A Recommender System-Based Approach to Risk Management in Scrum Projects

Ademar Sousa Neto
ademar.sousa@virtus.ufcg.edu.br
Intelligent Software Engineering Group - VIRTUS/UFCG
Campina Grande, PB, Brazil

Mirko Perkusich
mirkoperkusich@virtus.ufcg.edu.br
Intelligent Software Engineering Group - VIRTUS/UFCG
Campina Grande, PB, Brazil

Emanuel Dantas
emanuel.filho@ifpb.edu.br
Federal Institute of Paraíba
Monteiro, PB, Brazil

Felipe Ramos
Alexandre Costa
felipe.ramos@ifpb.edu.br
alexandre.costa@ifpb.edu.br
Federal Institute of Paraíba
Santa Rita, PB, Brazil

Hyggo Almeida
almeida@virtus.ufcg.edu.br
Intelligent Software Engineering Group - VIRTUS/UFCG
Campina Grande, PB, Brazil

Angelo Perkusich
perkusich@virtus.ufcg.edu.br
Intelligent Software Engineering Group - VIRTUS/UFCG
Campina Grande, PB, Brazil

ABSTRACT

Risk management is essential in software project management. It includes activities such as identifying, measuring, and monitoring risks. The increasingly popular agile methods don't offer specific activities to manage risk. The lack of risk management or its inadequate application is one of the reasons for the failure of software development projects. Therefore, we developed an approach to risk management in software development projects that use Scrum. The proposed approach provides a set of risk management practices and an iterative life cycle. Along with this approach, we developed a recommendation algorithm to assist decision-making when identifying risks. Thus, we performed an offline evaluation to verify the best configuration for the recommendation algorithm that will accompany our approach. We chose Manhattan similarity based on the experimental results collected, with a precision of 45%, recall of 90%, and F1-score of 58%. So it is possible to observe that the recommender system can perform risk predictions satisfactorily. Therefore, it is promising to assist in decision-making in Scrum-based projects.

CCS CONCEPTS

• **Software and its engineering → Risk management**; • **Computing methodologies → Artificial intelligence**.

KEYWORDS

Risk Management, Project management, Recommendation System, SCRUM

1 INTRODUCTION

A Risk is an uncertain event or condition that, if it happens, will have a positive or negative effect on at least one of the project's objectives [7, 19]. The ISO 31000 standard presents risk as the effect of uncertainty on achieving objectives [20]. Risk management is applying skills and knowledge to reduce threats to an acceptable level and maximize opportunities [18, 27].

There are several techniques and tools proposed to support risk management. For instance, graphic methods as cause and effect diagrams [34], SWOT matrix [35], or Intelligent techniques [29], which include Artificial Intelligence and Data Analytics. The techniques assist risk management activities in identifying, measuring, or monitoring these events [12].

In specific, agile methods don't have explicit techniques to manage risks in projects [37]. Agile practitioners perform risk management in an ad hoc way [41]. The lack of approaches to managing risk in agile projects can contribute to the failure of projects [15, 24, 30, 42]. A systematic approach to risk management can potentially reduce uncertainty and increase the chances of success in software projects [24].

This way, we developed an approach to risk management in software development projects. The proposed approach provides a set of risk management practices and an iterative life cycle. Along with this approach, we developed a recommendation algorithm to assist decision-making when identifying risks. To validate our research, we used data from scrum projects and performed an offline evaluation to find a better configuration of the recommendation algorithm.

We organize the rest of the article as follows: Section 2 summarizes risk management in agile projects. Section 3 presents the proposed approach. Section 4 describes the recommendation system evaluation that the approach brings. Finally, Section 5 discusses our concluding remarks.

2 RISK MANAGEMENT IN AGILE PROJECTS

Risk management is a popular topic in several industry guidelines or standards. The Project Management Body of Knowledge (P.M.B.O.K.) guide defines seven processes for risk management [19], with activities from planning to risk monitoring. On the other hand, the ISO 31000 standard presents principles and recommendations in five activities that address risks [20]. We also found good practices and processes for managing risk in CMMI [11] and PRINCE2 [7].

ISE '22, October 04, 2022, Virtual form

Moran [25] states that explicit risk management in agile projects is essential for:

- increase the team's level of awareness of which factors can harm project results and act proactively to address them;
- being able to propose risk response strategies taking into account their level of exposure;
- increase the level of awareness and engagement of the team in relation to risk-related activities;
- increase the level of awareness and engagement of the team about risk-related activities;
- understand the outcome of risk response actions and assess whether they were effective.

Risks in software projects may present a threat to project success [39]. It's essential to identify the risk and assess the probability of occurring and its possible impact on the project [26]. Risk management applied to software projects may improve your results [28]. Software projects typically use agile methods for software management and development [21].

Agile methods, traditionally, do not use any intentional risk management approach. However, they state that risks are managed by doing multiple iterations or Sprints [6]. Feedback mechanisms present in agile methods reduce (negative) risk by making information available promptly and thus reducing uncertainty, increasing transparency, and improving communication. Still, there is little guidance on how these mechanisms can be effectively implemented [38]. According to Albadarneh et al. [3], risk management isn't addressed explicitly in the Scrum method. But, we found good practices to minimize these events in Scrum Guide [36].

3 PROPOSED SOLUTION

This paper proposes an approach to managing risk in Scrum projects. Furthermore, we propose a recommendation algorithm based on knowledge of past projects to identify risk and help Scrum practitioners to make decisions regarding risk identification. This study covers stages: Structuring project profiles, Scrum Instrumentation and Risk Recommendation.

3.1 Structuring Project Profiles

The first step was to define projects characteristics essential to identify risks in agile projects. We base the proposed structuring approach on attributing characteristics to projects, aiming to provide a mechanism to compare projects to operationalize the recommendation system. Figure 1 shows an overview of the methodology applied for this study stage.

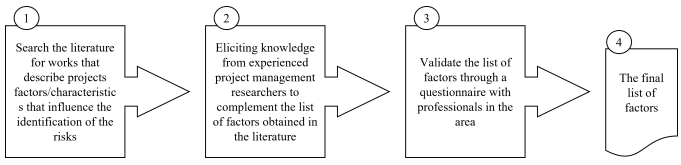

Figure 1: Methodology to define the characteristics

After searching the Literature, we elicited the knowledge of three researchers with experience in project management. Additionally, a questionnaire was used as an instrument to collect data from professionals in the area, aiming to validate and complement the knowledge of the researchers requested. Eight professionals responded: six Project Managers, a Scrum Master, and a Technical Leader.

To structure the project profiles, first, after analyzing the literature that addresses the topic, three relevant factors were identified [14] [23]: Application Domain, Used Technology, and Application Type. Subsequently, we confirmed the three factors identified in the literature with the help of three researchers from the ISE/VIRTUS group, experienced in project management. The experienced researchers elicited two more: Project Duration and Team Experience.

Finally, we evaluated the choice of the five factors in the questionnaire. Each participant could choose more than one factor from the five previously elicited in the literature and by the three researchers experienced in project management and optionally inform new factors not listed.

3.2 Risk Management for Scrum projects

Scrum is not a process, technique, or a definitive method [33]. Instead, it is a values and principles-based framework with events, roles, artifacts, and rules [33], within which you can employ various processes or techniques [36]. Therefore, we understand that the methodology is receptive to adding other processes.

To structure risk-related project information and create and maintain a structured risk base, we propose to add new elements to the Scrum framework. First, we propose a taxonomy for risk registration. Following the same methodology applied in the previous step, we identified the relevant information to compose a risk record after analyzing the literature. Thus, we identified 11 relevant pieces of information [17] [8] [4]: Description, Creation Date, Occurrence Probability, Impact, Team Member, Category, Priority, Status, Action, Sprint, Product Backlog Item. Subsequently, these 11 relevant pieces of information were confirmed with the three researchers experienced in project management without adding any further information. Finally, using a questionnaire, the professionals could evaluate these 11 relevant pieces of information.

The Scrum methodology has a series of events, also called rites or, simply, the Scrum process, according to Tanner and Mackinnon [40], which aim to create regularity in the process. Thus, to manage risks based on past knowledge and to assist Scrum practitioners in identifying risks, we propose some activities, roles, and artifacts for risk management within your process.

The proposed artifacts are an Organization Database and the Risk Register of the Target Project. The Organization Database contains all the organization's projects with their respective risk-related information. The Target Project Risk Register includes information on the risks recorded for the project in question that is under development.

The proposed roles are Risk Manager and Risk Supervisor. The Risk Manager is responsible for creating, modifying, and deleting risks in the corporation's risk memory. In addition, he is responsible for identifying risks that have the potential to be reused by other projects, making use of the recommendation system to assist him in decision making. The Manager is also responsible for adjusting the risks' language to store them in the corporate base

and make them reusable. The Risk Supervisor is responsible for monitoring risks during the sprint execution. That is, this role is responsible for analyzing, evaluating, implementing, and recording the risk events that happened throughout the sprint and updating the Target Project Risk Register. The suggestion is that these roles be performed by different people so that it is another filter to keep the database always consistent, but it does not prevent them from being performed by the same person.

The proposed activities include Risk Identification, Risks and Strategies Assessment, and Risks Critical Analysis. Tables 1, 2 and 3 present the activities.

Thus, this approach consists of providing risk management by the Scrum team, identifying them by reusing the information stored in the Organization's Database. Figure 2 illustrates risk management in the Scrum methodology. Note that the team reuses not only the risk itself but also the response strategies used by past projects.

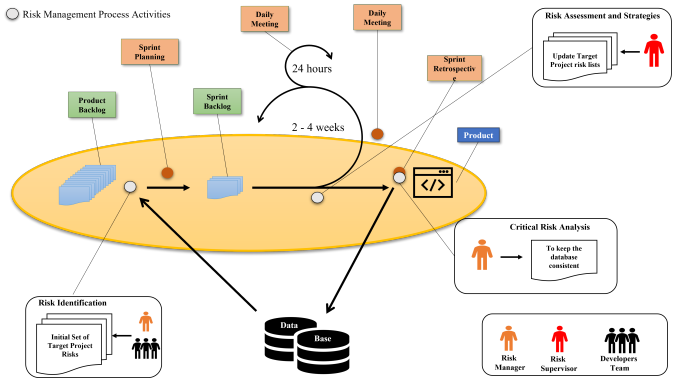

Figure 2: Risk Management in Scrum

3.3 Risk Recommendation

With the possibility to help Scrum practitioners identify risks during risk management and retrieve project history information, the next step was to design the recommendation system based on knowledge of past projects. The Risk Management approach for Scrum Projects, aligned with the Structuring of Project Profiles, enables the operationalization of the recommendation system.

Thus, adapting the generic definition of recommendation problems presented by Adomavicius and Tuzhilin [2] for this work, we have: let P be the set of Projects and R be the set of all risks that the algorithm may recommend. Let u be the utility function of a risk r for a project p, then, for each Project $p \in P$, we intend to recommend the risks $r' \in R$ that maximize the utility of p. Formally, the problem can be represented by Equation 1 [2]:

$$\forall p \in P, r'_p = \arg\max_{r \in R} u(p, r). \quad (1)$$

Since the recommendation system aims to provide as many risks as possible according to the defined objectives, we use the approach of finding all good items [31], that is, providing all possible risks that the user can reuse. The idea is that when starting the project in the agile method, the user provides the characteristics of the target project so the recommendation system can offer possible risks that can be selected and analyzed. Thus, the proposed recommendation system is composed of the following components: Data Collector, Data Transformer, and Recommender.

3.3.1 Data Collector. Data collection consists of extracting information from data sources to represent the recommender system elements to enable the creation of profiles. In this research, the data sources are the structured artifacts and the recommender system elements represented by project characteristics and risks.

In the proposed recommender system, the Data Collector component performs the task of collecting information. The algorithm extracts the information used to represent the characteristics of the structured projects. More specifically, project characteristics are collected by referring to five factors that represent the structured profile of the project to which the risks belong, and they are: Application Domain, Used Technology, Application Type, Project Duration, and Team Experience. Risk information comprises data from the structured model of the risk register.

3.3.2 Data Transformer. Once defined the Project Characteristics data generating the Project profile, the data transformer is responsible for receiving the profiles returned by the data collector as input and generating the characteristic vectors for the Target Project. That is the Project that the algorithm will provide the recommendations to allow the definition of proximity between the Projects.

The Data Collector retrieves Projects from the database and then uses the Target Project to transform the returned data into vectors with similar characteristics.

Thus, the algorithm defines the characteristics vectors by comparing the project profile data between the Target Project and the selected projects using the one hot encoding [9] technique, meaning that the characteristic values of the selected projects are equal to those of the selected projects. Characteristics of the Target Project receive the value of 1, and the values of the selected projects' characteristics differ from those of the Target Project receive the value of 0. With this, we make the selected projects with greater profile data equal to the Target Project, present more excellent proximity, and have their recommended risks.

3.3.3 Recommender. The recommender component performs the generation of the risk recommendation. In this research, collaborative filtering was used [13]. In this approach, we consider the recommendation of risks related to the closest neighbors of the Target Project. The similarity of their characteristics gives the proximity between the projects.

The recommender's role is to perform the K nearest neighbors calculation, using the K Nearest Neighbors (KNN) approach [13], which has advantages for the construction of recommender systems such as [5]: one of the simplest machine learning algorithms; doesn't require learning and maintaining models; can adapt to rapid changes in the classification matrix of recommender system elements. For calculation purposes, we used several distance metrics to compare performance; they are: Cosseno [2], Euclidiana [1], Manhattan [1], Jaccard [5] Chebyshev [22].

Given the distances between the target project p_a and the projects returned by the data collector p'_i, the algorithm calculates the similarity values according to Equation 2, normalizing the results of the

Table 1: Activity Risk Identification

Activity	Risk Identification
When? (Session)	First Sprint Planning (recommended) and during the Sprint execution.
As? (Detail)	First Sprint Planning, the team can identify an initial set of risks for the target project, creating the Risk Register for the Target Project. For this activity, the manager uses the recommendation system to assist in decision-making in identifying risks for the target project. During sprint execution: The Target Project Risk Register can be updated at any time by the Scrum team, with support from the Risk Manager.
Input	Target project characteristics (for the recommender system) and Tacit Knowledge of the professionals.
Output	Target Project Risk Register with elicited risks and possible mitigations.
Responsible Person	Risk Manager and the Team.

Table 2: Activity Risk Assessment and Strategies

Activity	Risk Assessment and Strategies
When? (Session)	The Risk Supervisor must facilitate this meeting whenever they deem it necessary.
As? (Detail)	Analyze and Evaluate such risks, probabilities, impacts, and mitigation and response plans. If necessary, update the risk register template of the target project (i.e., change the status of risks already identified or add new risks). Implement a Mitigation (if convenient) and Response plan (if the risk materializes), referring to the risks raised. Consult and monitor the risks raised, evaluating whether the analysis is valid and whether the mitigation and response plans remain correct. If any risk has materialized, assess its impact on the delivery of the product and record it in the project risk register.
Input	Target Project Risk Register
Output	Target Project Risk Register (Udated)
Responsible Person	Risk Supervisor

Table 3: Activity Critical Risk Analysis

Activity	Critical Risk Analysis
When? (Session)	Sprint Retrospective (Recommended)
As? (Detail)	It consists of the effort to maintain the quality of the corporate database. At the end of each sprint, when there are new risk records, the Risk Manager must validate them and save them in the corporate risk memory to ensure that the language is adequate and the information is consistent.
Input	Target Project Risk Register
Output	Updated Organization Database
Responsible Person	Risk Manager

distance in a closed interval of [0,1]. The K projects with the highest similarity value have their risks selected for recommendation.

$$sim(p_a, p') = 1 - \frac{\sum_{i=1}^{n} \left| p_{ai} - p'_i \right|}{n}. \quad (2)$$

4 EVALUATION

We perform an off-line evaluation of the recommender system. This experiment aimed to identify which recommender system configuration is more effective in generating risk recommendations.

Off-line evaluations allow the evaluation of large numbers of algorithms and their configurations and provide evidence of how the performance of each one reflects in the database used [31]. The main objective of the off-line evaluation is to discard algorithms and configurations with low performance or accuracy, leaving only a small set of candidate solutions for running tests with real users, which require a higher cost to run.

Thus, a widely used approach for evaluating recommender systems is cross-validation, were dividing the training and testing data into partitions varying in several iterations and making these variations avoid the high specialization of the algorithm in a fixed test base. This approach specializes in $K - fold$ cross-validation, which divides the complete database into K partitions and uses $K1$ to train the algorithm. In contrast, it uses the remaining partition for testing the recommendations [32].

In this research, we decided to evaluate the recommendation of test cases using 17-fold cross-validation, reducing the training data's bias by separating it into folds. With K equal to 17, we enable a training data mass (90%) very similar to the total data mass while decreasing the overlap between the training data masses. Therefore, we performed a preliminary evaluation collecting precision, recall, and F1-score (β) in this work.

4.1 Database definition

We collected data on project characteristics of the Center for Research, Development, and Innovation in Information Technology, Communication, and Automation (VIRTUS/UFCG), which uses Scrum. Altogether, we collected characteristics of 17 research and development projects in the software area and associated the projects with defined risks for use and recommendation. The purpose of this database is to run a cross-validation simulation to evaluate the functioning of the code generated for validation, thus allowing bugs before the main validation. For the synthetic database, we register projects with their characteristics and risks related to these projects.

4.2 Experimental Evaluation

In the assessment, we are only interested in binary scores, that is, whether the risk was selected for Project (1) or not (0). Thus, when returning a list of suggestions to the user, we have the following possible results: False-Positive, that is, an item that was recommended but is not accepted; True-Positive, which is an item that has been recommended and is accepted; False-Negative, corresponding to an item that would be accepted but not recommended; and the True-Negative, corresponding to an item that was not recommended and, if it were, wouldn't be accepted [16].

We develop the recommender system based on the k-NN ranking algorithm. The main purpose of offline validation is to examine which algorithm configuration is most effective in recommending risks.

4.2.1 Results. The precision, recall and F1-score results can be seen in Figures 3, 4 and 5, respectively. By analyzing the graphs, you observe an increase in the recall value and a reduction in the precision values as the number of k neighbors increases, regardless of which similarity measure is used. There was no variation in the results for the Manhattan, Euclidean, and Cosine similarity/distance measures. Possibly, this is because these measures have similar behavior with binary vectors [5] [10]. This way, there is no variation in the recommended risk lists or the calculated effectiveness metrics. However, the Jaccard measure is based on the differences between the feature vectors (dissimilarity) [10], so it showed a slight difference.

In Figure 5, resulting from the F1-score metric, you can see an increase in the F1-score results for all distances and k values greater than 1. However, to identify which treatment obtained the best result, we performed the Friedman test to verify if there would be a difference in the different configurations of the risk recommender system in relation to the F1-score metric.

When we identify once the difference between the studied groups, we performed a posthoc test using the Friedman method, which compares the different treatments. As it is impossible to say which one of them presents the best result, we evaluated a tie-breaking

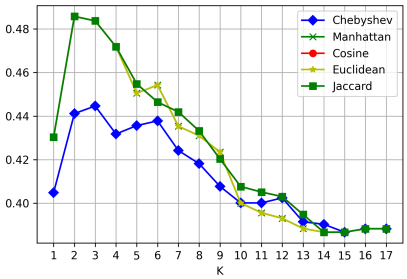

Figure 3: Precision results

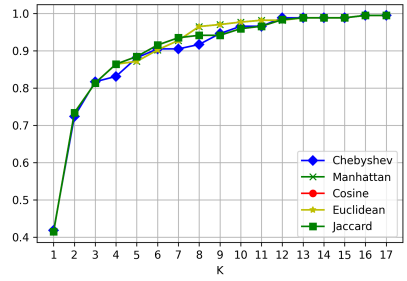

Figure 4: Recall results

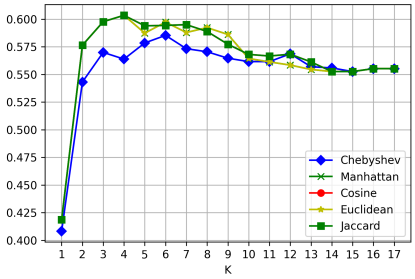

Figure 5: F1-score results

criterion, which refers to the similarity/distance measure type. We analyze the four measures' characteristics to identify the one best suited to the problem domain of the proposed recommendation system.

Thus, we opted for the Manhattan distance, which calculates the distance between two vectors of size d through the sum of their differences. Therefore, as the characteristic vectors considered in the proposed solution can only assume values equal to 0 (zero) or 1 (one), the calculation of the distance between them is given by counting the characteristics of the target Project that are not present in the Project profile neighbor candidate.

Therefore, we chose these settings, along with their results that we obtained in the experiment: Manhattan similarity; number of neighbors (k): 6; precision: 45%; recall: 90%; and F1-score: 58%

5 FINAL REMARKS

This article presents an approach to managing risk in Scrum projects. We base our approach on knowledge management, where creating and maintaining a database on organizational risks is possible.

In addition, it has a recommendation algorithm to support practitioners in identifying risks in Scrum projects. We evaluated the feasibility of our recommendation algorithm. We analyzed which configuration we achieved better with the data collected from agile projects running on [hidden] using the 17-fold cross-validation technique based on the F1 score. The configuration that presented the best results was the Manhattan similarity algorithm, with six neighbors—having an accuracy of 45%, a recall of 90%, and an F1 score of 58%.

In the future, we intend to conduct a case study to validate the solution in real projects and evaluate the ease of use and the perceived usefulness of the recommendation process.

REFERENCES

[1] Gediminas Adomavicius, Nikos Manouselis, and YoungOk Kwon. 2011. Multicriteria recommender systems. In *Recommender systems handbook*. Springer, Boston, MA, 769–803.

[2] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering* 17, 6 (2005).

[3] Aalaa Albadarneh, Israa Albadarneh, and Abdallah Qusef. 2015. Risk management in Agile software development: A comparative study. In *2015 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*. IEEE, IEEE, Jordan, 1–6.

[4] Eman Talal Alharbi and M Rizwan Jameel Qureshi. 2014. Implementation of risk management with SCRUM to achieve CMMI requirements. *International Journal of Computer Network and Information Security* 6, 11 (2014), 20.

[5] Xavier Amatriain, Alejandro Jaimes, Nuria Oliver, and Josep M Pujol. 2011. Data mining methods for recommender systems. In *Recommender systems handbook*. Springer, Springer, 39–71.

[6] Vitor Anes, António Abreu, and Ricardo Santos. 2020. A new risk assessment approach for agile projects. In *2020 International Young Engineers Forum (YEF-ECE)*. IEEE, IEEE, USA, 67–72.

[7] The Stationery Office (editor) AXELOS. 2018. *Managing Successful Projects with PRINCE2 2017 Edition* (2017 ed.). TSO, Online.

[8] Syrine Chaouch, Asma Mejri, and Sonia Ayachi Ghannouchi. 2019. A framework for risk management in Scrum development process. *Procedia Computer Science* 164 (2019), 187–192.

[9] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. online, USA, 7–10.

[10] Seung-Seok Choi, Sung-Hyuk Cha, and Charles C Tappert. 2010. A survey of binary similarity and distance measures. *JSCI* 8, 1 (2010), 43–48.

[11] Mary Beth Chrissis, Mike Konrad, and Sandra Shrum. 2011. *CMMI for development: guidelines for process integration and product improvement*. Pearson Education.

[12] Emanuel Dantas, Ademar Sousa Neto, Mirko Perkusich, Hyggo Almeida, and Angelo Perkusich. 2021. Using Bayesian Networks to Support Managing Technological Risk on Software Projects. In *Anais do I Workshop Brasileiro de Engenharia de Software Inteligente*. SBC, 1–6.

[13] Christian Desrosiers and George Karypis. 2011. A comprehensive survey of neighborhood-based recommendation methods. *Recommender systems handbook* 1, 2 (2011), 107–144.

[14] Alexsandro Filippetto, Robson Lima, and Jorge Barbosa. 2020. Átropos: towards a risk prediction model for software project management. *International Journal of Agile Systems and Management* 13, 3 (2020), 296–314.

[15] Benjamin Gold and Clive Vassell. 2015. Using risk management to balance agile methods: A study of the Scrum process. In *2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)*. IEEE, online, USA, 49–54.

[16] Asela Gunawardana and Guy Shani. 2009. A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research* 10, 12 (2009), 10 pages.

[17] Muhammad Hammad and Irum Inayat. 2018. Integrating risk management in scrum framework. In *2018 International Conference on Frontiers of Information Technology (FIT)*. IEEE, USA, online, 158–163.

[18] Kim Heldman. 2005. *Project manager's spotlight on risk management*. John Wiley & Sons, USA.

[19] Project Management Institute. 2021. *A Guide to the Project Management Body of Knowledge (PMBOK Guide) – and the Standard for Project Management* (7 ed.). Project Management Institute, USA.

[20] ISO. 2018. *Norma Internacional ISO 31000:2018*. ISO, USA.

[21] Magne Jørgensen. 2018. Do agile methods work for large software projects?. In *International Conference on Agile Software Development*. Springer, 179–190.

[22] Giuseppe Jurman, Samantha Riccadonna, Roberto Visintainer, and Cesare Furlanello. 2009. Canberra distance on ranked lists. In *Proceedings of advances in ranking NIPS 09 workshop*. Citeseer, USA, Citeseer, 22–27.

[23] Sana Khan. 2009. An approach to facilitate software risk identification. In *2009 2nd International Conference on Computer, Control and Communication*. IEEE, IEEE, USA, 1–5.

[24] Luanna Lopes Lobato, Thiago Jabur Bittar, Paulo Anselmo da Mota Silveira Neto, Ivan Do Carmo Machado, Eduardo Santana De Almeida, and SILVIO ROMERO DE LEMOS MEIRA. 2013. Risk management in software product line engineering: a mapping study. *International Journal of Software Engineering and Knowledge Engineering* 23, 04 (2013), 523–558.

[25] Alan Moran. 2014. Agile risk management. In *Agile Risk Management*. Springer, USA, 33–60.

[26] Stefanut Morcov, Liliane Pintelon, and Rob J Kusters. 2020. Definitions, characteristics and measures of IT project complexity-a systematic literature review. *International Journal of Information Systems and Project Management* 8, 2 (2020).

[27] Cinzia Muriana and Giovanni Vizzini. 2017. Project risk management: A deterministic quantitative technique for assessment and mitigation. *IJPM* 35, 3 (2017).

[28] Per Rådberg Nagbøl, Oliver Müller, and Oliver Krancher. 2021. Designing a risk assessment tool for artificial intelligence systems. In *International Conference on Design Science Research in Information Systems and Technology*. Springer, Springer, USA, 328–339.

[29] Mirko Perkusich, Lenardo Chaves e Silva, Alexandre Costa, Felipe Ramos, Renata Saraiva, Arthur Freire, Ednaldo Dilorenzo, Emanuel Dantas, Danilo Santos, Kyller Gorgônio, Hyggo Almeida, and Angelo Perkusich. 2020. Intelligent software engineering in the context of agile software development: A systematic literature review. *Information and Software Technology* 119 (2020), 106241. https://doi.org/10.1016/j.infsof.2019.106241

[30] Muhammad Akil Rafeek, Adila Firdaus Arbain, and Endah Sudarmilah. 2019. Risk mitigation techniques in agile development processes. *Int. J. Supply Chain Manag* 8, 2 (2019), 1123–1129.

[31] F Ricii, L Rokach, B Shapira, and PB Kantor. 2011. Recommender System Handbook.

[32] Kenneth A. Ross. 2009. *Cache-Conscious Query Processing*. Springer US, Boston, MA, 301–304. https://doi.org/10.1007/978-0-387-39940-9_658

[33] Kenneth S Rubin. 2012. *Essential Scrum: A practical guide to the most popular Agile process*. Addison-Wesley, USA.

[34] Robert S Russell and Bernard W Taylor-Iii. 2008. *Operations management along the supply chain* (1a ed.). John Wiley & Sons. ISBN 978-0470233795.

[35] Ashish B Sasankar and Vinay Chavan. 2011. SWOT analysis of software development process models. *IJCSI* 8, 5 (2011), 390.

[36] Ken Schwaber and Jeff Sutherland. 2020. The Scrum Guide. *URL: https://gomyskills.com/wp-content/uploads/2021/01/2020-Scrum-Guide-US.pdf* 2, 2 (2020), 20 pages.

[37] Pedro Serrador and Jeffrey K. Pinto. 2015. Does Agile work? — A quantitative analysis of agile project success. *International Journal of Project Management* 33, 5 (2015), 1040–1051. https://doi.org/10.1016/j.ijproman.2015.01.006

[38] Xiaoyuan Su and Taghi M Khoshgoftaar. 2006. Collaborative filtering for multi-class data using belief nets algorithms. In *2006 18th IEEE international conference on Tools with Artificial Intelligence (ICTAI'06)*. IEEE, IEEE, USA, 497–504.

[39] Carlos Tam, Eduardo Jóia da Costa Moura, Tiago Oliveira, and João Varajão. 2020. The factors influencing the success of on-going agile software development projects. *International Journal of Project Management* 38, 3 (2020), 165–176.

[40] Maureen Tanner and Angela Mackinnon. 2015. Sources of interruptions experienced during a scrum sprint. *Electronic Journal of Information Systems Evaluation* 18, 1 (2015), pp3–18.

[41] Breno Gontijo Tavares, Carlos Eduardo Sanches da Silva, and Adler Diniz de Souza. 2019. Risk management analysis in Scrum software projects. *International Transactions in Operational Research* 26, 5 (2019), 1884–1905.

[42] Juliane Teller, Alexander Kock, and Hans Georg Gemünden. 2014. Risk management in project portfolios is more than managing project risks: A contingency perspective on risk management. *Project Management Journal* 45, 4 (2014), 67–80.