder of the paper is organized as follows: Section 2 presents the proposed methodology of the paper. Section 3 consists of experimental results. In Section 4, we discuss the conclusion of this work.

2 PROPOSED METHODOLOGY

Since the research question of this study aims to gather experts' opinions, we chose a survey as our research instrument [23, 33]. We designed the Survey to be as short as possible while still collecting all relevant information [36]. In a previous work, Dantas et al. [5] identified twenty-two technological risks in software projects. In this work, we want to validate these risks with more professionals through a Survey.

2.1 Survey Questions

We divided the survey into two parts. Table 1 presents the questions we used to collect demographic information about respondents. Table 2 shows the risks we evaluated in this research, using the question: "What is your level of understanding of the following risks in software projects?" and the possible answers. We drew inspiration for several questions from other surveys [33].

ISE '23, CBSoft '23: Brazilian Conference on Software: Theory and Practice
© 2023
https://doi.org/10.5753/ise.XXXX

Table 1: Questions about the respondent's demographic information

Question Text	Answer Choices
How old are you?	#
Which gender do you identify yourself with?	[Male, Female, Prefer not to disclose]
What is your highest education level?	[High school, Bachelors, Masters, Ph.D]
How many years of software development experience do you have?	[Less than a year, between one to five years, between six to ten years, more than ten years]
What is your current position in the company where you work?	[Trainee, Jr. Developer, Full Developer, Senior Developer, Architect, Manager]

Table 2: Risks evaluated in this research

Risks	Answer Choises
Integration with plugins failed	(1) Unaware of this risk, (2) Aware of the existence of the risk, (3) Aware of and treated thus risk.
Failed to close notification cycle	
Two Phase Commit	
Fail with hotspots in third-party app	
Configuration loss in versioning	
Connection failed to view media	
Loss of data stored on hybrid server	
Loss of connection in HTTP request	
Slowness in operations with aligned vectors	
Failure of data scalability	
Data durability state failure	
Failed to perform asynchronous tasks	
Offline persistence security flaw	
Encryption engine error	
Security flaw in decoding tokens	
Loss of authentication in restricted area	
Low performance in the use of sockets	
Low performance in microservice communication	
Failed to update drivers	
Failure in geolocation treatment	
Low performance of builds	
Unavailability of multimedia resources	

To help ensure the understandability of the survey, we asked Computer Science professors and graduate students with experience in Software Engineering (SE) and survey design to review the survey and ensure the questions were clear and complete. The feedback only suggested minor edits. The changes we made include adding more answer choices (three options for risks) and clarifying examples about the list of risks.

2.2 Participant Selection

We sought only software project developers with sufficient experience to ensure valid results. We identified 52 professionals based on the following three criteria:

- Program in at least two programming languages.
- Know development processes and project management.
- Have worked on at least one corporate software project.

We use LinkedIN[1] to identify contributors. We sent out 94 invites and had 52 responses.

[1]https://www.linkedin.com/

2.3 Data Collection

On June 23, 2023, we sent details of the research to each of the 94 experts. We also asked the respondents through both solicitation emails and a reminder in the survey to answer our questions based on their personal experiences with software projects. Since 6 of our solicitation emails bounced, we had 88 potential participants, assuming all other emails reached their intended recipient. On July 07, 2023, we sent a reminder email. We closed the survey on July 25, 2023, after the response rate slowed to almost no daily response.

Data from the survey link created with Google's URL shortener showed 68 clicks on the survey URL (72.34% of the invitations). Of those clicks, 55 people took the survey with a response rate of 80.88% (55/68). Since some of the questions were optional, many respondents skipped some of the questions. Only 45 respondents answered all the questions. After excluding the three responses that did not answer at least 25% of the questions, we were left with 52 responses for analysis.

3 EXPERIMENTAL RESULTS

The following subsections describe the results of our survey by answering the questions introduced in Section II. Initially, we show

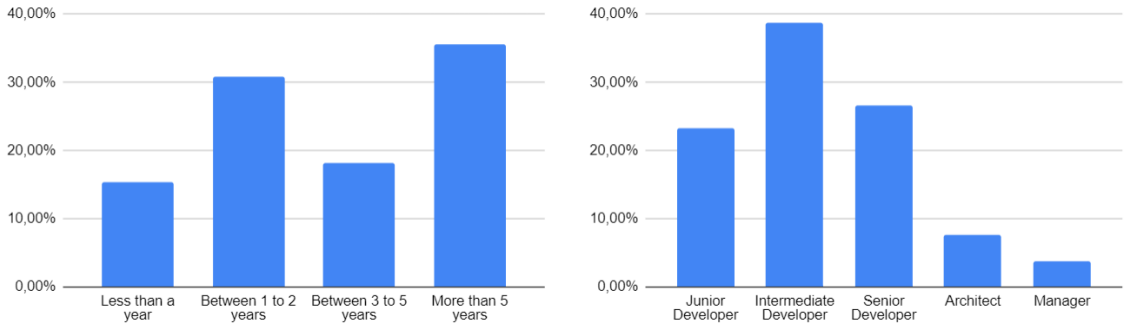

Figure 1: Demographics of the respondents

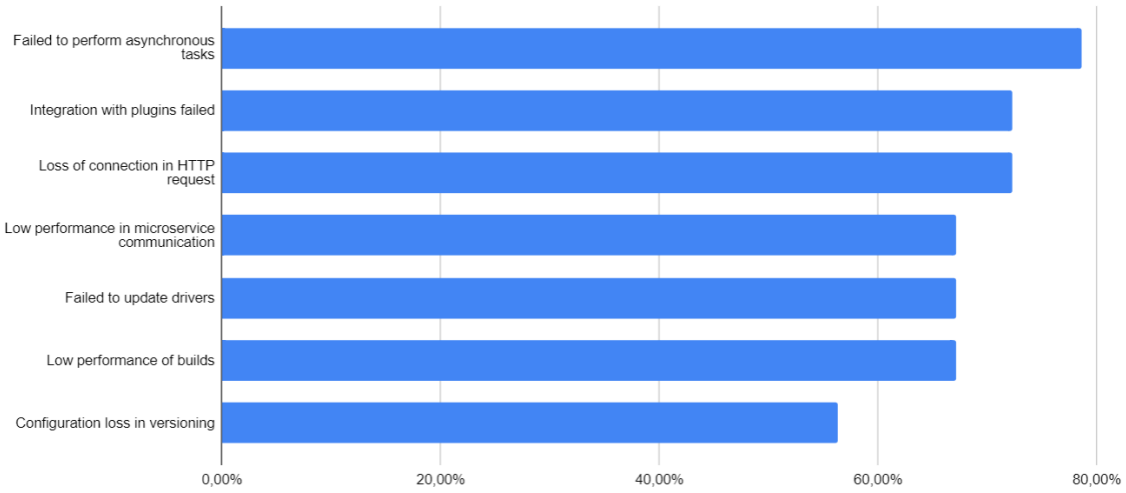

Figure 2: Technological Risks More Cited for Participants

the participants' demographic data resulting from the first questions on the form. Next, we present the quantitative values for each evaluated technological risk.

3.1 Respondents' Demographics

In response to demographic questions, 64.4% of our respondents indicated having less than five years of software development experience, while 15.40% had less than a year (Fig. 1A). In terms of age, 54.12% are under 28 years old, and only 7.7% are over 40 years old. Most are male (69.2%) and have completed higher education (92.6%). Regarding the positions held in the companies, the majority hold positions of Intermediate developers or higher functions with 76.75% (Figure 1B).

3.2 Technological Risks

Below are some analyses of the results collected. First, Figure 02 shows the number of risks. For each risk, the participant assesses whether the risk is recurrent in the projects in which they participate. For reasons of space, we present only the first seven cited.

Participants were free to choose one or more risks on the completed form. We concluded that the most common risks in software projects are "Failed to perform asynchronous tasks", "Integration with plugins failed" and "Loss of connection in HTTP request". It should also be noted that at least one of the survey participants cited all risks identified in the bibliographical research.

The risk of running asynchronous tasks is related to the programming language's ability to create event loops. In other words, starting concurrent code without using multiple threads. The solution is not trivial according to the languages and libraries used in software development. It is a recurring risk in software projects in the view of the participants.

The risk of integrating plugins is related to security issues and incompatibility of libraries. In some programming languages, managing plugins used in development is not automatic, and using different resources becomes a challenge for software development.

Finally, HTTP request failure risks refer to client and server errors that prevent a website from loading. Some of these errors are common (i.e., errors 403, 404, 500, and 503) and are considered risks to projects. According to the languages used for software development, scripts or requests may not be understood. Library changes must be made to solve problems and result in code refactoring.

4 CONCLUSION

Despite many technological risks common in software projects, few empirical SE research studies have explored this area [2, 11]. To fill this gap, we surveyed professionals to assess the main technological risks in software projects. For this, we used the pioneering work by Dantas et al. [5] and expanded the analysis with a survey involving 52 participants.

Our results suggest that the twenty-two risks identified in [5] work are present in the participants' projects in our study. We also concluded that the three most common risks are "Failing to perform asynchronous tasks," "Integration with plugins failed," and "Losing connection in HTTP request." Over 70% of the survey participants cited these events.

From the findings of our study, software professionals can gain some technical knowledge. With the most common risks identified, solutions to mitigate them can be obtained in advance to avoid real project problems. This study also identifies the permanent need for research to assess technological risks and the scientific community's search for solutions that minimize these events and contribute to the success of software projects.

In future work, we visualized a case study where researchers follow real projects in software factories and assess technological risks. It is also expected as future work to create a data dictionary with more explanations of each technological risk with examples of situations of how they happen.

5 ACKNOWLEDGEMENTS

We are grateful to [5] for providing us with the dataset of their Survey. We thank the experts for their answers and appreciate the comments and suggestions, which significantly contributed to improving the quality of this publication. We thank the anonymous respondents of our Survey.

REFERENCES

[1] Allan Afuah. 2020. Innovation management-strategies, implementation, and profits. (2020).
[2] John Bowers and Alireza Khorakian. 2014. Integrating risk management in the innovation project. *European Journal of innovation management* (2014).
[3] Saad Yasser Chadli, Ali Idri, José Luis Fernández-Alemán, Joaquín Nicolás Ros, and Ambrosio Toval. 2016. Identifying risks of software project management in Global Software Development: An integrative framework. In *IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*.
[4] Danijela Ćirić, Bojan Lalić, and Danijela Gračanin. 2016. Managing innovation: Are project management methods enemies or allies. *International Journal of Industrial Engineering and Management* 7, 1 (2016), 31–41.
[5] Emanuel Dantas, Ademar Sousa Neto, Dalton Valadares, Mirko Perkusich, Felipe Ramos, Hyggo Almeida, and Angelo Perkusich. 2022. Investigating technological risks and mitigation strategies in software projects. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*. 1527–1535.
[6] Karel De Bakker, Albert Boonstra, and Hans Wortmann. 2010. Does risk management contribute to IT project success? A meta-analysis of empirical evidence. *International Journal of Project Management* (2010).
[7] Anna Małgorzata Deptuła and Ryszard Knosala. 2015. Risk assessment of the innovative projects implementation. *Management and Production Engineering Review* (2015).
[8] Veli-Pekka Eloranta, Kai Koskimies, and Tommi Mikkonen. 2016. Exploring ScrumBut—An empirical study of Scrum anti-patterns. *Information and Software Technology* 74 (2016), 194–203.
[9] Norman Fenton, William Marsh, Martin Neil, Patrick Cates, Simon Forey, and Manesh Tailor. 2004. Making resource decisions for software projects. In *Proceedings. 26th International Conference on Software Engineering*. IEEE, 397–406.
[10] Norman Fenton and Martin Neil. 2018. *Risk assessment and decision analysis with Bayesian networks*. Crc Press.
[11] Iris Figalist, Christoph Elsner, Jan Bosch, and Helena Holmström Olsson. 2020. Breaking the Vicious Circle: Why AI for software analytics and business intelligence does not take off in practice. In *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 5–12.
[12] Daya Gupta and Mohd Sadiq. 2008. Software Risk Assessment and Estimation Model. In *2008 International Conference on Computer Science and Information Technology*. 963–967. https://doi.org/10.1109/ICCSIT.2008.184
[13] Wen-Ming Han. 2015. Discriminating risky software project using neural networks. *Computer Standards & Interfaces* 40 (2015), 15–22.
[14] Wen-Ming Han and Sun-Jen Huang. 2007. An empirical analysis of risk components and performance on software projects. *Journal of Systems and Software* 80, 1 (2007), 42–50.
[15] David Hinde. 2018. *PRINCE2 Study Guide: 2017 Update*. John Wiley & Sons.
[16] Chun hsien Wang, Iuan yuan Lu, and Chie bein Chen. 2008. Evaluating firm technological innovation capability under uncertainty. *Technovation* 28, 6 (2008), 349–363. https://doi.org/10.1016/j.technovation.2007.10.007
[17] ISO Central Secretary. 2018. *ISO 31000: risk management–Guidelines*. Standard. International Organization for Standardization, Geneva, CH.
[18] James Jiang and Gary Klein. 2000. Software development risks to project effectiveness. *Journal of Systems and Software* (2000).
[19] Harold Kerzner. 2019. *Innovation Project Management: Methods, Case Studies, and Tools for Managing Innovation Projects*. John Wiley & Sons.
[20] Sukulpat Khumpaisal and Zhen Chen. 2010. Risk assessment in real estate development: an application of analytic network process. *Journal of Architectural/Planning Research and Studies (JARS)* 7, 1 (2010), 103–118.
[21] Chandan Kumar and Dilip Kumar Yadav. 2015. A probabilistic software risk assessment and estimation model for software projects. *Procedia Computer Science* 54 (2015), 353–361.
[22] Andreas R. Köhler and Claudia Som. 2014. Risk preventative innovation strategies for emerging technologies the cases of nano-textiles and smart textiles. *Technovation* 34, 8 (2014), 420–430. https://doi.org/10.1016/j.technovation.2013.07.002 Risk and Uncertainty Management in Technological Innovation.
[23] Johan Linaker, Sardar Muhammad Sulaman, Martin Höst, and Rafael Maiani de Mello. 2015. Guidelines for conducting surveys in software engineering v. 1.1. *Lund University* 50 (2015).
[24] Slobodan B Malbašić and Stefan V Đurić. 2019. Risk assessment framework: Application of Bayesian Belief Networks in an ammunition delaboration project. *Vojnotehnicki glasnik/Military Technical Courier* 67, 3 (2019), 614–641.
[25] Jhon Masso, Francisco J Pino, César Pardo, Félix García, and Mario Piattini. 2020. Risk management in the software life cycle: A systematic literature review. *Computer Standards & Interfaces* (2020), 103431.
[26] Júlio Menezes, Cristine Gusmão, and Hermano Moura. 2019. Risk factors in software development projects: a systematic literature review. *Software Quality Journal* 27, 3 (2019), 1149–1174.
[27] Wang Min, Zhang Jun, and Zhang Wei. 2017. The application of fuzzy comprehensive evaluation method in the software project risk assessment. In *Proceedings of the 2017 International Conference on Management Engineering, Software Engineering and Service Sciences*. 76–79.
[28] Rogério Feroldi Miorando, José Luis Duarte Ribeiro, and Marcelo Nogueira Cortimiglia. 2014. An economic–probabilistic model for risk analysis in technological innovation projects. *Technovation* 34, 8 (2014), 485–498. https://doi.org/10.1016/j.technovation.2014.01.002 Risk and Uncertainty Management in Technological Innovation.
[29] CMMI What Is Capability Maturity Model. 2017. Integration (CMMI)®?| CMMI Institute. *CMMI Institute* (2017).
[30] Mirko Perkusich, Lenardo Chaves e Silva, Alexandre Costa, Felipe Ramos, Renata Saraiva, Arthur Freire, Ednaldo Dilorenzo, Emanuel Dantas, Danilo Santos, Kyller Gorgônio, et al. 2020. Intelligent software engineering in the context of agile software development: A systematic literature review. *Information and Software Technology* 119 (2020), 106241.
[31] J. Petronijevic, A. Etienne, A. Siadat, and S. Bassetto. 2019. Operational Framework for Managing Risk Interactions in Product Development Projects. In *International Conference on Industrial Engineering and Systems Management (IESM)*. 1–6.
[32] PMI. 2019. *A Guide to the Project Management Body of Knowledge (PMBOK Guide)*. Vol. 6. EUA.
[33] Teade Punter, Marcus Ciolkowski, Bernd Freimut, and Isabel John. 2003. Conducting on-line surveys in software engineering. In *2003 International Symposium on Empirical Software Engineering, 2003. ISESE 2003. Proceedings.* IEEE, 80–88.
[34] Robert S Russell and Bernard W Taylor-Iii. 2008. *Operations management along the supply chain*. John Wiley & Sons.
[35] Ashish B Sasankar and Vinay Chavan. 2011. SWOT analysis of software development process models. *International Journal of Computer Science Issues (IJCSI)* 8, 5 (2011), 390.
[36] Tim Storer. 2017. Bridging the chasm: A survey of software engineering practice in scientific programming. *ACM Computing Surveys (CSUR)* 50, 4 (2017), 1–32.
[37] Jeff Sutherland and Ken Schwaber. 2013. The Scrum Guide. http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf. Acessado em: 01-06-2020.
[38] Yasunari Takagi, Osamu Mizuno, and Tohru Kikuno. 2005. An empirical approach to characterizing risky software projects based on logistic regression analysis. *Empirical Software Engineering* 10, 4 (2005), 495–515.
[39] Linda Wallace and Mark Keil. 2004. Software project risks and their effect on outcomes. *Commun. ACM* (2004).
[40] Zhiwei Xu, Taghi M Khoshgoftaar, and Edward B Allen. 2003. Application of fuzzy expert systems in assessing operational risk of software. *Information and software technology* 45, 7 (2003), 373–388.