Bulletin of Electrical Engineering and Informatics
Vol. 9, No. 3, June 2020, pp. 1299~1310
ISSN: 2302-9285, DOI: 10.11591/eei.v9i3.1905

A risk management framework for distributed scrum using PRINCE2 methodology

Mohammad Esteki[1], Taghi Javdani Gandomani[2], Hadi Khosravi Farsani[3]
[1]Department of Computer Engineering Isfahan (Khorasgan) Branch, Islamic Azad University Isfahan, Iran
[2,3]Department of Computer Science, Shahrekord University, Iran

Article Info

Article history:

Received Nov 16, 2019
Revised Jan 8, 2020
Accepted Feb 19, 2020

Keywords:

Agile software development
Distributed software development
PRINCE2
Risk management
Scrum

ABSTRACT

The distributed Agile development approach has been accepted by software companies due to its promised benefits. However, due to the controversial nature of distributed and Agile development, significant challenges arise from spatial, temporal, social, and cultural differences between distributed teams. Scrum, as the most popular Agile methodology, assumes that team members work together in the same room. But this principle does not apply in a realistic scenario where Scrum teams are distributed in different locations. Hence, proposing a risk management framework is necessary in order to succeed such teams. The purpose of this research was to propose a risk management framework in Scrum using the PRINCE2 methodology, which includes the perceived risks in distributed Scrum projects and their causes and roots for managing these risks. By embedding distributed Scrum in delivery layer of PRINCE2 and considering perceived risk factors, along with a hybrid model, a risk management framework was suggested. This framework has been used in a case study, and the results showed its proper functionality in detecting and eliminating potential risks in the case under study. Also, using this framework led to higher team efficiency in terms of increasing the number of completed user stories in each sprint.

Corresponding Author:

Taghi Javdani Gandomani,
Department of Computer Science, Shahrkord University,
Rahbar Boulevard, 64165478, Shahrekord, Iran.
Email: javdani@sku.ac.ir

1. INTRODUCTION

Risk management in software development as one of the pillars of software project management has always been a serious concern of researchers. Global experiences have shown that the success of these projects depends largely on proper planning and preparedness to reduce potential risks. This is mostly because software risk management involves approaches, processes, and tools that can reduce potential challenges in the management of risks [1]. This is important due to major changes in software development. Today, organizations use the hybrid approach including Agile software development (ASD) and distributed software development (DSD), which is called distributed Agile development (DAD), to develop better software products with better quality, time, and cost use. This approach helps to get the benefits of Agile and distributed development simultaneously. However, due to some inconsistencies, this approach is subject to significant challenges and risks, which will bring some kind of risk in software development [2]. Scrum, as an Agile popular methodology, is nowadays popular among many software teams in Agile distributed development. But the distribution of project stakeholders in the global software development (GSD) projects with the time period, geography, and culture often results in some of the challenges or risks

that may affect the processes of communication and collaboration that have been emphasized in this methodology [3].

Several studies have been conducted on risk management in Agile and DAD. Some of them merely point out the need to pay attention to risk management without specifying a particular agile method [4-6]. Others have focused on approaches and project management standards, such as PMBOK and PRINCE2, to provide risk management strategies for agile development in general or in a particular methodology [7-10]. Although the analysis of these studies shows that there are opportunities to provide more effective solutions in this field. In particular, the appropriate strategy and framework for efficient risk management in Scrum seems to be a serious necessity in DAD.

By focusing on the mentioned challenge, this research has tried to provide a PRINCE2-based risk management framework in distributed Scrum and proves its effectiveness and operational effectiveness in practice. The rest of this paper is organized as follows; Section 2 introduces Scrum and distributed Scrum. Section 3 outlines the PRINCE2 methodology and then section 4 describes the related work. Section 5 describes the research method adopted in this study, and then, in section 6, the proposed framework is introduced. Section 7 describes the results of using the framework in the case study, and finally, in section 8, conclusions and future work are expressed.

2. SCRUM AND DISTRIBUTED SCRUM

Scrum is an iterative and incremental framework for product management. In fact, it is a flexible and comprehensive development strategy in which the development team works as a unit to achieve a common goal, challenging the traditional approach for product development. It also enables the team to be organized through its daily communications. The different stages of Scrum are provided in Figure 1 [11]. Scrum, with a focus on project management in development, has only three roles in developing software products, including scrum master, product owner, and member of the development team [12]. Scrum, in theory, is recommended for small and medium-sized teams and small or medium-sized projects, but today it is used in large projects, multiple and non-integrated teams, and multi-site companies [13-16]. In 2013, the concept of LeSS for Scrum was introduced on a large scale with two basic frameworks [17]. The first framework, as shown in Figure 2, is suitable for a single project that has only one owner.

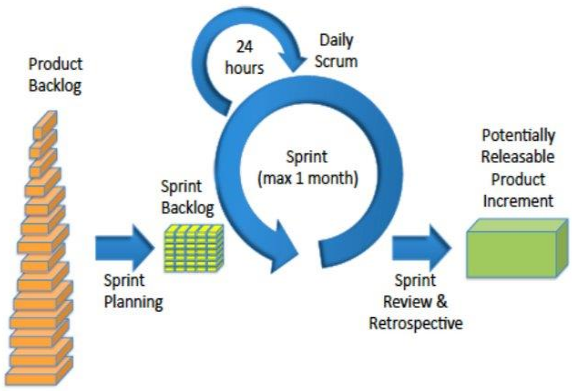

Figure 1. Scrum framework for software development

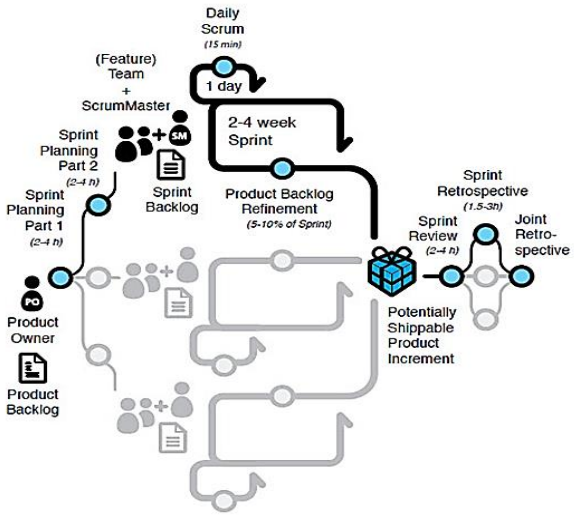

Figure 2. Scrum in large scale (framework 1) [17]

The second framework, as shown in Figure 3, is appropriate for a product that has more than 1,000 people working on several different sites. In this framework, an important role, called area product owner (APO), has been added, whose special task is to prioritize and manage the site of the product backlog. Within this framework, the product backlog is divided into several regions. Each area includes a bunch of customer needs. Between 3 and 10 teams can operate in each area. Along with the benefits of DAD, there are many challenges [18], such as documentation, temporal, spatial, and cultural differences, team distributions, division and distribution of work, and so on, which are the most important factors in DAD.

3. PRINCE2 METHODOLOGY

PRINCE2 is one of the most famous project management methodology used by individuals and organizations in various industries [19]. This method helps to successfully manage the projects, regardless of type or scale, through the requirements of the project. PRINCE2 is built on seven principles, backgrounds, and processes that can fit specific needs. A special version of PRINCE2, called Agile PRINCE2, has been presented in order to take advantage of PRINCE2 in Agile development, which is somehow the most sophisticated agile project management solution [20].

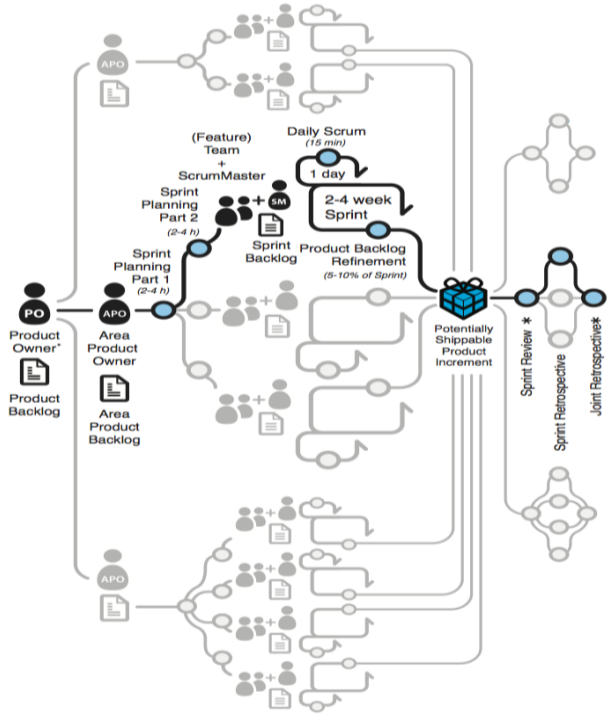

Figure 3. Scrum in large scale (framework 2) [17]

PRINCE2, as shown in Figure 4, consists of four integrated elements inducing principles, themes, processes, and project environment. There are seven principles that a project is not in compliance with PRINCE2 until they are all implemented [20]. The themes in PRINCE2 explain the aspects of project management that need to be addressed in parallel through the project. The themes can be seen in Figure 4. They explain the specific method used in PRINCE2 to manage projects [20]. Processes describe the project life cycle from initial idea to project closure as well as measuring the benefits. Each process provides lists of recommended activities, relevant responsibilities, and guidance on how to adapt to a specific environment [20]. The processes are starting up a project (SU), initiating a project (IP), directing a project (DP), controlling a stage (CS), managing a stage boundary (SB), managing product delivery (MP), and closing a project (CP). The risk management processes in PRINCE2 include five steps: Identify, assess, plan, implement, and communicate.

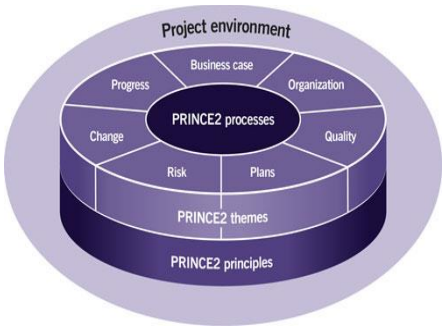

Figure 4. PRINCE2 methodology in project management

4. RELATED WORKS

Literature review shows a few risk management approaches in agile methods. In 2008, an integrated model of risk management and agile processes was proposed. The proposed model consists of three phases including product vision planning, product roadmap, and implementation, in which each phase has its specific risk management. Risk management includes risk identification processes, risk analysis, risk management planning, risk monitoring and control, end of risk and examination after the end of risk [21]. Failures in software projects have serious warning statistics and symptoms, one of which is the lack of risk management in the project management process. According to a report in 2016, during the period from 2004 to 2012, more than 71% of projects were delivered after the due time, and more than 51% of the projects cost more than contracted [22].

In 2009, key challenges in the use of Scrum in distributed environments were identified, as well as the strategies that project managers used to overcome these challenges [23]. Finally, a conceptual framework for identifying and reducing risks in the Scrum-based GSD was presented. This framework includes a general overview of risks, current strategies, and approaches to reduce risks. Several case studies have been reported in this regard, but these studies are limited to one or more software companies. In 2010, by reviewing twelve case studies, thirteen DAD challenges were identified and classified into seven categories. These categories include culture, time zone, communication, trust, customer cooperation, training, and technology [24].

In 2015, a risk management model was proposed for Scrum using the PRINCE2 methodology [25]. The main reason for this combination, as the authors explained, is the process-oriented nature of both methods. That study showed that the most important feature of risk is its severity, and the most important place to talk about risk is Sprint planning. The results of that research show that the inclusion of Scrum in the PRINCE2 methodology leads to improved product delivery time [25]. In 2016, a rating method was proposed based on agile risk scoring to prioritize risk factors in Agile software development. To optimize risk factors, particle swarm optimization has been used as an iterative approach [26]. Organizations combine the DSD with Agile approach to achieve higher quality and, lower cost and time. In 2015, forty-five risk factors were identified in DAD projects and then categorized into five categories including lifecycle, collective awareness, project management, external stakeholder engagement, and technology launch [6]. In 2017, the same researchers developed their work and created a risk management framework for DAD projects. The framework has been implemented partially in three different companies, with satisfactory results showing that most of the risks have been defeated or their impact reduced [9].

In 2017, a study addressed forty solutions for risk management in Scrum projects based on experts' opinions [27]. The four common practices of risk management in Scrum projects relate to aspects of communication, lessons learned, and individual behavior. On the other hand, the four less commonly used methods in Scrum refer to the main responsibility of the SM, the implementation of formal planning, and the use of risk management techniques and artifacts [27]. In 2017, in a research study, software agents were used to perform risk management tasks and collect data from the project environment for risk identification [28]. In this research, a risk management model was introduced in a risk management tool that uses software agents to manager the potential risks [28]. Also, another study aimed to define risk indicators and assess the probability of risk occurrence in Agile software projects in order to identify the potential risk factors [4].

In 2014, risk management researchers used the Scrum to reach the CMMI in the organization, which their proposed solution showing 71.94% of the success. They also predicted that their proposed solution would enable software companies to achieve a desirable CMMI and improve software products of high quality, and they concluded that by implementing risk management in the Scrum, the desired CMMI could be reached. This demonstrates the adaptability of the Scrum to the CMMI standard [29]. Some methodologies formally investigate risks and usually use some informal methods for risk management [30]. However, for software and IT professionals, software project failures are not uncommon. In fact, according to the Chaos report in 2009 from the standards group, only 32% of software projects were successful, which this issue has caused many challenges for industry and academic researchers in software projects to consider many risk factors. Software projects have been analyzed for many risk factors. In a study conducted in the China software industry, more than 31 risk factors have been analyzed in six dimensions [31].

The Scrum and PRINCE2 standards emphasize on risk identification in projects. In the traditional project management methods, less success is due to uncertainty issues. In uncertain environments, the Scrum approach can greatly help to identify risks rather than traditional ones. [32]. In another study, a combination model of DSDM and PRINCE2 was introduced, which issues related to development were handled by DSDM and management issues were carried out by PRINCE2. This study shows better project management dynamics and better control as well [33]. In a study, a five-step approach to agile risk management has been used, and a SWOT matrix has been used to analyze it [34]. To manage risk in software projects, the software team can use a combination of the same methods used for non-software projects. For example, PMBOK includes knowledge domains for the success of high-level projects that can be used independently of the type

of methodology [35]. In another study has proposed a Scrum-based risk management framework, called RBSM that combines risk management processes to improve the Scrum project issues [36].

One of the challenges of agile methods is the lack of regular planning. In a research study, a hybrid XP model and the PRINCE2 were introduced aimed at introducing a systematic, flexible, and agile project management method with agile characteristics [37]. The results indicated that using their proposed method, they presented a real quality software product with a lower cost and time in a real project. Table 1 shows a summary of the most important related studies.

Table 1. Most important related works

Study	Achievement	Limitations
Nyfjord J. 2008 [21]	Proposing an integrated model with ASD And defining processes and roles for risk management	Not applicable in DAD
Hossain E. 2009 [23]	Risk classification Addressing of existing strategies Providing a solution for risk reduction in Scrum	The lack of importance of each class Not paying attention to cultural and managerial issues General classification
Kajko Mattsson 2010 [24]	Focus on classifying DAD issues Attention to various issues such as culture and trust	The lack of importance of each class The overall classification and relevance of DAD
Tomanek M. 2015 [25]	Embedding risk management into Scrum Setting key risk identification meetings Determine the risk characteristics	Not applicable in DAD
Agrawal R. 2016 [26]	Prioritization of risks	Lack of a systematic prioritization process Not applicable in DAD
Shrivastava S. V. 2015 [6]	Identifying 45 risk factors Categorizing risk factors and providing the methods for managing them	Lack of a systematic prioritization process
Shrivastava S. V. 2017 [9]	Risk classification Determine the importance of each area Examine the exact risk factors	Lack of details of the identification process
Tavares B. G. 2017 [27]	Focus on dealing with risks Provide 40 risk management strategies in Scrum	Not paying attention to the source of the risks and identification process
Odzaly E. E. 2017 [28]	Realistic model Support for risk management tools	Lack of providing risk handling process
Rai A. K. 2017 [4]	Estimation of risk occurrence	Not applicable in DAD
Mousaei M. Gandomani T. J. 2018 [8]	Embedding risk management in Scrum Determining meetings to key risk identification Determining risk characteristics Officially risk documentation	Not applicable in DAD

5. RESEARCH METHOD

This paper explored how to use the PRINCE2 methodology to create a risk management framework. Then, the proposed framework was evaluated in a case study.

5.1. Case study

The company under study was a software company working on core banking systems. In the case study, 3 teams were working in different sites. The delivery time of the product was estimated at 150 days and the duration of each Sprint was 2 weeks. The detail of the teams can be seen in Table 2.

Table 2. The detail of the teams in the case study

Team	Size (number)	Average experience in software (years)	Average experience in Agile (years)	Team location	Description
Team 1	8	5.5	2	Company headquarter	-
Team 2	5	5	2	Customer's site/ Bank	Product owner worked 2 days a week
Team 3	6	5	2	Company 2nd building	Product owner worked 2 days a week

6. RISK MANAGEMENT FRAMEWORK

The main idea behind the development of the proposed framework is the nature of PRINCE2 and Scrum methodologies. PRINCE2 is the most commonly used project management method in the world and is increasingly used in conjunction with agile methodologies. Since many companies intend to use agile methods, preparing special guidance on employing PRINCE2 in an Agile environment is necessary, as shown in Figure 5 [38]. Initially, distributed Scrum was combined with PRINCE2 methodology. In other words, Scrum was used in the PRINCE2 delivery layer and a hybrid model was created. Reviewing the literature helped to collect and categorized the risks reported in Scrum projects, as shown in Table 3. Then, these risks

were placed next to the designed framework for ease of prediction. Figure 6 illustrates the risk management framework in Scrum distributed using the PRINCE2 methodology.

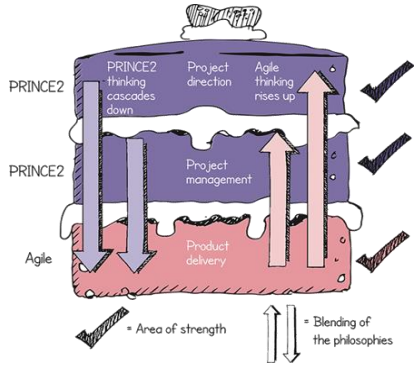

Figure 5. Combining PRINCE2 methodology with agile methods [38]

Table 3. List of risk factors in distributed scrum

R#	Risk factor	Risk area	Risk category	DSD property	Ref.
R1	Higher Interdependency Between the Teams (ISF)	Project Management	Project Organization	Spatial Distance	[9]
R2	Team Recognizing in Every Sprint (ISF)	Project Management	Project Organization	Spatial Distance	[9]
R3	Growth in Team Size or Development Site (ISF)	Project Management	Project Organization	Spatial Distance	[9]
R4	Lower initial Velocity	Project Management	Project Planning and Control	Spatial Distance	[9]
R5	Difficult to Execute Fixed Price Projects	Project Management	Project Initiation	Spatial Distance	[9]
R6	Unavailability of Business Analyst (ISF)	Project Management	Acquisition and Development of Project Teams	Work/Development Culture	[9]
R7	Lack of Uniformity in Multisite Team's Capabilities (ISF)	Project Management	Acquisition and Development of Project Teams	Work/Development Culture	[9]
R8	The Emergence of excessive Competition between Teams	Project Management	Acquisition and Development of Project Teams	Work/Development Culture	[39]
R9	The Emergence of excessive Competition between Scrum Masters/Product Owners	Project Management	Acquisition and Development of Project Teams	Work/Development Culture	[39]
R10	Ineffective Standup Meetings	Software Development Lifecycle	Standards of Agile Ceremonies	Language Barrier	[9]
R11	Difficulty in System Release Management and Development (ISF)	Software Development Lifecycle	Release Management	Temporal Distance	[9]
R12	Losing on Time on End-to-End Extensively Interdependent Transaction Rich Test Cycle across Distributed Teams	Software Development Lifecycle	Test and Integration	Temporal Distance	[9]
R13	Code Integration across Multiple Sites	Software Development Lifecycle	Coding and Integration	Temporal Distance	[9]
R14	Requirements Conflicts amongst Multiple Product Owner	Software Development Lifecycle	Requirement Elicitation	Large Project Scope	[9]
R15	Problem Related to the Priority of the Requirements Because of the Multiplicity of Product Owners	Software Development Lifecycle	Requirement Elicitation	Large Project Scope	[40]
R16	Cross Functional Teams insufficient for testing of Large Projects	Software Development Lifecycle	Test and Integration	Large Project Scope	[9]
R17	Test data management	Software Development Lifecycle	Test and Integration	Large Project Scope	[9]
R18	Inadequate Prioritization of Requirements	Software Development Lifecycle	Requirement Analysis and Specification	Large Project Scope	[9]
R19	Area Requirements Mismatch with Feature Team Capabilities	Software Development Lifecycle	Requirement Analysis and Specification	Large Project Scope	[39]
R20	Requirements unclear to the Team	Software Development Lifecycle	Requirement Elicitation	Spatial Distance	[9]

Table 3. List of risk factors in distributed scrum *(continue)*

R#	Risk factor	Risk area	Risk category	DSD property	Ref.
R21	Inadequate Communication about End User Requirements	Software Development Lifecycle	Requirement Elicitation	Spatial Distance	[9]
R23	Unavailability of Requirement Documents for Testing	Software Development Lifecycle	Test and Integration	Spatial Distance	[9]
R24	Poor Technical Debt Management	Software Development Lifecycle	Coding and Integration	Spatial Distance	[9]
R25	Issues with Pair Programing	Software Development Lifecycle	Coding and Integration	Spatial Distance	[9]
R26	Frequent Architectural Changes (ISF)	Software Development Lifecycle	Design	Spatial Distance	[9]
R27	Inconsistency in Design Standards of Distributed Teams (ISF)	Software Development Lifecycle	Design	Work/Development Culture	[9]
R28	No Common Definition of Done	Software Development Lifecycle	Standards of Agile Ceremonies	Work/Development Culture	[9]
R29	Differences in Agile Practices and Standard of Processes followed by Multiple Teams	Software Development Lifecycle	Standards of Agile Ceremonies	Work/Development Culture	[9]
R30	Lack of Communication between Team and the Client	Group Awareness	Communication	Spatial Distance	[9]
R31	Lack of Communication between the Team Members	Group Awareness	Communication	Spatial Distance	[9]
R32	Under investment on Travel by the Management	Group Awareness	Communication	Spatial Distance	[9]
R33	Lack of Documentation	Group Awareness	Communication	Spatial Distance	[9]
R34	Lack of Face to Face Communication	Group Awareness	Communication	Spatial Distance	[39]
R35	Poor Collaboration between Different Sites	Group Awareness	Coordination and Collaboration	Spatial Distance	[9]
R36	Issue of Coordinating the Members of Scrum Masters and Product Owners Team	Group Awareness	Coordination and Collaboration	Spatial Distance	[39]
R37	Lack of Trust between the Client and Offshore Teams (ISF)	Group Awareness	Trust	Spatial Distance	[9]
R38	Lack of Trust between the Onshore and Offshore Teams (ISF)	Group Awareness	Trust	Spatial Distance	[9]
R39	Lack of Collaboration between Developers and Quality Assurance Members	Group Awareness	Coordination and Collaboration	Work/Development Culture	[9]
R40	Ineffective Scrum of Scrums Meetings	Group Awareness	Coordination and Collaboration	Work/Development Culture	[40]
R41	Poor Coordination between Multiple Teams	Group Awareness	Coordination and Collaboration	Temporal Distance	[9]
R42	Unsuitability of Agile approach for Large Organization	Group Awareness	Communication	Large Project Scope	[9]
R43	Delays and problems in group decision making	Group Awareness	Communication	Large Project Scope	[39]
R44	The Complexity of Communication between teams Due to The Responsibility of Several Teams on the Area Product Owner	Group Awareness	Communication	Large Project Scope	[39]
R45	Uncommon Language	Group Awareness	Communication	Language Barrier	[9]
R46	Lack of Communication Infrastructure (ISF)	Technology Setup	Infrastructure and Resources	Spatial Distance	[9]
R47	Inappropriate Tools Selection (ISF)	Technology Setup	Tools Selection	Work/Development Culture	[9]
R48	Unavailability of Product Owner (ISF)	External Stakeholder Collaboration	Customer Collaboration	Spatial Distance	[9]
R49	Poor Coordination between Multiple Vendors	External Stakeholder Collaboration	Multiple Vendor Involvement	Spatial Distance	[9]
R50	Risk in Code Integration with Multiple Vendors	External Stakeholder Collaboration	Multiple Vendor Involvement	Spatial Distance	[9]
R51	Dependency on Third Party	External Stakeholder Collaboration	Multiple Vendor Involvement	Spatial Distance	[9]
R52	Inappropriate User Story Estimates with Multiple Vendors	External Stakeholder Collaboration	Multiple Vendor Involvement	Work/Development Culture	[9]

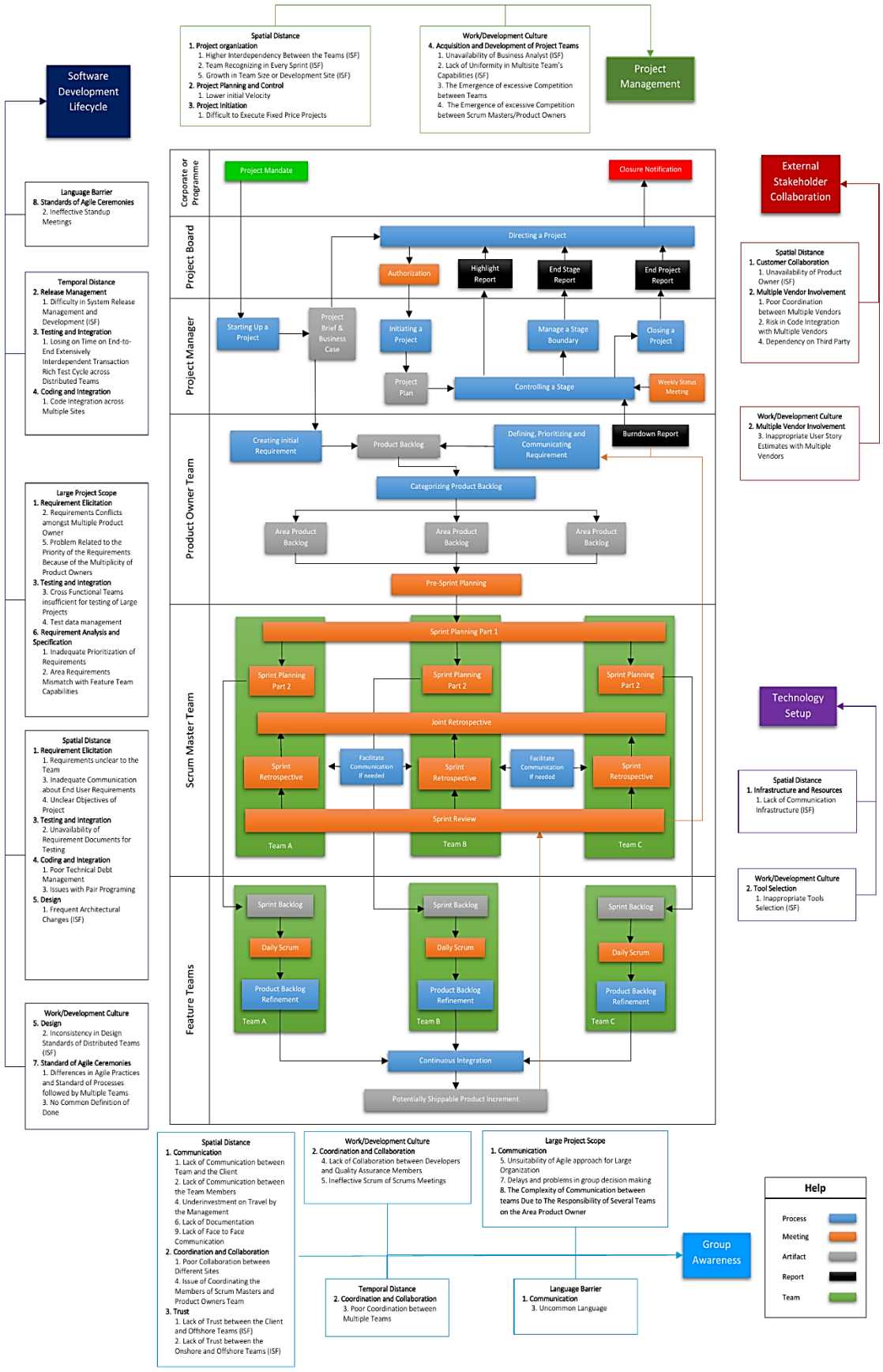

Figure 6. Risk management framework in the distributed scrum

In the first step, the project mandate can identify the responsibilities of individuals, since one of the Scrum weaknesses is a strong dependence on the team members. In the case of leaving the team, it may cause the project to be stopped and broken. However, if defined in the project charter, it can greatly overcome this challenge with the project charter. The project board takes decisions to begin and enforces start-up permissions. The project manager starts the project by putting together the requirements and costs, and the project manager must work with the product owners to evaluate and clarify the project's requirements. When the project starts, the project manager will design steps and details. The project manager must connect the steps with Sprint, then the project enters the preparation phase. One of Scrum's weak points is that the area involved in the project is not yet measurable and cannot be estimated. By this preparation phase, the scope and extent of the project involved can be identified to a certain extent, which, by identifying the scope of the project, can be better off the probable risks of the project to determine. In border management, the risks are recorded and reported, and by the project management, the next steps are carried out.

Each team has a Product Owner who is responsible for managing the requirements. The product owners and the development team discuss the product backlog and requirements that must be delivered to the next sprint. Scrum Masters must also design Sprints together with Project Managers and facilitate their implementation. Product backlogs are also prioritized by the PRODUCT OWNERS and are planned by the Scrum Master on the sprints of each team. The works are delivered to the development team as Sprint backlog, and development teams also publish a version of the product in time periods, which will be followed by the Sprint review meeting. Project Manager after the Sprint review meetings, must prepare a special report on the progress of the project and then submit it to the project board. The product owners and project manager are also explored through the burndown chart and weekly sprint meetings, which can partly identify and verify the project risks.

7. RESULTS AND DISCUSSION

The application of the framework in the case study was carried out by addressing 52 known risks in the literature review and their occurrence in practice. The results of applying it were analyzed by observing and collecting data in the range of 1 to 5 and analyzing them in three desirable categories including favorable (1 to 2.33), relatively favorable (2.34 to 3.66) and unfavorable (3.67 to 5). Figure 7 shows that out of the 75 discovered risks, 42 risks (56%) have been eliminated. Of the 33 risks not eliminated 9 risks (12%) controlled, 8 risks (10.66%) mitigated, 3 risks (4%) aggravated and 13 risks (17.33%) only discovered. Figure 8 shows that 26 risks (50%) of 52 risks were discovered. Figure 9 shows that the factors of geographic distance and large project scope have led to a higher number of the probable risks, while different time offset and language have had the least effect on risk events in the studies. As shown in Figure 10, the risk areas of coordination and collaboration, communication, coding and integration, acquisition and development of project teams have been high-risk areas for DAD. Figure 11 also shows that group awareness, the software development lifecycle, and project management are the riskiest categories, the collaboration of external stakeholders, and the technology set up are the least risky ones. Figure 12 shows the discovered and eliminated risk in the case study.

In order to ensure the effectiveness of the framework, in another part of the study, the framework was used to investigate and analyze the number of selected user stories in each Sprint and compare them with the completed ones. The results of using this framework in these two teams are shown in Figure 13. The statistics shown in Figure 13 show that during the project, the level of the finished or completed user stories gradually grew. In fact, applying the framework has helped the team to gradually increase its efficiency in completing and expanding more user stories.

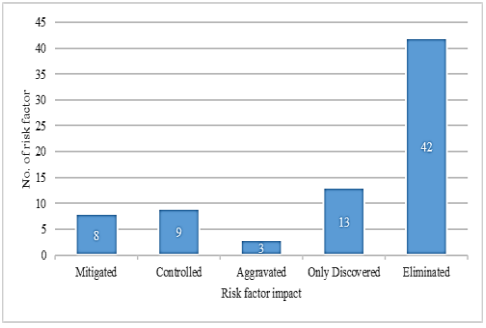

Figure 7. The impact of discovered risk factors

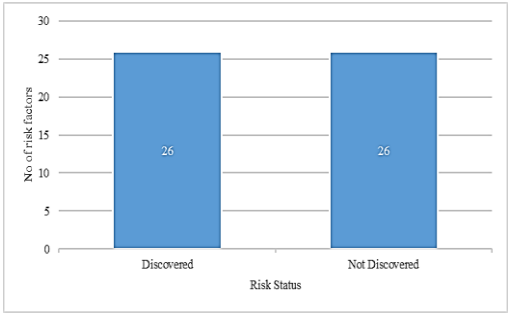

Figure 8. The status of risk factors

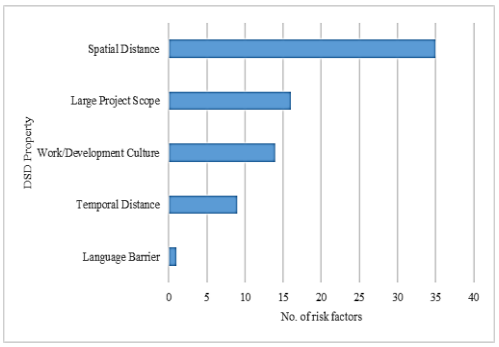

Figure 9. The effect of DSD characteristics in the occurrence of risk factors

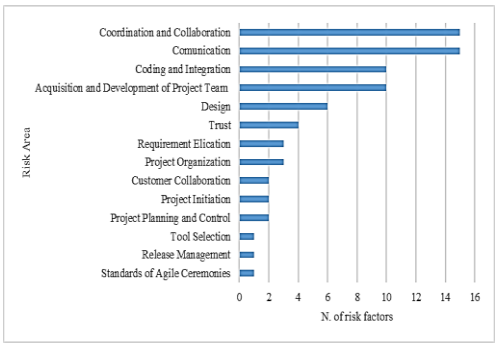

Figure 10. Risk areas and their risk factors

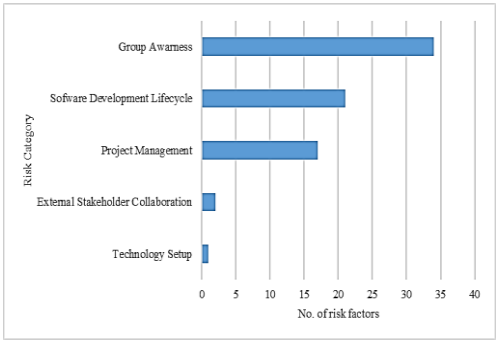

Figure 11. The number of risk factors in risk categories

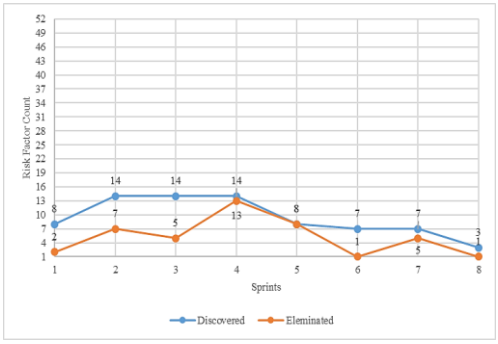

Figure 12. Discovered and eliminated risks

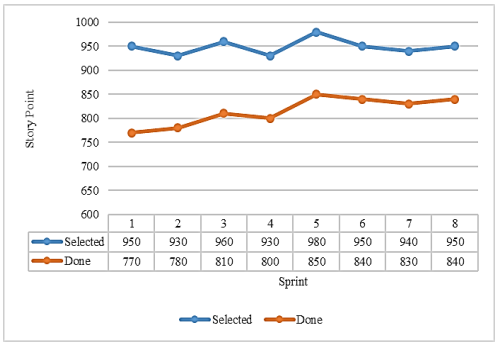

Figure 13. The number of user stories selected and done in the study

8. CONCLUSION

In this study, a risk management framework was developed by embedding Scrum in PRINCE2 methodology and identifying risk factors along with five categories of software development including lifecycle, collective awareness, project management, external stakeholder collaboration, and the launch of the technology. The results of the framework implementation showed that the life cycle of software development, collective awareness, and project management were the most-risky categories of study. The success rate of the project, given the Story Points, is 85.89%, which is a desirable rate. Also, 78.66% of known risk factors (identification, evaluation, and control) were managed, which is significant. Given the definition of activity and roles, it was expected that Agile principles would decrease somewhat. However, according to the results, agile principles are relatively well preserved.

Considering the scope of the proposed framework, it was expected that replacing it all at once would increase the risk in the project. The results show that the framework has been relatively favorable in this regard. Also, in terms of the impact of the framework on project costs, the performance is relatively

favorable. The results show that the framework has succeeded in filling the gap in the lack of a risk management method in Scrum. Also, the framework has worked well in terms of understanding and ease of learning and working in real environments. In terms of the flexibility of the framework in the operating environment and adaptation to team limits, there is also a relatively good performance. A significant number of risk factors are proposed by the framework. The results suggest that introducing these factors to the team will help to predict the challenges that may arise when implementing a Scrum project.

REFERENCES

[1] T. Quadri, M. Komal, and Z. Khalil, "A comprehensive study on risk analysis and risk management in IT industry," *International Journal of Computer and Communication System Engineering*, vol. 2, no. 4, pp. 561-568, 2015.

[2] Ramesh, L. Cao, K. Mohan, and P. Xu, "Can distributed software development be Agile?," *Communications of the ACM*, vol. 49, no. 10, pp. 41-46, 2006.

[3] E. Hossain, M. A. Babar, and H.-y. Paik, "Using scrum in global software development: a systematic literature review," *Global Software Engineering, Fourth IEEE International Conference*, pp. 175-184, 2009.

[4] K. Rai, S. Agrawal, and M. Khaliq, "Identification of Agile software risk indicators and evaluation of Agile software development project risk occurrence probability," *International Journal of Engineering Technology, Management and Applied Sciences*, vol. 5, no. 7, pp. 403-408, 2017.

[5] M. Niazi et al., "Challenges of project management in global software development: A client-vendor analysis," *Information and Software Technology*, vol. 80, pp. 1-19, 2016.

[6] S. V. Shrivastava and U. Rathod, "Categorization of risk factors for distributed Agile projects," *Information and Software Technology*, vol. 58, pp. 373-387, 2015.

[7] R. Jain and U. Suman, "A project management framework for global software development," *ACM SIGSOFT Software Engineering Notes*, vol. 43, no. 1, pp. 1-10, 2018.

[8] M. Mousaei and T. J. Gandomani, "A new project risk management model based on scrum framework and PRINCE2 methodology," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 4, pp. 442-449, 2018.

[9] S. V. Shrivastava and U. Rathod, "A risk management framework for distributed Agile projects," *Information and software technology*, vol. 85, pp. 1-15, 2017.

[10] Albadarneh, I. Albadarneh, and A. Qusef, "Risk management in Agile software development: A comparative study," *2015 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, pp. 1-6, 2015.

[11] D. Morris, "Scrum in easy steps," *Easy Steps Limited*, 2017.

[12] M. Cohn, "Succeeding with Agile: Software development using Scrum" *Pearson Education*, 2010.

[13] N. Santos, J. Pereira, F. Morais, J. Barros, N. Ferreira, and R. J. Machado, "An experience report on using architectural models within distributed Scrum teams contexts," ed: XP, 2018.

[14] Y. Khmelevsky, X. Li, and S. Madnick, "Software development using Agile and scrum in distributed teams," *Annual IEEE International Systems Conference (SysCon) IEEE*, pp. 1-4, 2017.

[15] J. Söderback, S. Hrastinski, and L.-M. Öberg, "Using distributed scrum for supporting online collaborative learning: A qualitative descriptive study of students perceptions," *IRIS*, 2015.

[16] M. Paasivaara, C. Lassenius, and V. T. Heikkilä, "Inter-team coordination in large-scale globally distributed scrum: Do scrum-of-scrums really work?," *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, pp. 235-238, 2012.

[17] C. Larman and B. Vodde, "Scaling Agile development," *CrossTalk*, vol. 9, pp. 8-12, 2013.

[18] S. V. Shrivastava, "Distributed Agile software development: A review," *Computer Science and Engineering*, vol. 1, no. 1, pp. 10-17, 2010.

[19] C. Bentley, "PRINCE2: A practical handbook," *Routledge*, 2010.

[20] AXELOS, "Managing successful projects with PRINCE2," *Stationery Office Limited*, 2017.

[21] J. Nyfjord and M. Kajko-Mattsson, "Outlining a model integrating risk management and Agile software development," *34th Euromicro Conference IEEE Software Engineering and Advanced Applications, SEAA'08*, pp. 476-483, 2008.

[22] S. Hastie and S. Wojewoda, "Standish group 2015 chaos report," *Retrieved*, vol. 1, no. 15, 2015.

[23] E. Hossain, M. A. Babar, H.-y. Paik, and J. Verner, "Risk identification and mitigation processes for using scrum in global software development," *Software Engineering Conference, Asia-Pacific, IEEE*, pp. 457-464, 2009.

[24] M. Kajko Mattsson, G. Azizyan, and M. Katrin Magarian, "Classes of distributed Agile development problems," *presented at the Agile Conference*, 2010.

[25] M. Tomanek and J. Juricek, "Project risk management model based on PRINCE2 and SCRUM frameworks," *International Journal of Software Engineering & Applications*, vol. 6, no. 1, pp. 81-88, 2015.

[26] R. Agrawal, D. Singh, and A. Sharma, "Prioritizing and optimizing risk factors in Agile software development," *presented at the Contemporary Computing (IC3), Ninth International Conference on*, 2016.

[27] B. G. Tavares, C. E. S. da Silva, and A. D. de Souza, "Risk management analysis in scrum software projects," *International Transactions in Operational Research*, pp. 1-22, 2017.

[28] E. E. Odzaly, D. Greer, and D. Stewart, "Agile risk management using software agents," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, pp. 823-841, 2017.

[29] E. T. Alharbi and M. R. J. Qureshi, "Implementation of risk management with SCRUM to achieve CMMI requirements," *International Journal of Computer Network and Information Security (IJCNIS)*, vol. 6, pp. 20-25, 2014.
[30] B. Chiste Brandão, "Risk management in software projects using scrum framework," *The 28th International Conference on Software Engineering & Knowledge Engineering, At Redwood, California, USA*, 2016.
[31] L. Enfei, "Risk factors of software development projects in Chinese IT small and medium sized enterprises," *Thesis*, 2015.
[32] J. Irizar and M. G. Wynn, "Centricity in project risk management: New dimensions for improved practice," *International Journal on Advances in Intelligent Systems*, vol. 8, no. 2, pp. 209-218, 2015.
[33] M. Griffiths, D. Harrison, M. Hartell, G. Hay, A. Kent, and S. Messenger, "Using DSDM with PRINCE2," *DSDM Consortium*, 2000.
[34] M. Lant, "Five simple steps to Agile risk management," *Software Development, Agile Methods and the Intersection of People Process and Technology, ed*, 2010.
[35] V. Ylimannela, "A model for risk management in Agile software development," *Communications of Cloud Software*, vol. 3, pp. 1-10, 2012.
[36] N. Uikey and U. Suman, "Risk based scrum method: a conceptual framework," *Proceedings of the 9th INDIACom; INDIACom-2015, IEEE Conference ID*, vol. 35071, pp. 4.120-4.125, 2015.
[37] J. Nyfjord and M. Kajko-Mattsson, "Commonalities in risk management and Agile process models," *Software Engineering Advances, ICSEA 2007. International Conference*, pp. 18-18, 2007.
[38] AXELOS, "The rationale for blending PRINCE2 and Agile," 2015. [Online]. Available: https://publications.axelos.com/Prince2Agile2015/content.aspx?page=cros_19&showNav=true&expandNav=true
[39] C. Larman, "Practices for scaling lean & Agile development: large, multisite, and offshore product development with large-scale scrum," *Pearson Education India*, 2010.
[40] J. Sutherland, A. Viktorov, J. Blount, and N. Puntikov, "Distributed scrum: Agile project management with outsourced development teams," *40th Annual Hawaii International Conference on System Sciences (HICSS'07), IEEE*, pp. 1-10, 2007.

BIOGRAPHIES OF AUTHORS

Mohammad Esteki received his B.S. degree in software engineering from Isfahan (Khorasgan) Branch, Islamic Azad University Isfahan, Iran in 2016. He is currently an M.S. student in software engineering in Department of Computer Engineering Isfahan (Khorasgan) Branch, Islamic Azad University Isfahan, Iran. His research interests lie in the areas of software methodologies, software testing, and software evolution.

Taghi Javdani Gandomani received his Ph.D. degree in software engineering from Universiti Putra Malaysia (UPM), Malaysia in 2014. He is currently serve as an Assistant Professor in Shahrekord University, Shahrekord, Iran. His research interests include software methodologies, Agile methods, software processes, and software metrics.

Hadi Khosravi Farsani received his Ph.D. degree in software engineering from Isfahan University, Iran in 2012. He is currently serve as an Assistant Professor in Shahrekord University, Shahrekord, Iran. His research interests include software development processes, semantic web, web engineering, and data mining.