A Risk Management Framework for Large Scale Scrum using Metadata Outer Request Management Methodology

Rehab Adel[1], Hany Harb[2], Ayman Elshenawy[3]
Faculty of Engineering, Al-Azhar University, Cairo- Egypt[1, 2, 3]
Faculty of Engineering and Technology, Egyptian, Chinese University, Cairo- Egypt[3]

***Abstract*—Recently, most software projects became naturally Distributed Agile Development (DAD) projects. The main benefits of DAD projects are cost-saving and being close to markets due to their distributed nature, such as in large-scale Scrum (LeSS). Developing LeSS projects leads to the emergence of challenges in risk management, especially the team collaboration challenges, where there is no standardized process for teams to communicate collaboratively. Team collaboration and the knowledge sharing is a vital resource for a large Scrum team's success. Hence, finding a dynamic technique that facilitates team collaboration in the LeSS environment is necessary. This paper proposes a risk management framework for LeSS using outer metadata requests. The proposed framework manages the outer requests amongst the distributed team. Therefore, it avoids missing team collaboration, risks, and threats to project completion. It also contributes to exchanging team skills and experience. The proposed framework is evaluated by applying it to two different case studies for large-scale Scrum projects. The evaluation results are given. The evaluation proved the effectiveness of the proposed framework.**

Keywords*—Distributed agile development; knowledge sharing; risk management; large scale scrum; metadata outer request management

I. INTRODUCTION

Agile is more robust than traditional software development methods. The agile manifesto formulation emphasizes customer involvement in the project, change request flexibility at any stage of the project, and delivers quality software at a cost-effective low and on time. Hence, there is a trend for software companies to globalize their agile development. A new type of agile software development has appeared in which team members work from various remote sites, referred to as distributed agile development (DAD) [1,2,3,4,5,6,7]. DAD incorporates many benefits, such as low production cost, the opportunity to involve the most developers around the world, and faster time to market [8].

Agile methodologies include Extreme Programming (XP), Scrum, Dynamic System Development (DSD), Lean Development (LD), etc. Most agile methods promote development iterations, working software, close collaboration between customers and developers, and process adaptability. The most widely used methodologies based on agile principles are XP and Scrum, where the most recently used agile methodology is Scrum [5, 7, 9, 10].

The Scrum framework comprises three components: roles, ceremonies, and artifacts. First, there are three distinct roles in the Scrum process (i) Scrum master: who organizes the Scrum process, review sessions, and meet with the team members, (ii) the product owner responsible for managing the project requirements, and (iii) the development team responsible for developing the validated requirements. The product owner and the development team can be grouped into feature teams. Secondly, the ceremonies have activities such as daily Scrum every day and sprint planning. A sprint is started, reviewed against the product owner's feedback, and possible changes are analyzed and completed retrospectively to suggest process improvements after sprint completion. Thirdly, there are three artifacts: (i) product backlog, (ii) sprint backlog, and (iii) burn down chart [7, 8, 11, 12, 13].

There is a shortage of highly skilled software development human resources in some software project locations. The migration of the skillful team from one physical location to another is a costly and challenging task. In this case, the IT projects are either challenged, impaired, or completed but failed due to a lack of IT human resources with the desired level of expertise [14,15]. Consequently, there is a need to improve software development infrastructure and human resources.

Organizations have to implement appropriate knowledge management practices. Previous studies have been analyzed proving that there are some problems of collaboration between distributed team members that affect knowledge sharing. Besides that, there are documentation obstacles like outdated documents and knowledge vaporization that result from much of the conversation and communication via chat [16].

In LeSS projects, Scrum team members can work from various remote sites to gain the maximum benefits of Scrum methodology. LeSS is a type of DAD. Many challenges are encountered when using Scrum methodology on LeSS projects, which are considered a primary source of emerging risks [7, 11]. These challenges related to daily Scrum meeting sessions based on team communication and customer involvement. The main difficulties result from: (i) geographical distances that cause many challenges in communication, (ii) Poor coordination between multiple teams, (iii) Conflict in requirements amongst the development team and numerous product owners, and (iv) Cultural differences such as language, religion, and social status between team members. These

difficulties reduce team cohesion and interdependence, besides causing a lack of collaboration and experience in managing distributed projects [2,3,4,7,17,19,20,21].

Such challenges lead to the appearance of some risks in LeSS. The potential risks are grouped into categories, each category contains several risk factors (RF). The most common RFs that significantly influence LeSS are:

1) Communication: There is no standardized process for the teams to communicate collaboratively [4].

2) Collaboration and coordination: due to the nature of LeSS, there are some difficulties in team coordination rules, plans, and feedback that cause misaligned software development activities during collaboration and iterative meetings among the teams [4].

3) Project management: One of its most essential tasks is risk management. Risk management implements several activities, such as (i) risk identification to identify and classify risks, (ii) risk evaluation to assess risks into three levels, such as (low, moderate, or high), and (iii) risk response to satisfy suitable actions and strategies to mitigate the impact of the risk, and (iv) risk monitoring to control and update the risk plan are all activities of project management [1,4,5,22,23]. IT project management activity includes all the structuring of the different phases of projects, so project objectives can be achieved optimally. As a result, there is a need to aggregate management methodologies into a single model, such as the approach to agile framework, Model-Driven Engineering (MDE) [24].

4) Software development lifecycle (SDLC): SDLC is made up of several phases that must be completed during the software development process, such as planning, analysis, design, implementation, and testing. Agile principles emphasized the individual's involvement in all SDLC phases, which is difficult in LeSS team development [23].

The goal of the proposed framework can represent many issues that can be summarized as follows:

- Achieve a formal coordination strategy based on centralization: Each Scrum master on the sender side receives requests from his feature team and forwards them to the Scrum masters on the receiver side. Scrum masters on the receiver side communicate with their teams to find replies to these requests. They deliver these replies to the Scrum master on the sender side.
- Help the Scrum masters carry out their risk management activities by:(i) risk identification through using the meta-data outer request attributes, where any request point includes a request from one side to the other side, and each request statement can be classified into a certain risk factor attribute. Therefore, the proposal can accommodate any type of risk factor that is involved in a specific request. (ii) risk evaluation. The feature team on each side assesses the requests' points by assigning each request point a reward value. (ii) risk response. The main plan for risk mitigation is that each team should receive accepted replies to each request's points. (iii) risk monitoring and control. This is achieved based on central management, where the Scrum masters can monitor the risks and their replies that are stored in the meta-data outer request attributes for all LeSS teams. Consequently, the Scrum masters successfully control these risks, ensuring that any emergence of new risks is covered.
- Sharing knowledge and Exchanging experience: The coordination process's results are used in building the knowledge repository, thereby contributing to increasing the team's learning process.

This paper is organized as follows: Section II represents related work and background. Section III introduces the LeSS development methodology. Section IV explains the metadata management. Section V presents the proposed model (the metadata outer request risk management framework). Section VI presents the implementation plan for the proposed model. Section VII introduces the discussion. Section VIII introduces the conclusion. Finally, the implications, limitations and future work are expressed.

II. Related Work and Background

There are several studies related to DAD risk management. Some of these studies introduce many new agile risk strategies to identify risks in DAD projects. Other studies determine new risk management practices for DAD projects. Some researchers have focused on finding solutions for knowledge sharing problems in distributed team's environment and have suggested frameworks in large-scale practices.

First, in [17],Suprika et al. have identified and classified DAD risks into categories, each category is ranked numerically. In [18], Mohammad Shameem et al. have suggested three stages for defining DAD risks: (i) risk definition and categorization, (ii) verifying the validity of the risk's definition stage with expertise, and (iii) risk priorities according to their importance. In [21], Shrivastava et al. have proposed an approach to identify risks according to three goals: saving time, quality, and cost for DAD projects.

Secondly, in [23], S. Bick et al. have applied a grounded theory data analysis on various datasets to prove that a lack of dependency awareness causes ineffective inter-team coordination, leading to misaligned planning activities. In [25], FS Rahayu et al. have proposed a Scrum framework based on the perspective of Scrum's stakeholders; they have analyzed the risk breakdown structure, the root of which represents the risk category and its related subcategories. In [26],Breno Gontijo Tavares et al. have presented a survey on risk management practices in agile projects. In [27], Rizwan Qureshi et al. have proposed a novel framework to improve communication and coordination among the Scrum master and team in Scrum methodology. Also, the proposed framework is validated through a questionnaire. In [28], Hoda et al. have suggested a framework for multi-level project management to achieve the "self-organized team" principle. The framework levels are (task-individual-team-project) and represent the role involved at each level. In [13], Tavares et al. have analyzed survey data, suggested risk management practices, and explained how risk management is carried out in Scrum software projects. In [29],

Bruno Gontijo Tavares et al. have proposed the Rm4Am (risk management for agile methods) tool to rank the list of 127 risk management practices into 48 subcomponents and then into five components (artifacts, features, events, roles, and methods). According to large-scale agile frameworks, there are difficulties encountered by companies. In [30], Kieran Conboy et al. have presented the three frameworks with LeSS complex adoption processes: (i)Safe, adoption of SAFe provides a comprehensive view of projects while requiring no significant restructuring of the company's processes. It is also a well-documented framework, more complex compared to the other frameworks. Work is delivered by individual teams that collaborate and contribute to the larger whole, (ii) Scrum at Scale, it is a straightforward and effective framework for reducing and avoiding the introduction of new complexity. (iii) Spotify, which addresses short-term challenges effectively and responds quickly to changes. The success of the three frameworks depends on the effectiveness of cooperation between teams and the exchange of skills and experiences. There is also a need for management centralization to facilitate project management practices. Consequently, overcoming risks appears due to the nature of LeSS team and to be able to control all elements of management.

Thirdly, in [31], Sara Waheed et al. have focused attention on finding a solution for only the knowledge vaporization problem that is related to the documentation process. It has proposed a framework for the documentation process to avoid the knowledge vaporization. The framework is evaluated using a real-life case study for distributed team members. The team members are satisfied with the proposed framework. In [32], Agile enterprise architecture (AEA) has been introduced for reducing IT costs and skill variation. Using the AEA, artefacts or models can enhance DAD team performance. It accommodates agile principles, focuses on collaborative incremental development and sharing of team skills and business information.

There are some limitations to the previous studies. Each of the previous studies focused on solving a specific problem facing distributed teams. But to avoid distributed teams' risks, there is a need to develop a comprehensive solution to avoid risks and help the team develop as well. Especially this is due to the multiplicity of sources and reasons for the emergence of these DAD-related risks.

Some of the previous studies relied on gathering limited data and risk management practices suggestions either from analyzing previous data surveys or from risk management practitioners. The collected data is mainly based on personal human observations, which raised some doubts about the validity of the data. Some of these studies suffered from the absence of dynamic risk management. So, the control and data updating have been carried out manually as a traditional risk management technique. Contrary to these studies, this paper proposes a comprehensive solution to the DAD-related problems. This paper proposes a risk metadata outer request risk management framework for LeSS projects.

The proposed framework is embedded in the LeSS organization to manage four RFs in LeSS development: (i) communication, (ii) collaboration and coordination, (iii) project management, and (iv) SDLC. The proposed framework also contributes to knowledge sharing amongst the leSS distributed team to increase the team experience. It is applied to a case study of four user stories in a LeSS project sprint spread across three locations. Besides, being applied to two projects for a company, the proposed framework helps the Scrum master to manage the outer requests amongst the LeSS distributed team.

III. Large Scale Scrum Process

The Scrum process has a management framework that manages complex software products and integrates many processes. The Scrum management methodology is applied to iterative and incremental life cycle models in software development. As presented in Fig. 1, the Scrum life cycle is divided into several stages. Each stage is called a "sprint", and the sprint period is usually from two to four weeks. It depends on five ceremonies; each ceremony has a short duration. If anyone's ceremony is not performed, they may lose an opportunity to complete a project.

Fig. 1. A Scrum Framework for Software Development [7].

Scrum, as a management framework, has the facility of monitoring the developed product and identifying project risks. Scrum teams are multifunctional teams in which every member has a good understanding of all the development functions. Also, Scrum teams are self-organized and can satisfy the best way to carry out their work without being led by anyone [7,13].

The Scrum process is suitable for small or medium-sized teams and projects. Nowadays, there is a trend toward using Scrum in large-scale projects with multiple and distributed teams, especially in multi-site projects. The main disadvantage of the Scrum process is the daily meeting ceremony sessions in LeSS that require a face-to-face meeting, which is challenging to use in large-scale projects[33,34].

In project planning, the product owner on the customer side prepares the product backlog, which is divided into chunks of small, desired functions. For sprint planning, a set of user stories are created from the ready items offered by the product owner. Each user story describes who uses the user story, its value to the customer, and its function [33,34].

The best management for the LeSS process has helped in solving this LeSS limitation. The workflow of LeSS is presented in Fig. 2, where the LeSS project development can be described as follows:

1) Each feature team in a location is assigned to one or several user stories from the planned sprint. They do the work plan and establish the sprint backlog to satisfy user interface (UI), code, unit tests, user acceptance tests, and task estimation time [33,34].

2) After the sprint development started, there was a daily stand-up meeting between the Scrum master and the feature team in each location. The Scrum master is responsible for ensuring that the team delivers value, helping to build a self-organizing team, and removing impediments [33,34].

3) After the sprint development is finished, a demo is prepared by the Scrum master to review the developed function with the development team. Then, sprint user stories in locations are integrated and thoroughly tested under the supervision of the product owner [33,34].

Finally, a retrospective meeting is conducted, and Scrum masters in each location integrate the results of their retrospective meeting [33,34].

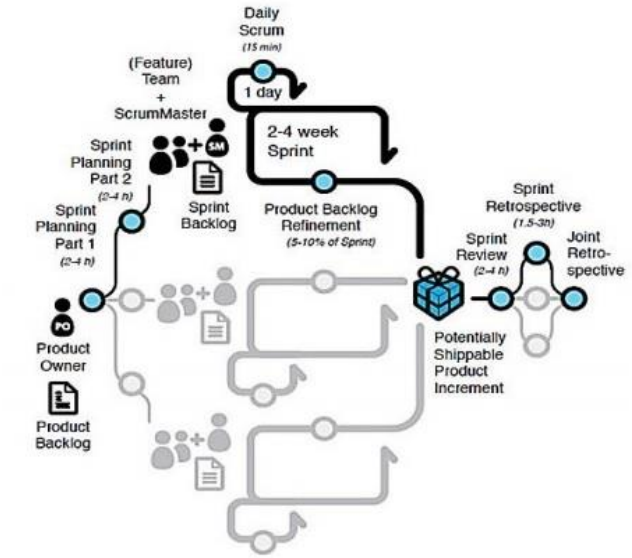

Fig. 2. A Large-Scale Scrum Framework [7].

IV. Metadata Management

The metadata management helps the project manager to perform all tasks related to project management by using data attributes. These data attributes carry the coordination and cooperation process of data through exchanging questions and dialogues among the project locations. The data recorded for these attributes facilitates the decision-making process.

In addition, these stored coordination's results are shared in building the team knowledge. Every organization can identify the data attributes of interest to them in the management or team coordination process.

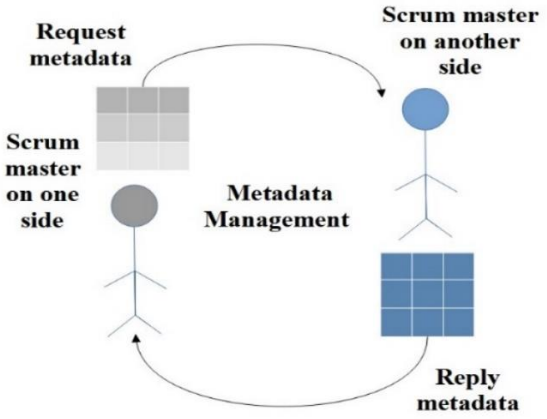

Fig. 3. Metadata Management on Two LeSS Sides.

This study applies metadata management to LeSS as a type of DAD project. Fig. 3 represents a metadata management process among two LeSS sides. The LeSS team coordination and the management process have been achieved through exchanging request points among the distributed team.

There are two types of metadata attributes needed:

1) Request metadata attributes: The request metadata attributes are request date, sprint number, point number, request description, risk factor (RF), and point reward ratio (PRR).

2) Reply metadata attributes: The reply metadata attributes are point number, reply date, reply description, reply status, reply content accepted, reply period status, and reply point reward ratio. Each Scrum master of a location is responsible for sending or receiving the exchanged request points. He also delivers the metadata to their featured team.

The request and reply metadata attributes will be explained in the next section.

V. Metadata Outer Request Risk Management Framework for LeSS

The architecture of the proposed metadata outer request risk management framework for LeSS is depicted in Fig. 4. It was applied to work on different sites in three locations. Each location has two roles, the feature team, and the Scrum master. Location A is for the sender's side, and locations B and C are for the receptor side. The framework consists of two main models: (i) the collaboration and coordination model and (ii) the knowledge sharing model.

A. Collaboration and Coordination Model

This model is responsible for collaborating and coordination between the sender and receiver sides by exchanging requests and replies for the shared tasks.

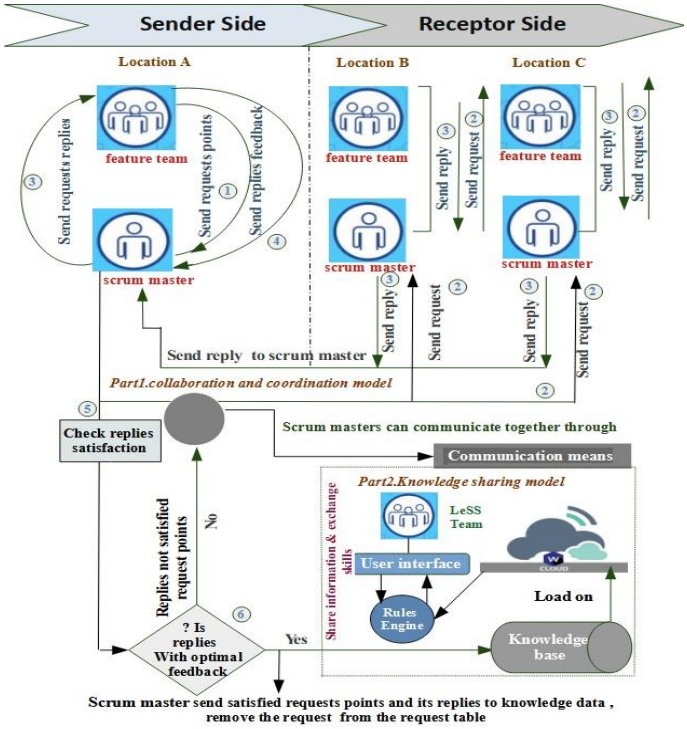

Fig. 4. Metadata Outer Request Risk Management Framework for LeSS.

Fig. 5 represents the flow sequence for a meta-outer request management from one location to the other locations, as, e.g., The process flow sequence for a request sent from side (A) to the different sides (B, C) can be summarized as follows:

1) At time t, the Scrum master on the sender side initializes the accumulative request reward at episode 0. The feature team on the sender side delivers the Scrum master metadata request that contains the obstacles that arises during the development process. It also includes the required coordination data with other locations.

2) The Scrum master on the sender side conducts a meeting with the Scrum masters on the receptor side to discuss the metadata request that the sender team has received from the sender. Each receptor Scrum master sends the request to their feature team during their meeting together. Each feature team on the receptor side studies the metadata requests and prepares replies wherever there is a reply to each request point. Then, the feature team forwards the replies to the receptor Scrum master at time t+1. After that, the receptor Scrum master delivers the replies to the sender Scrum master.

3) The feature team at the sender side evaluates the received replies using the assessment attributes for each reply point shown in Table I. The request point reward is calculated using a point reward (PR) function illustrated as follows:

PR = FRS * Point Reward Ratio (PRR). (1), and FRS = RS * RCA * RPS. The feature team at the sender side issues a new request state with the reset of un-replied request points.

4) Whenever request's replies are received, the sender Scrum master computes the Accumulative Reply Function (ARF) as in (Eq.2) that illustrate as follow: ARF = Sum previous (TPR) value + Sum current (TPR) value. (2), and Total points reward (TPR) = sum (PR) (3).

5) The Scrum master at the sender side checks if the request replies with optimal feedback rewards; if the optimal reward is not reached, the Scrum master resends the new request state to the receptor's Scrums and starts a new request action.

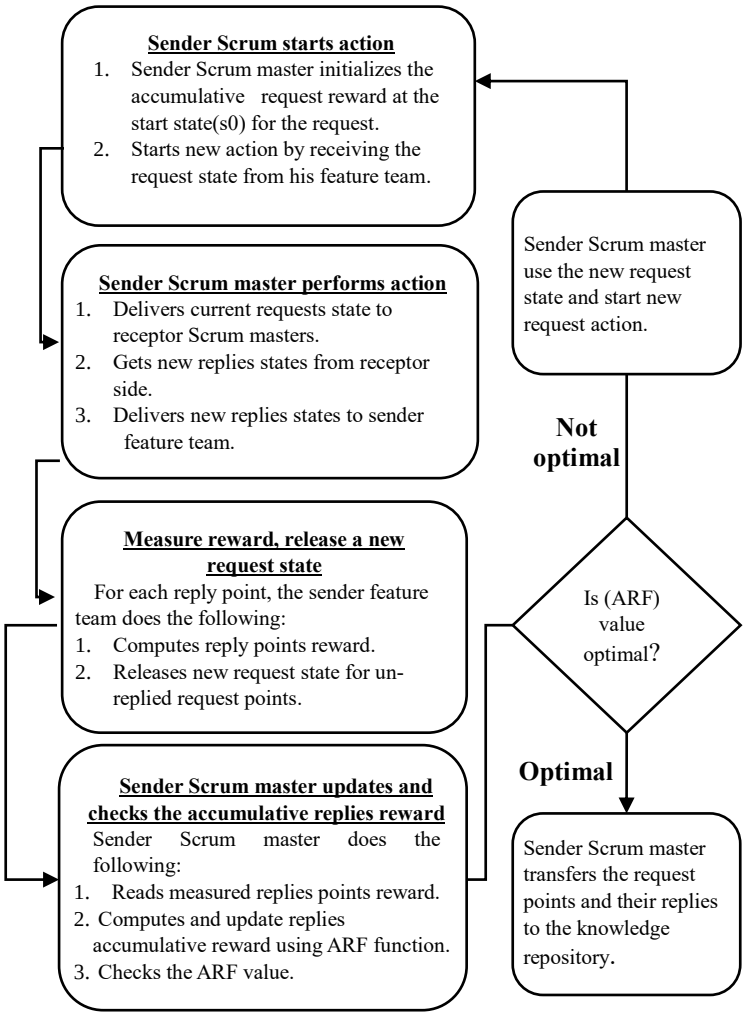

Fig. 5. Flow Sequence for Meta Outer Request Management.

TABLE I. REPLY ATTRIBUTES FOR THE ASSESSMENT REPLIES

Factor name	Factor description and related values
Reply Status (RS)	Indicate if there are a reply for the current request (value =1) point or not (value =0).
Reply Content Accept (RCA)	If the reply content is acceptable for the sender (value =1) If the reply content is not acceptable (value =0).
Reply Period Status (RPS)	Represents the reply to request points in time if the receptor responds to the request point in time less than or equal to 24 hours (value =1) else if greater than 24 hours (value =0).
Point Reward Ratio (PRR)	The reward ratio for each request point is according to the evaluation of the sender feature team.

B. Knowledge Sharing Model

The main objective of this model is to share information and exchange skills among the distributed team members. This can be achieved by building knowledge from the requests and replies to data exchanged amongst the LeSS teams. This knowledge is considered as a result of the coordination process. The scenario for building the knowledge repository is illustrated above at step (v) of the flow sequence. If a request-reply achieves the optimal reward, the Scrum master will send the complete information for the request and their replies to the knowledge sharing repository to be used by the LeSS team.

VI. Implementation

In this section, the proposed metadata outer request risk management framework is implemented and applied to two case studies. The first one is related to developing one sprint of a sales project using the LeSS developing method for only one request. The second one is related to developing several sales and purchase projects sprints in a medical service company using the LeSS developing method for many requests. The two case studies consist of many projects distributed in various locations. Many scenarios are applied to show the effectiveness of the proposed model.

A. Case Study 1: Sales Project

This section applies the proposed framework to one sprint of developing a sales project (project ID = P_sale1). First, the product owner and the Scrum masters fill the product backlog and highlight the high priority requirements to be developed first. The product owner discusses the highlighted user stories of one sprint and assigns user stories to suitable locations.

Table II shows the user stories assigned to locations A, B, and C in one sprint. User stories 1 and 4 are assigned to the development team in location A, user story 3 to the development team in location B, and user story 2 to the development team in location C. The sequence for managing the outer metadata request from one location to another can be described as the following:

1) The feature team in location A delivers the Scrum master metadata request state (s0) as shown in Table III with obstacles that faced them while completing their work. Initially, at time t, as shown in Table IV, the Scrum master at the sender side creates a reward table and initializes the replies' accumulative reward to zero. It also determines the reward ratio for each request point according to their job priority.

2) The Scrum master at location A executes an action a1 at time t+1 to change the request state. During their meeting, he delivers the request state to the receptor Scrum masters of locations B and C. Then, the receptor Scrum masters of locations B and C send the metadata replies, including the overall state of their locations. Finally, the Scrum master in location A passes these replies to his feature team. Tables V and VI include the reply states (s1) and (s2) for locations A and B.

3) The reward for reply state (s1) and (s2) for location A and location B is measured by the feature team in location A, as shown in Tables V and VI. The reward is computed using Total Points Rewards (TPR) function, where TPR for s1 = Sum (PR for s1) = (0 *0.20) + (0 *0.30) + (0 * 0.15) + (0 *0.35) = 0, TPR for s2 = Sum (PR for s2) = (1 *0.20) + (0 *0.30) + (1 * 0.15) + (0 *0.35) = 0.35. The feature team sends rewards for s1, s2, and sends the new request state(s3) (with the reset un-replied request points as shown in Table VII) to their Scrum master.

TABLE II. A Sprint with Four user Stories were Distributed amongst Three-Team Locations A, B, A, C

<table>
<tr><th>User story of Location A</th><th>User story of Location B</th></tr>
<tr><td>User story 1
As a system admin
I want to add and control new users.
So that I can control users and the user access to program components.
Acceptance criteria
1- I can enter a new user
2-configure these users
3-Assign them accessing some functions</td><td rowspan="2">User story 3
As A salesman
I want a review of the available quantity and last price for a product.
I can carry out operation processes,
I can review my sales daily
So that I can do sales operation in a suitable way
Acceptance criteria
1- Review available quantity and last price for a product?
2- Enter sales data for a customer contains?
3-Review our daily job is in excel formats</td></tr>
<tr><td>User story 4
As an account manager
I want a sales report to be sent daily to my mailbox. So that I can review the sale progresses.
Acceptance criteria
1- the report is sent daily to my mailbox
2-report contains important sales details
3-report is in excel formats</td></tr>
<tr><td colspan="2">User story of Location C</td></tr>
<tr><td colspan="2">User story 2
As A storekeeper
I want to enter store item quantity and price; I want to withdraw an available quantity from the store item.
So that I can do sales operation in a suitable way
Acceptance criteria
1- I can enter store items, review the data for entered items?
2- if withdraw a quantity larger than available quantity, system refuse</td></tr>
</table>

TABLE III. State (S0) is the Initial State for the Request

Metadata request from (A) for req. #1 for project (P_sale1)					
Request date	Sprint #	Point #	Request Description	RF	PRR
2020/01/01 01:12:00 PM	1	1	What are user data required to enter New user?	Communication With product owner	0.20
2020/01/01 01:12:00 PM	1	2	What are items balance attributes?	Coordination	0.30
2020/01/01 01:12:00 PM	1	3	What is UI and implementation to print excel report?	SDLC	0.15
2020/01/01 01:12:00 PM	1	4	What are the sales attributes?	Collaboration	0.35
Total Request Rewards					1

TABLE IV. Initialize Request Reward

Request state	**Request reward**
s0 (Initial state for the request)	0

TABLE V. THE REPLY STATE (S1) FROM (B) AFTER ACTION A1

Metadata reply from (B) after action a1 at t for request #1							
Point #	**Reply date**	**Reply description**	**RS**	**RPS**	**RCA**	**FRS**	**PR**
1	2020/01/01 07:1:00PM	Rep: user attributes are name, job, phone no	1	1	0	0	0 *0.20
2	2020/01/01 09:12:00 PM	Rep: Not mine	1	1	0	0	0*0.30
3		No reply	0	0	0	0	0*0.15
4	2020/01/01 03:12:00PM	Rep: Under construction	1	1	0	0	0 *0.35
Total points rewards (TPR) = sum (RP)							0

TABLE VI. THE REPLY STATE (S2) FROM (C) AFTER ACTION A1

Metadata reply from (C) after action a1 at t for req. #1							
Point #	**Reply date**	**Reply description**	**RS**	**RPS**	**RCA**	**FRS**	**PR**
1	2020/01/01 07:1:00PM	Rep: user attributes are name, job, phone no	1	1	1	1	1 *0.20
2	2020/01/01 09:12:00 PM	Rep: - Under construction	1	1	0	0	0 *0.30
3	2020/01/01 05:12:00 PM	Rep: -Ok, we try to do it	1	1	1	1	1*0.15
4	2020/01/01 03:12:00PM	Rep: -Not mine	1	1	0	0	0*0.35
Total points rewards (TPR) = sum (RP)							0.35

4) Following action a1, the Scrum master at location A updates the reward table (as shown in Table VIII) with the replies accumulative reward using the accumulative reward (ARF) mentioned in (Eq. 2), where the sum of previous TPR = 0, and the sum of current TPR =0.35. Then, ARF =0 + 0.35 = 0.35, so the accumulative reward for initial request state is changed from 0 to 0.35.

5) The scrum master checks: if the replies' accumulative reward is with optimal feedback (equal 1), so all request points that are replied with acceptable content, the sender Scrum master transfers the request and their replies to the knowledge sharing repository. The Scrum master starts a new action with the new request state if the optimal reply is not reached. Now, the Scrum master checks the request ARF value. He finds that it equals 0.35, which is not an optimal reward, so he starts a new action with the new request state (s3) shown in Table VII.

TABLE VII. NEW REQUEST STATE (S3) AFTER ACTION A1

Metadata request from feature team in location A for request #1 for the project (P_sale1)					
Request date	**Sprint #**	**Point #**	**Request description**	**RF**	**PRR**
2020/01/01 01:12:00PM	1	2	What are item balance attributes?	Coordination	0.30
2020/01/01 01:12:00 PM	1	4	What are the sales attributes?	Collaboration	0.35
Total Request Rewards					0.65

TABLE VIII. REWARD TABLE AFTER ACTION A1

Request state	Action(a1)
s0 (Initial state for the request)	0.35
s1	0
s2	0.35

6) The scrum master performs action a2 at time t+2 and receives a new reply state (s4) and (s5) for locations A and B, respectively, as shown in Tables IX and X. He sends two reply states to his feature team.

TABLE IX. THE REPLY STATE (S4) FROM (B) AFTER ACTION A2

Metadata reply from B after action a2 at t+1for request #1							
Point #	**Reply date**	**Reply description**	**RS**	**RPS**	**RCA**	**FRS**	**PR**
2	2020/01/04 05:12:00PM	Not mine	1	1	0	0	0 *0.30
4	2020/01/04 08:12:00PM	Customer attribute-: name, phone, address, Item attributes: code, qty, unit price	1	1	1	1	1 *0.35
Total points rewards (TPR) = sum (RP)							0.35

TABLE X. THE REPLY STATE (S5) FROM (C) AFTER ACTION A2

Metadata reply from C after action a2 at t+1for request #1							
Point #	**Reply date**	**Reply description**	**RS**	**RPS**	**RCA**	**FRS**	**PR**
2	2020/01/04 05:12:00 PM	Reviewing with product owner	1	1	0	0	0 *0.30
4	2020/01/04 08:12:00 PM	Not mine	1	1	0	0	0*0.35
Total points rewards (TPR) = sum (RP)							0

7) The feature team in location A measures the reward for reply state (s4) and (s5) for locations A and B, respectively, as shown in Table IX and X. The reward for reply state (s4) and (s5) for location A and location B is measured by the feature team in location A, as shown in Tables IX and X. Total points rewards (TPR) for s4 = Sum (PR for s4) = (0 *0.30) + (1 *0.35) = 0.35, TPR for s5 = Sum (PR for s5) = (0 *0.30) + (0 *0.35) =0. The feature team sends s4, s5 rewards and sends requests to state s6 (as shown in Table XI) to their Scrum master.

8) Following action a2, the Scrum master at location A updates the reward table (as shown in Table XII) with the replies accumulative reward using the accumulative reward (ARF) mentioned in (Eq. 2), where the sum of previous TPR = 0.35, and the sum of current TPR =0.35. Then, ARF =0.35 + 0.35 = 0.70, so the accumulative reward for initial request state is changed from 0.35 to 0.70.

9) The scrum master checks: if the replies' accumulative reward is with optimal feedback (equal 1). He finds that it equals 0.70, which is not an optimal reward, so he starts a new action with the new request state (s6) shown in Table XI.

*10)*The Scrum master performs an action a3 at time t+3 and receives a new reply state (s7) and (s8) for locations A and B, respectively, as shown in Tables XIII and XIV. He sends two reply states to his feature team.

*11)*The reward for reply state (s7) and (s8) for location A and location B is measured by the feature team in location A, as shown in Tables XIII and XIV. Total points rewards (TPR) for s7 = Sum (PR for s7) = (0 * 0.30) = 0, TPR for s8 = Sum (PR for s8) = (1 * 0.30) = 0.30. The feature team sends the rewards for s7 and s8 and the new request state (s9) (with the reset un-replied request points shown in Table XV) to its scrum master. There are no un-replied request points, so the sender feature team releases all request points and their replies to their scrum master.

*12)*Following action a3, the scrum master at location A updates the reward table with the replies accumulative reward, where the sum of the previous TPR = 0.70, and the sum of current TPR =0.30. Then ARF = 0.70 + 0.30 = 1, This means that the accumulative reward for the initial request state has been increased from 0.70 to 1(optimal reward). As shown in Table XVI, states s1, s2, s4, s5, s7, and s8 are real states for replies, whereas s0, s3, and s6 are transition states that serve as the starting point for the next state.

*13)*Now, the Scrum master checks the requested ARF value. He finds that the value equals 1, an optimal reward, so the goal is reached. The sender Scrum master transfers the request points and their replies that he accepted from his feature team (as shown in Table XV) to the knowledge repository state.

TABLE XI. NEW REQUEST STATE (S6) AFTER ACTION A2

Metadata request from feature team in (A) for request #1 for the project (P_sale1)					
Request date	**Sprint #**	**Point #**	**Request description**	**RF**	**PRR**
2020/01/01 01:12:00PM	1	1	What are user data required to enter new user?	Communication With product owner	0.30
Total Request Rewards					0.30

TABLE XII. REWARD TABLE AFTER ACTION A2

Request state	**Action(a1)**	**Action(a2)**
s0 (Initial request)	0.70	0
s1	0	0
s2	0.35	0
s3	0	0
s4	0	0.35
s5	0	0

TABLE XIII. THE REPLY STATE (S7) FROM (B) AFTER ACTION A3

Metadata reply from (B) after action a3 at t+2for request #1							
Point #	**Reply date**	**Reply point description**	**RS**	**RPS**	**RCA**	**FRS**	**PR**
1	2020/01/04 05:12:00 PM	Not mine	1	1	0	0	0*0.30
Total points rewards (TPR) = sum (RP)							0

TABLE XIV. THE REPLY STATE (S8) FROM (C) AFTER ACTION A3

Metadata reply from C after action a3 at t+2for request #1							
Point #	**Reply date**	**Reply point description**	**RS**	**RPS**	**RCA**	**FRS**	**PR**
1	2020/01/04 05:12:00 PM	Ok, will sent to your scrum master	1	1	1	1	1 *0.30
Total points rewards (TPR) = sum (RP)							0.30

TABLE XV. REWARD TABLE AFTER ACTION A3

Request state	**a1**	**a2**	**a3**
s0	1	0	0
s1	0	0	0
s2	0.35	0	0
s3	0	0	0
s4	0	0.35	0
s5	0	0	0
s6	0	0	0
s7	0	0	0
s8	0	0	0.30

TABLE XVI. THE GOAL STATE AFTER ACTION A3

Request date	Point #	Request description	PRR	Reply date	Reply description	Location
2020/01/01 01:12:00 PM	1	Communi-cation with the product owner	What are user data required to enter new user?	2020/01/01 06:12:00PM	Rep: user attributes are name, job, phone no	C
2020/01/01 01:12:00 PM	2	Coordination	What are item balance attributes?	2020/01/04 05:12:00PM	Ok, will sent to your scrum master	C
2020/01/01 1:12:00 PM 0	3	SDLC	What is UI and implementation to print excel report?	2020/01/01 05:12:00PM	Rep: - Ok, we try to do it	C
2020/01/01 01:12:00 PM	4	Collaboration	What are the sales attributes?	2020/01/04 08:12:00 PM	Customer attribute-:name, phone, address, Item attributes : code, qty, unit price	B

B. Case Study 2: Sales and Purchase Project

In this section, the proposed model is implemented in real-time for developing a LeSS project of two sub-projects distributed physically in two locations. The LeSS team collaborated to develop two projects for a medical services company. The first project is a purchase project with a project ID = P_sale. This project belongs to a branch of the medical services company specialized in purchasing medicines and medical supplies. After the purchasing process, these items are stored in the company's warehouses. The second project is a sales project with ID= P_pur. It belongs to another branch of the medical services company for selling and marketing medical items.

Metadata attributes derived from team coordination after their development activities can be summarized as follows: (i) Request number; (ii) Request date; (iii) Project identifier; (iv) No sprint; (v) Number of request points sent from one location's development team to another location's development team; (vi) Number of replies to points; (vii) Total reward (Number of reply points/Number of request points); (viii) The number of episodes per request; (ix) Reliability (total reward/number of episodes); and (x) The number of risk factors covered in replies. Fig. 5 and 6 depict the data presented in Table XVII. The plot represents the relationship between reliability and the requests for the purchase project (P_pur) and the sales project (P_sale). The data plotted in Fig. 6 represents the reply points covered for each RF.

TABLE XVII. OUTER REQUEST DATA FOR PROJECT (P_SALE, P_PUR)

# Of req. and reply. points per project							Covered RF in replies			
Req. #	Sprint #	# Of req. points	# Of repl. points	Total reward	# Of episode per req.	Reliability =Tot. reward/# of episode	Communication	Coll.& Coord.	Project manag.	SDLC
Project ID = P_sale										
10	2	7	6	0.86	3	0.29	2	1	1	2
11	2	10	8	0.80	2	0.40		4	2	2
12	2	8	7	0.88	2	0.44	1	2	3	1
13	2	13	12	0.92	2	0.46	5	1	2	4
14	3	14	14	1	1	1		5	2	7
15	3	13	13	1	1	1	2		2	9
16	3	17	17	1	1	1	2	4	4	7
17	3	13	13	1	1	1	1	7	3	2
Project ID = P_pur										
19	1	13	12	0.92	4	0.46	3	2	4	3
20	2	15	14	0.93	3	0.33	2	9	2	1
21	3	8	7	0.87	1	0.87		1	3	3
22	4	11	11	1	1	1		4	5	1
23	4	7	7	1	1	1	1	3	1	2
24	5	20	20	1	1	1	1	5	8	6
10	2	7	6	0.86	3	# RF	20	48	42	50
		Sum = 169	Sum = 161			% RF	13 %	30 %	26 %	31 %

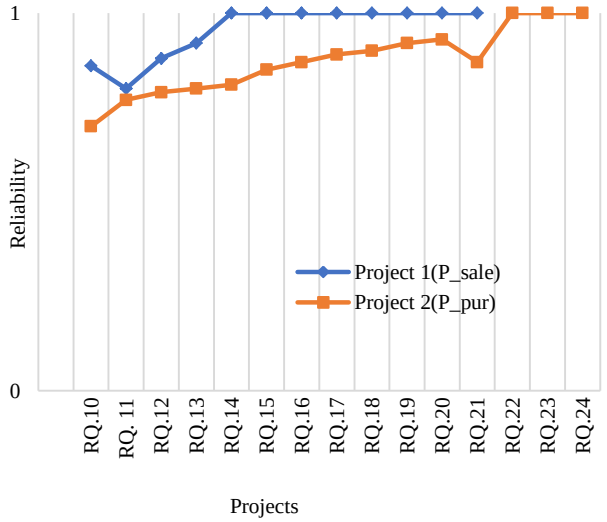

Fig. 6. Relation between Project Request and Reliability.

In Fig. 7, it is observed that the first two most frequent RFs are "Collaboration and Coordination" and "SDLC". On the other hand, the least frequent RFs are "Communication" and "Project management ".

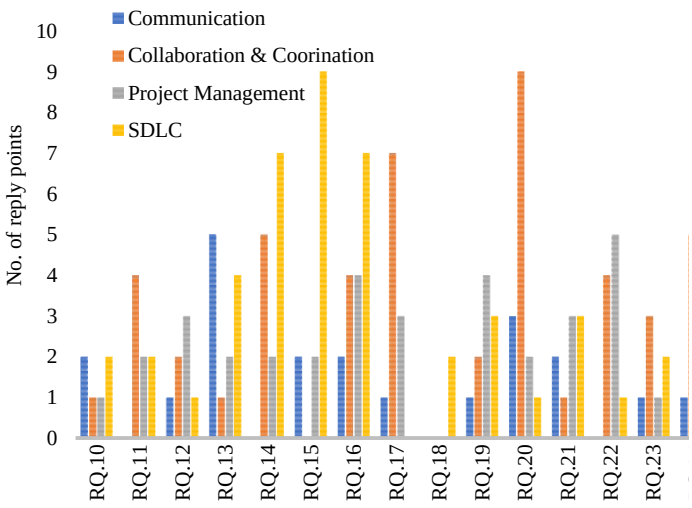

Fig. 7. Requests Points Numbers per each RF.

VII. Discussion

The obtained results show that the total covered risks in the accepted replies are 161 points out of 168. 31% are related to risks in the SDLC, 30% are related to the risks resulting from collaboration and coordination, and 26% are related to risks resulting from project management. Finally, 13% is associated with the risk of results due to miscommunication between the teams. The proposed model achieves a success rate of 95% of replies to requests initialized by the teams. The reliability results indicate that two factors affect the reliability of any request: (i) the number of reply points per request. The number of reply points per request increases proportionally to the number of request points. (ii) The number of episodes per request. With a decreasing number of episodes per request, the reliability is increased, and the learning process occurs faster. The LeSS team experience will be rapidly improved.

VIII. Conclusion

A large-scale Scrum is a type of agile development project with a distributed team that faces several challenges. This study provides comprehensive suggestions for formal coordination strategies based on centralization. The suggested framework is embedded in the LeSS organization. So, it is successful in managing LeSS risk factors and highly centralized coordination has been achieved. The successful coordination results contribute to the sharing of information and knowledge amongst the LeSS distributed team, so success in increasing team experience is achieved, and the team becomes an agile mindset team. The paper has illustrated the methodology applied to overcome DAD risks in LeSS and how this framework can build experience and skills for a distributed team. Finally, the team is enabled to achieve openness and success in working in distributed agile environments.

IX. Implications

This framework contributes to increasing LeSS team cohesion and motivates the software professional team to work together to achieve the project's progress and performance. This framework is based on building a formal management strategy. Therefore, the management organization can control LeSS software development practices. This can be achieved by concentrating project management on Scrum masters on each side of the dependent. Software professionals and managers should encourage members to share their skills and experience with their colleagues. The project manager should understand that knowledge sharing within the team provides clarity to members about the project and its deliverables. It is possible for researchers to apply the suggested framework to more LeSS projects and monitor the number of risk factors that are actually avoided. In this framework, the researchers can suggest different strategies to benefit from the knowledge data accumulated as a result of the coordination process. They can also observe any shortcomings in this proposal and try to suggest improvements.

X. Limitations and Future Work

The proposed framework is evaluated via two case studies. It yields positive results in terms of increasing team collaboration and applying risk management through centralized management. The current study has two limitations. One of these limitations is clarifying the concept of central management using Scrum masters. This is due to applying the proposed framework to only two studies, especially the first case, which contains one request proposed by a team on the DAD project side. The second limitation arises because there are no methods satisfied for how to store the data and how to retrieve data from the knowledge repository. For further research, it will be exciting to apply the proposed framework to more real-life case studies for the LeSS team practices in distributed organizations. It is also interested in spreading the idea of centralization and studying suitable methods that can be applied to store the knowledge data in a knowledge repository. Searching for the required technique and learning how to retrieve the requests and their replies from the knowledge repository, it is an important issue that needs to be resolved.

References

[1] Torgeir Dingsøyr, Nils Brede Moe1, Tor Erlend Fægri, Eva Amdahl Seim, "Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation", Empir Software Eng, Vol. 23, No. 1, pp. 490–520,2018, Doi: 10.1007/s10664-017-9524-2.

[2] Manasés Jesús Galindo Bello, Hochschule Fulda, Natarajan Meghana than et al, "SOFTWARE ENGINEERING IN GLOBALLY DISTRIBUTED TEAMS", Computer Science & Information Technology, pp. 01–16, CSCP2018, Doi: 10.5121/ csit .2018.81301.

[3] Madan Singh, Naresh Chauhan, Rashmi Popli,"A Framework for M. Singh, N. Chauhan and R. Popli, "A Framework for Transitioning Of Traditional Software Development Method To Distributed Agile Software Development," 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), 2019, pp. 1-4, Doi: 10.1109/ICICT46931.2019.8977654.

[4] Wan Suzila Wan Husina, Yazriwati Yahyab, Nurulhuda Firdaus Mohd Azmib, Nilam Nur Amir Sjarifb, Suriayati Chupratb, Azri Azmib, "Risk Management Framework for Distributed Software Team: A Case Study of Telecommunication Company", Procedia Computer Science, Vol.161, pp. 178–186, 2019.

[5] Abdelghany Salah Abdelghany, Nagy Ramadan Darwish, Hesham Ahmed Hefny , "Towards a Hybrid Approach for Software Project Management using Ontology Alignment", International Journal of Computer Applications Vol. 168, No. 6, pp. 0975 – 8887,2017,Doi: 10.5120/ijca2017914438.

[6] Sinha, Richa, Shameem, Mohammad, Kumar, Chiranjeev, "SWOT: Strength, Weaknesses, Opportunities, and Threats for Scaling Agile Methods in Global Software Development", Innovations in Software Engineering Conference, Jabalpur, India, pp.27–29, 2020, Doi: 10.1145/3385032.3385037.

[7] Mohammad Esteki, Taghi Javdani Gandomani, Hadi Khosravi Farsani, "A risk management framework for distributed scrum using PRINCE2 methodology", Bulletin of Electrical Engineering and Informatics, Vol.9, No. 3, pp. 1299~1310, 2020, Doi: 10.1109/ICISE.2019.00014.

[8] Miloš Jovanović, Antoni-Lluís Mesquida,Antonia Mas, Ricardo Colomo-Palacios, "Agile transition and adoption frameworks issues and factors: A systematic mapping", IEEE Access, vol. 8, pp. 15711-15735, 2020.

[9] Melaku Girma, Nuno M. Garcia, Mesfin Kifle, "Agile Scrum Scaling Practices for Large Scale Software Development", 4th International Conference on Information Systems Engineering, pp.1291–2160, 2020, Doi: 10.1109/ICISE.2019.00014.

[10] Waqar Aslam, Farah Ijaz, "A quantitative framework for task allocation in distributed agile software development", IEEE Access, vol. 6, pp. 15380-15390, 2018.

[11] Ayesha Khalid, Shariq Aziz Butt, Tauseef Jamal, Saikat Gochhait, "Agile Scrum Issues at Large-Scale Distributed Projects: Scrum Project Development At Large", International Journal of Software Innovation, Vol. 8, No. 2, pp. 85–94,2020., Doi: 10.4018/IJSI.2020040106.

[12] Gabriela Castro Flores, Ana M. Moreno, Lawrence Peters, "Agile and Software Project Management Antipatterns: Clarifying the Partnership" in IEEE Software, vol. 38, no. 05, pp. 39-47, 2021, Doi: 10.1109/MS.2020.3001030.

[13] Tavares, Breno, Silva, Carlos, Diniz de Souza, Adler, "Risk management analysis in Scrum software projects", International Transactions in Operational Research, pp.1–22,2017, Doi: 10.1111/itor.12401.

[14] Hui Yi Chiang, Bertrand M. T. Lin, "A Decision Model for Human Resource Allocation in Project Management of Software Development," in IEEE Access, vol. 8, pp. 38073-38081, 2020, Doi: 10.1109/ACCESS.2020.2975829.

[15] Prasad Sharma, Durga, Cloud-Based Outsourcing Framework for Efficient IT Project Management Practices (October 1, 2020). (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 11, No. 9, 2020, https://thesai.org/Publications/ViewPaper?Volume=11&Issue=9&Code=IJACSA&SerialNo=18, Available at SSRN: https://ssrn.com/abstract=3703028.

[16] César France, Fabio Q. B. da Silva, Helen Sharp, "Motivation and Satisfaction of Software Engineers," in IEEE Transactions on Software Engineering, vol. 46, no. 2, pp. 118-140, 1 Feb. 2020, Doi: 10.1109/TSE.2018. 2842201.

[17] Suprika Vasudeva Srivastava, Urvashi Rathod b, "A risk management framework for distributed agile projects", Information and Software Technology Vol 85, pp. 1-15, 2017.

[18] Mohammad Shameem, Rakesh Ranjan Kumar, Chiranjeev Kumar, Bibhas Chandra, Arif Ali Khan, "Prioritizing challenges of agile process in distributed software development environment using analytic hierarchy process", J Softw Evol Proc.30: e1979,16, 2018.

[19] Yahia Ibrahim Alzoubi, Asif Qumer Gill a, Ahmed Al-Ani, "Empirical studies of geographically distributed agile development Communication challenges: A systematic review", Information &management, Vol.53, pp.22-37, 2016.

[20] Muhammad Akil Rafeek, Adila Firdaus Arbain, Endah Sudarmilah, "Risk mitigation techniques in agile development processes", International Journal of Supply Chain Management, Vol. 8, pp. 1123-1129, No.2, 2019.

[21] Shrivastava, Suprika, Rathod, Urvashi, "A Goal-driven Risk management approach for Distributed Agile Development Projects", Australasian Journal of Information Systems, Vol 23, pp. 4-29,2019, Doi 10.3127/ ajis. v23i0.1843.

[22] Gu¨nther Ruhe, Claes Wohlin,"Software Project Management in a Changing World", ISBN 978-3-642-55034-8 ISBN 978-3-642-55035-5 (eBook)- Doi:10.1007/978-3-642-55035-5.

[23] Saskia Bick, Kai Spohrer, Rashina Hoda, Alexander Scheerer, Armin Heinz, "Coordination Challenges in Large-Scale Software Development: A Case Study of Planning Misalignment in Hybrid Settings," in IEEE Transactions on Software Engineering, vol. 44, no. 10, pp. 932-950, 1 Oct. 2018, Doi: 10.1109/TSE.2017.2730870.

[24] Hamzane Ibrahim, Belangour Abdessamad,"Project Management Metamodel ConstructionRegarding IT Departments", (IJACSA) International Journal of Advanced Computer Science and Applications,Vol. 10, No. 10, 2019.

[25] F S Rahayu, T Indrawan, S Kamarudin, "Risk Mitigation Strategies in Implementing Scrum Framework for Internet-Based IT Companies in Indonesia", Journal of Computer Information Systems", Indonesian Journal of Information Systems (IJIS)), Vol. 3, No. 1, ISSN 2623-0119, E- ISSN 2623-2308, August 2020.

[26] Breno Gontijo Tavares, Carlos Eduardo Sanches da Silva, and Adler Diniz de Souza, "Practices to Improve Risk Management in Agile Projects", International Journal of Software Engineering, and Knowledge Engineering, Vol. 29, No. 3, pp. 381–399,2019, Doi: 10.1142/S0218194019500165.

[27] Rizwan Qureshi, Mohammed Bashiri, Ahmad A AL Zahrani, "Novel Framework to Improve Communication and Coordination among Distributed Agile Teams", I.J. Information Engineering and Electronic Business, Vol.4, No. 3, pp. 16-24, 2018, Doi: 10.5815/ijieeb.2018.04.03.

[28] Hoda, Rashina, "Multi-Level Agile Project Management Challenges: A Self-Organizing Team Perspective", Journal of Systems and Software, Vol. 117, pp.245–257, Doi: 10.1016/j.jss.2016.02.049.

[29] Breno Gontijo Tavares, Mark Keil, Carlos Eduardo Sanches da Silva & Adler Diniz de Souza (2020),"A Risk Management Tool for Agile Software Development", Journal of Computer Information Systems, Vol.61, Issue 4, Doi: 10.1080/08874417.2020.1839813.

[30] Kieran Conboy, Noel Carroll, "Implementing Large-Scale Agile Frameworks: Challenges and Recommendations", IEEE Software, vol. 36, no. 2, pp. 44-50, 2019.

[31] Sara Waheed, Bushra Hamid, NZ Jhanjhi, Mamoona Humayun, Nazir A Malik5,"Improving Knowledge Sharing in Distributed software Development",(IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 10, No. 6, 2019.

[32] Yehia Ibrahim Alzoubi, Asif Qumer Gill, "An Empirical Investigation of Geographically Distributed Agile Development: The Agile Enterprise Architecture is a Communication Enabler," in IEEE Access, vol. 8, pp. 80269-80289, 2020, doi: 10.1109/ACCESS.2020.2990389.

[33] Apoorva Srivastava, Sukriti Bhardwaj, Shipra Saraswat, "SCRUM model for agile methodology," 2017 International Conference on Computing, Communication and Automation (ICCCA), 2017, pp. 864-869, Doi: 10.1109/CCAA.2017.8229928.

[34] Muhammad Hammad, Irum Inayat, "Integrating Risk Management in Scrum Framework," 2018 International Conference on Frontiers of Information Technology (FIT), 2018, pp. 158-163, Doi: 10.1109/FIT.2018.00035.