Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 6696695, 18 pages
https://doi.org/10.1155/2021/6696695

Review Article

Requirement Engineering Challenges in Agile Software Development

Aqsa Rasheed,[1] Bushra Zafar,[2] Tehmina Shehryar,[1] Naila Aiman Aslam,[1] Muhammad Sajid,[3] Nouman Ali,[1] Saadat Hanif Dar,[1] and Samina Khalid[4]

[1]*Department of Software Engineering, Mirpur University of Science & Technology (MUST), Mirpur 10250, AJK, Pakistan*
[2]*Department of Computer Science, Government College University, Faisalabad 38000, Pakistan*
[3]*Department of Electrical Engineering, Mirpur University of Science & Technology (MUST), Mirpur 10250, AJK, Pakistan*
[4]*Department of Computer Science & Information Technology, Mirpur University of Science & Technology (MUST), Mirpur 10250, AJK, Pakistan*

Correspondence should be addressed to Nouman Ali; nouman.ali@live.com

Received 2 December 2020; Revised 20 April 2021; Accepted 27 April 2021; Published 10 May 2021

Academic Editor: Zhibin Wu

Agile software development has large success rate due to its benefits and promising nature but natively where the size of the project is small. Requirement engineering (RE) is crucial as in each software development life cycle, “Requirements” play a vital role. Though agile provides values to customer’s business needs, changing requirement, and interaction, we also have to face impediments in agile, many of which are related to requirement challenges. This article aims to find out the challenges being faced during requirement engineering of agile projects. Many research studies have been conducted on requirement challenges which are somehow biased, no suggestions are given to improve the agile development process, and the research does not highlight large-scale agile development challenges. Hence, this article covers all the challenges discussed above and presents a comprehensive overview of agile models from requirement engineering perspective. The findings and results can be very helpful for software industry to improve development process as well as for researchers who want to work further in this direction.

1. Introduction

Agile software development has many advantages over traditional and linear development models. Up-front and preliminary preparation is sufficient to get all factors for consideration that might influence the development process [1], but think for while is it possible to get all those factors or variables that affect the system? Empirically, the answer should be “no.” To cope change, uncertainty, and unknown factors, an adaptive approach should be used.

Agile is an adaptive approach, and factors like high success rate and remarkable results have made it very popular in software development community. Agile software development (ASD) works best with respect to traditional development approaches and practices due to many reason as rapidly changing needs of customers, urgent business requirements, and flexible nature besides many others. The choose of right SDLC (software development life cycle) while developing software solution is very important task either water fall, V model, or agile all of them have their own advantages and disadvantages [2]. The choice of SDLC approach mainly depends on the nature of the product being developed.

In software development, ”Requirement engineering” (RE) plays a vital role as the whole success of software depends on the correctness of requirements gathered requirement in engineering phase [3, 4]. So, when the nature of requirement is dynamic, it is not a simple task to gather, analyze, understand, and manage requirements [5]. The dynamic nature of requirements and their consideration according to project completion is necessary as we move from first phase of the project to the final phase thus change consideration is important factor and applied in large ratio in project postmaintenance [6]. In traditional sequential methods, the RE phase is freezed and then later there is no user involvement and there is no way to cope with a change. Though agile software development (ASD) is better than other traditional software methods in many ways, i.e., it supports direct team involvement, early release of software, embracing change, cooperate the need, and change of emerging technology, and many other. Although there are so many benefits of agile development, but still proper RE is critical for successful projects. Contrary to the traditional SE process, the agile method relies on iterative development and face-to-face communication.

In software industry, the success rate of project is depended on the correctness of requirements irrespective of how they are gathered? Whether they are gathered correctly or not according to user perspective? According to [7], in agile development, the process of requirement engineering (RE) is not just restricted to the early phase as in beginning such as in traditional models but continues till whole development cycle.

Balaji and Murugaiyan [2] presented a comparative study on SDLC and discussed the pros and cons of waterfall, V model, and agile model. Their findings are if project is small and requirements change continuously chooses agile approach and if project is large and requirements are clear, then waterfall approach is the right choice, and if project is large and requirement changed continuously, then the V model serves the purpose. Due to proven success rate of agile methodology, it is now-a-days being adopted by larger organization for complex and large projects. De Lucia and Qusef [8] discussed how to obtain and define user specification with some results and solution confronted with agile approaches. The Standish Group International CHAOS Survey identified many causes, challenges, and failure factors of software projects. Studies show that the project development challenges are 37% from the requirement phase as illustrated in Figure 1 [8, 9]. The problems and solutions of requirement traceability in ASD and some guidelines for ARE (agile requirement engineering) have been addressed. According to [10], the majority of the software projects fail due to the poor requirements management and due to common challenges which cause the failure of project.

According to Kettunen and Laanti [11], software development companies use agile approaches where small team works very closely with customers to build high-quality software with frequent iterations and feedback. If the nature of the project is small, agile methodology works very well and produces effective products. To gain benefits of agile in case of complex software products and organization, the basic assumptions will not work and there is a need to satisfy additional constraints. Nerur et al. [12] reported the migration challenges to agile. There are many scenarios where companies that are following full or partly agile practices for large-scale or distributed projects had positive impact of using agile in large-scale software projects [13–15]. For example Yahoo, Google, Microsoft, Siemens, CNBC, AOL, and others are using agile in their development projects [16]. There are many principals and guidelines available for using agile methodology for complex and large-scale software development such as the utility of rapid application for the development of large and complex projects [17–19]. Numerous research studies have shown that agile approach has proven effective for large-scale projects [20]. Bano et al. [21] reported the causes of requirement change and classified them into two categories as accidental and essential. For better understanding, McGee and Greer [22] used a taxonomy, and Saher et al. [23] categorized the causes of requirement change factors into reason and change of origin. Agile requirement in large scale is presented in [24] and analysis and management of conflicts between team are described in [25]. The authors in [26] performed a brief comparative study on agile software development approaches. Knaster and Leffingwell [27] applied scaled agile methodology into system engineering and lean softwares. Putta et al. [28] discussed the challenges and advantages of applying agile methodology SAFe (Scaled Agile Framework) and [29] discussed the scaled agile methodology.

Although numerous studies have been done on agile development and adoption of agile practices, there is a limited research on challenges being faced in RE using agile development strategy and their solutions. Mainly those reviews somewhere and somehow are selective and biased in nature. In [19], the authors discussed RE challenges with only limited reviews, and no case study, survey, or industrial experience has been shared. In their scheme of study, there is biasness, and imprecision and the underlying cause of the challenges reported have not been discussed in detail [30]. In [31], the authors have summarized the results and decisions thus leading to biasness. The existing literature lacks the discussion on RE issues for large-scale projects, and limited reviews are available for RE challenges. In addition to this, no improvement and suggestions are described [32].

The limitations stated above motivated us to conduct a literature review on "Challenges of Requirement Engineering in Agile Software Development." Section 2 presents discussion on agile software developments and requirement engineering. In Section 3, we discuss Requirement engineering challenges for small- to mid-scale agile projects. Section 4 presents requirement engineering challenges for large-scale agile projects. Section 5 focusses on reported challenges, their cause, and potential solutions. In Section 6, generic guidelines to improvise the agile requirement engineering process are presented. Section 7 highlights the limitation of agile development. Lastly, Section 8 concludes the article and provides directions for future research.

According to the literature, many industrial project teams work in an agile environment in order to have rapid delivery of high-quality software for both small- and large-scale projects. The scheme of proposed methodology is represented in Figure 2.

This article presents a detailed review and presents appropriate findings and suggestions. Research aims to

find out the problems and challenges being faced in requirement engineering of small-scale, mid-scale, and large agile projects

design some possible suggestions to improve the general RE process in agile based on the literature

analyze and evaluate the literature in order to propose a suggestion on the future work

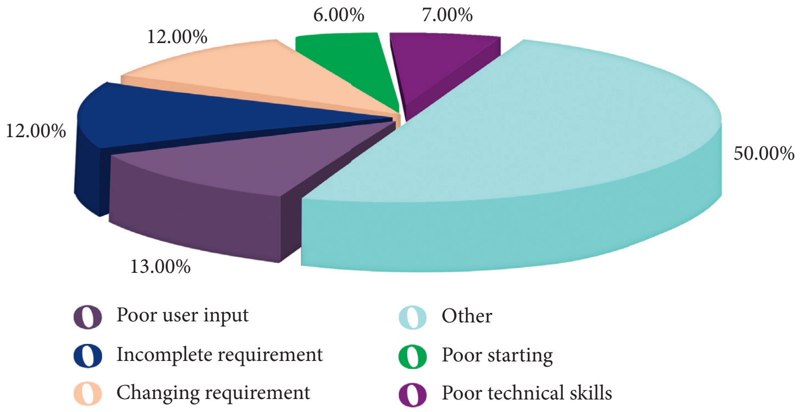


Figure 1: Causes of project failure.

The article aims to address the above stated problems with recent published articles in few years using the agile development with main focus on requirement engineering process and to find challenges involved in agile engineering activities/practices.

2. Agile Software Development and Requirement Engineering

Agile itself is not a method but an approach based on manifesto that mainly focusses on 12 principles and 4 core values. The values of agile manifesto are working software with comprehensive documentation, individuals and interaction among tools and process, customer collaboration, and quick response to a change [33–35]. Agerfalk and Fitzgerald [36] stated that agile development techniques significantly differ from the traditional planned approaches by focusing on productivity rather than following the rigourous processes. According to [37], more than 90% practitioners of agile methodology use user stories to capture requirements and [38, 39] provide how to improve the quality of agile requirements.

ASD is an adaptive process with less documentation, limited resources, and small- to medium-sized highly skilled and motivated team members. The application of agile methods enables the development of reliable and usable software using multiple iterations. Agile addresses the issues of the RE traditional development model as the major challenges are related to requirement and due to the change of business process [40]. Agile development overcomes the issues of the sequential model but still there are some drawbacks associated with it [40].

2.1. Agile Frameworks. De Lucia and Qusef [8] and Malik et al. [40] discussed agile models such as Agile modeling (AM), feature-driven development (FDD), extreme programming (XP), and SCRUM from requirement engineering point of view. The overview of these agile models is listed as follows:

AM is a technique that provides some guidelines related to design models and is used for keeping the documentation and models as minimum as possible [8]. Requirement engineering approach, i.e., brainstorming is supported by some of AM applications.

FDD is a 5-step procedure with some specific entry and exit criteria mainly focused on construction and design phases, building feature list, and planning which are iteratively designed and built by features steps, where feature represents a client valued function [8]. According to Malik et al. [40], the product is developed in terms of individual or separate features instead of whole or full product.

DSDM is the extension of rapid application development and has five principles including user participation, consistent implementation, team decision making, comprehensive life-cycle monitoring, and reversible developmental changes. The initial phases implement feasibility and business study to capture baseline requirements, and further requirements are extracted in subsequent development phase [8]. In the development phase, DSDM supports any requirement engineering approach.

XP is a famous agile method used for successful development of software and supports ease, communication, feedback, and simplicity despite continuously changing requirements. Some XP common practices are as follows: pair programming, fast iterations with small releases, fast feedback, close customer interaction and participation, and so on. In [8], the authors describe the RE activities in detail. XP focusses on coding and development instead of gathering requirements [40]. In XP, the team passes the accountability for the usability of the product to stakeholders [41].

SCRUM is a flexible, adaptable, and project management approach with 30 day sprint cycle. For coordination and integration purpose, 15 minutes daily meeting is conducted [8]. In SCRUM, the developed product is always in working condition and only functionality of the product is increased after every iteration [40, 42].

According to De Lucia and Qusef [8], RE activities such as discovering, analyzing, specifying, and documenting are

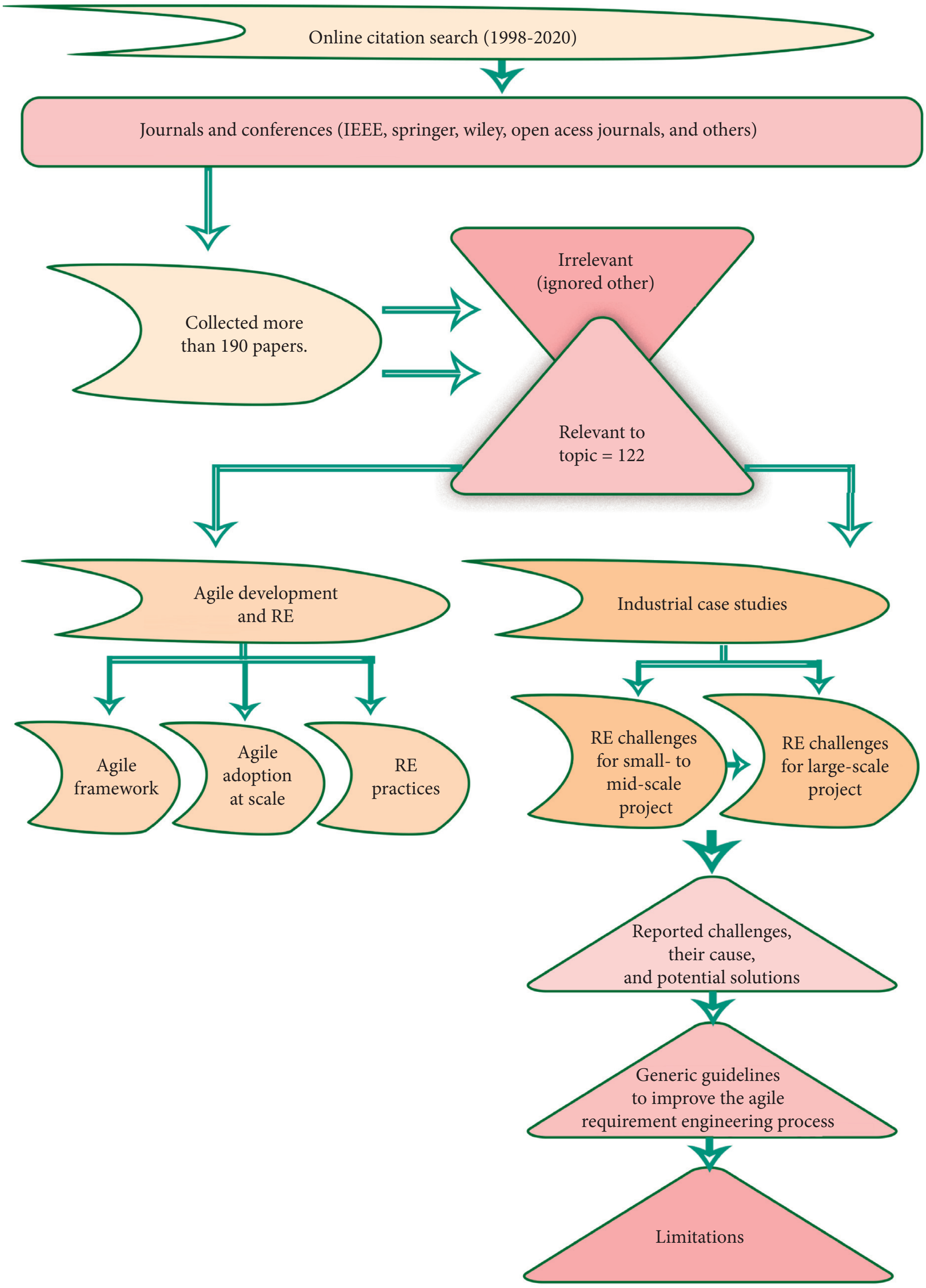

Online citation search (1998-2020)
Journals and conferences (IEEE, springer, wiley, open acess journals, and others)
Collected more than 190 papers.
Irrelevant (ignored other)
Relevant to topic = 122
Agile development and RE
Industrial case studies
Agile framework
Agile adoption at scale
RE practices
RE challenges for small- to mid-scale project
RE challenges for large-scale project
Reported challenges, their cause, and potential solutions
Generic guidelines to improve the agile requirement engineering process
Limitations

FIGURE 2: Scheme of research methodology.

the activities of great importance. The RE activities feasibility study, elicitation and analysis, documentation, validation, and management have also been discussed. Kumar et al. [43] presented a hybrid approach for RE in agile system development by using JAD. Requirement prioritization in agile development is done with the help of viewpoints. High-quality projects are developing in less time using agile development methodologies including iterative development, increments of project, adaptation, collaboration, and emergence. For software requirement gathering, JAD session provides customer specification with collaboration of customers and view point technique is helpful in prioritizing software requirements.

2.2. Agile Adoption at Scale. A large development project can be specified with respect to some criteria used by a company or with absolute measures such as number of developers involved, budget size, complexity, or number of development teams [44]. Positive results and success rate of agile implementation in small projects have created the interest of software development teams and industry to adopt agile in large projects [44, 45] as agile adoption is rapidly replacing the traditional methods [46–49]. In [50], the authors describe how environment helps and supports in distributed agile framework.

The adoption of large-scale implementation of software project may cause project management challenges [51] despite of issues the adoption rate of agile at large scale is increasing rapidly [52, 53]. The survey on management of software development projects is conducted in Oslo, Norway, March 2015. The information about 108 software projects has been examined, and their result demonstrates that agile methodology implementation rate over small, medium, and large projects is successful [54]. The results of agile implementation rate over projects are shown in Figure 3.

On the other side, the adoption of agile at scale has some limitations in some cases, i.e., SCRUM is suitable for small-scale projects with less number of team members, and when it is applied on scaled projects, it makes development less efficient [55]. Abrar et al. [55] identified 21 motivators for the adoption of agile at large scale. Furthermore, their analysis and classification such as into critical factors have been done based on defined criteria. Identified motivators are validated by surveys, questionnaires, and with the empirical study in presence and help of agile software development practitioner and experts [55]. Khalid et al. [56] identified the challenges for SCRUM methodology for distributed environment.

Laanti [57] presented 5 stages of agile transformation for large-scale software development. Salesforce.com completed an agile large project with 200 project members just in the time of 3 months [58]. Bick et al. [59] described five ways of practicing agile in large projects. Turetken et al. [60] discussed the levels of the scaled agile implementation and maturity model. The success rate of agile models in terms of acceptability % and success rate% with respect to budget size of the software project is always greater than the rate of partially agile and not agile methods in each small, medium, and large budgeted projects [54]. The facts summarized above prove and make a way to adopt agile at large scale in software projects.

2.3. Requirement Engineering Practices in Agile Software Development. According to Kovitz [61], RE in agile development is carried throughout the entire development cycle but in small sessions of conservations as compared with phased requirement engineering. In [62, 63], the authors focus on description of important key practices for ASD. Characteristics and principles for large-scale agile development are presented in [64, 65]. Some common agile RE practices are described as follows:

To benefit from agile process, the stakeholders and the team members should communicate directly [66].

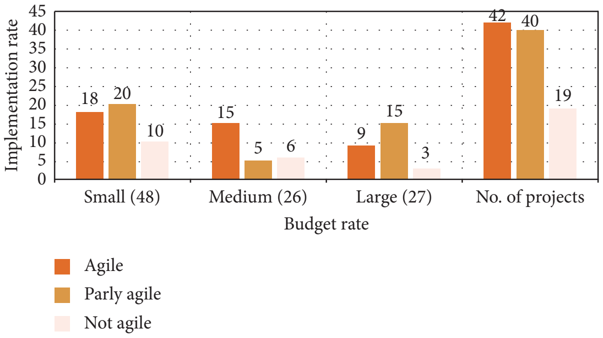


Figure 3: Agile implementation rate over projects.

For successful project and to avoid failure, the main contributing factor is the need for consistent and prioritized requirements by the customer. This emphasizes on the need of client participation and interaction [67].

Requirement emergence, prioritization, and change management including updation of features is important [68].

Continuous planning is required to cope with new and changing requirements, and requirement engineering activities are performed with shared concepts [69].

Use prototype to get feedback from client for requirement refinement and prioritization [8].

Remove an outdated requirement, and for each iteration, prioritize the requirements from beginning [5].

Test first, refactoring, pair programming, small releases, and on-site customer are the fundamental practices that are important to understand the complex skills [61]. Other than this, breaking big things into tiny things, giving up control, writing meaningful test, communication, object oriented design, and usage of advance tools for fast development cycle are considered as important skills in agile development [61]. The common challenges faced in agile projects during the requirement engineering process along with their activities are identified and presented in Table 1.

Rodriguez et al. [70] described some findings related to requirement engineering activities of agile system development. The reported findings are limitations in success of agile development. The intention of this project is to develop a product, so for development of product, some previous details are available. Nonfunctional requirements and management of requirement interrelationship are major issues. A case study of TOPEN (Test Operation Environment) primer development including features, scope, and process by using conventional methodology is briefly discussed. This architecture has four components as TOPEN Engine, Mission Information Base (MIB), gateway, and a GUI TOE. Testing and monitoring of the complex system are done in a specific domain environment by using the conventional method. A list of existing and dropped features of initial product as well as a list of some new functionalities is presented. There are many issues in conventional processes, so they can be more critical for agile development processes.

Table 1: Summary of challenges faced in agile requirement engineering with their activities.

Requirement engineering (phase)	Activities in agile	Challenges	Their impact
Elicitation	Interviews and questionnaires	Problem scoping, misunderstanding, ambiguity in language create flaws	The ill-defined requirements affect requirement analysis
Elicitation	Brainstorming	Group brainstorming	Ambiguous or ill-defined
Elicitation	Prototyping	Security, scalability, and robustness	Maintenance problem
Analysis	Prioritisation	Conflicting ideas	Ambiguity
Documentation	User stories	Irrelevant user stories, unavailability of customer representatives	Mislead implementation
Documentation	Product backlog and index cards	Minimal documentation	Loss of knowledge
Validation	Feedback from clients	Lack of proper validation methods and tools, prototyping issues	Low quality
Management	Change and version control	Selection of an appropriate tool	Wastage of time
Management	Requirement traceability	No effective way of requirement management and traceability	No traceability

3. Requirement Engineering Challenges for Small- to Mid-Scale Agile Projects

Agile development overcomes the issues of the sequential model but still there are drawbacks associated with it [40]. According to Malik et al. [40], the challenges of RE are incompatible interfacing, negligence of nonfunctional requirement, lack of unambiguous, and RE activates. Agile approaches do not have any proper framework by which the user requirements are properly listed or documented, and due to changing requirements, extensive amount of work needs to be done repeatedly. Sometimes due to changing needs of the customer, the developed product in previous iteration causes interface compatibility issues. There is no rule or technique for gathering and evaluation of nonfunctional requirements. The quality of the product being checked by the user's feed back after iteration is complete [40]. Sebega and Mnkandla [6] discussed the issues of South Africa software industry. The challenges reported by the authors are as follows:

In requirement elicitation: lack of clarity, requirement prioritization, and problem scoping

In requirement management: prioritization issue

In requirement documentation: lack of proper documentation and no customer representative is available

In requirement validation: no techniques and tools for validation process

In requirement management: lack of requirement change management and appropriate management tools

Lack of proper documentation may create some issues for the team; hence, De Lucia and Qusef [8] suggested some guidelines for their long-term use. These guidelines are assignment of some people to produce little documentation, use of computer tools for modeling, and development of reverse engineering techniques. According to Alam et al. [19], less direct communication and insufficient documentation cause requirement tracking issue and less feedback and interaction of client, handling requirement change causes more work and estimation of time and goes out of plan, and creative and innovative requirement in agile itself is challenging, missing, and conflicting requirements which causes large number of iterations and no focus on nonfunctional requirements. Hajjdiab and Al Shaima Taleb [16] identified the challenges faced in SCRUM. The challenges faced in requirements are no team involvement in initial project estimation, impact of ambiguous requirements on quality and schedule, minimum documentation, different levels of abstraction, and inconsistency of user stories in product backlog when product owner is appointed. According to Amir et al. [71], due to changing requirement, the agile project becomes expensive, so project cost and maintenance cost are increased. Minimal documentation in requirement gathering process creates issues such as no staff help and provides no support for reverse engineering practices. Lack of documentation creates other issues when the project is transferred to maintenance, when staff changes, or when new team or staff is hired [72].

Cao and Ramesh [68] stated that in agile approach, the RE focusses more on requirement validation but there is no formal verification of requirements as there is no model for formal verification of detailed requirements in agile. For requirement prioritization, the only criteria are business value as time to market is important factor which can create major issues later. In agile RE, the use of prototyping creates the wrong understanding in the stakeholder's mind about development speed. Too much implementation in initial prototyping causes issues such as quality concerns generated by the practice of reusing code in prototype, and stakeholders may not accept other development cycles which are more scalable and robust. According to Ganesh and Thangasamy [73], the challenges of RE lie through clear communication and information sharing, domain knowledge, and less documentation on software requirements.

Primary reason is that the client and whole team are no longer available for daily face-to-face meetings. No regular communication is possible, so minimal specification causes many issues such as difficulties for quality assurance team and loss of knowledge either domain or product.

Cho [74] discussed the challenges and issues of SCRUM (agile development model) through a case study of a company that used SCRUM in their web-based project from small- to mid-sized projects. In reported issues, the developer said that tricky code requires more explanation and for the changes they made. Whereas many developers said they are easy without documentation. Developers stated the issues regarding the importance of specification, such as no shared knowledge, what if the person who has all information leave company, and some other important team member [74]. Stakeholders are not involved in the decision-making process, and the customer is not readily available for regular meetings which leads to a communication gap. According to Helmy et al. [75], agile approach relies on the basic assumption that getting and eliciting all the requirements from user as they evolve with time so customer changed their requirements and mind. Thus, no one knows full requirements of the product at start of the product that causes missing an interface between requirements and causes more rework in next iteration. Other reported challenges are lack of RE activities and no focus on nonfunctional requirements [75]. According to Helmy et al. [75], the issues related to nonfunctional requirements should be identified and team should adopt the nonfunctional requirements to improve the overall product quality in each iteration; otherwise, it may also cause security and scalability-related issues, which will deeply affect the final version of the product.

According to Daneva et al. [76], in requirement prioritization, users play an important role as delivery story which include important technical details and which are created with collaboration of users. Thus, inability of user such as users are not able to tell exactly what they need and what they actually want to say due to high authority pressure, environmental facts, or hesitation and less representation among their group create issues in creating delivery stories. In agile, customers are not involved in decision making process and it creates concerns related to requirement prioritization. Rahim et al. [77] stated that requirement prioritization in agile approach after each iteration is challenging in absence of any technique, and when each project requirement demands specific consideration of each aspect, it causes starvation problem. Shameem et al. [78] stated that due to rapid development rate and less development cost, software organizations are focusing on applying agile methods in distributed environments which is known as distributed software development (DSD). This study involves distributed agile practices and is focused on identification of intended challenges and their priorities in context of agile development in distributed environment. Literature review was conducted to find out these challenges, and they were further divided into four categories such as team, process, current technology, and management. Comparison of challenges of two studies is performed, and prioritization of challenges on major four categories is done by using the APH method.

4. Requirement Engineering Challenges for Large-Scale Agile Projects

The use of agile strategies is being so popular in software industry due to its numerous benefits. In complex software development projects as supply chain management having unstable and uncertain requirement, development is difficult if the predictable process model is used [79]. As agile strategies are adaptive, they can be used, but using the agile methodologies helps and improves one aspect in large-scale software development (LS-SD) while creating issues in another part. Mishra and Mishra [79] reported RE-related issues adopting agile practices in LS-SD, such as long duration of RE due to complex decision making process, difficulty in creating and maintaining requirement priority list, and waiting time of requirements. Dikert et al. [80] discussed the challenges and success element that influence large-scale agile transformation. The reported 35 challenges are categories in to lack of investments, implementation challenges, RE challenges, quality assurance challenge, and others. The listed RE challenges are missing high-level requirement management, refinement challenges, gap between short- and long-term planning, and hard to create and estimate user stories [80]. In large development projects, the understanding and management of high-level requirements are necessary. So, handling high-level requirements and understanding are important as it is well known that ambiguity in requirements affects the quality. Bjarnason et al. [81] presented a case study of CASE company which develops embedded systems with 60 to 80 new features, 700 to 1000 detailed requirements, and with lead time of 2 years. Agile practices, their advantages, and challenges being in agile requirement engineering (ARE) are reported. The reported challenges using agile practices for large-scale software projects are planning, weak requirement in start, weak requirement prioritization and effort estimation, quality issues, customer proxy role, taking innovation, ensuring RE, validation and verification, motivation for requirement work, lack of requirement documentation and less preliminary planning, and focus that can lead to rework [81]. Missing the scope of requirements in start causes the difficulty in estimation of project schedule and budget.

Large-scale agile projects with four iterations and some specified roles using the SCRUM model that aimed to be more flexible and to avoid specifying details are signed by GOV in order to modernize their existing legacy system [82]. According to Rolland et al. [82], in this replacement project, a major challenge is to scale product owners so that they manage to facilitate the requisite `translation` of user stories, which are usually multiple, including complex interdependencies and relatively high-level development teams. According to literature [7], in agile development, the process of RE is not just restricted in beginning such as in traditional models but it continues till whole development cycle. Fægri and Moe [7] presented a finding from a case study of large agile-based project. According to Xu [83], in continuous changing requirements, technology, and environment, there is a need to focus on agile methodologies to design and develop a software on time with less defects and early releases thereby achieving customer satisfaction. Xu [83] described coordination challenges faced in large-scale agile projects and discussed the coordination in terms of decision making, communication, and control mode. As the decision making process is difficult for developers using informal way of communication, the problems faced are participant's lack of interaction, communication problems, information loss, difficult and unstable requirements, complex interdependence tasks, and technical problems.

Kasauli et al. [84] have discussed requirement challenges through multiple case studies of four companies for large-scale system development. Among those, two are car manufacturers, one is telecommunication company, and last one is technology-based company. In LS-SD, scope of the agile method and role of RE are briefly discussed. Different cases of companies are discussed, and result of each case is concluded. Main challenges of requirement engineering of agile system development for large scale of case 1 are related to communication and knowledge management. Other challenges are also discussed which are related to two areas of requirement knowledge such as shared value of user and creating and maintaining system understanding for Case 2. In Case 3, main challenges found are related to interplay of stakeholder from different domains. Those domains are customer, integration, and testing and development domains. In studied case of case companies, requirements specification is needed to better understand the current system for analysis of new feature requests and maintenance of the system.

Lindvall et al. [85] demonstrated that organizations have identical needs and need to experience effect of change before adopting new methodologies in large organizations. This is just because of their complex nature and need to transform current technologies and processes with existing technologies. In [85], the authors tried to establish their results by analyzing their experience in specific framework of large organizations by using well-structured processes. Experience collected on basis of analysis is then shared among SEC members which are recognized as Nokia, ABB, Daimler Chrysler, and Motorola, and their areas of interest are also discussed in detail. Moreover, data across 15 pilot projects of these four organizations are affected by extreme programming that gives a wider overview of implementation of agile methods in large companies. Issues related to implementation of agile in large companies are also discussed. Developers identified requirement problems as powerful drivers among these four organizations. It is difficult to split requirements into detailed specifications on delivery of these requirements to high-level development teams.

5. Reported Challenges, Their Cause, and Potential Solutions

According to Lee et al. [86], the problems with software are as follows: (1) it is rarely delivered on time due to schedule slippiness which causes cancellation; (2) poor softwares which are not able to solve the right problem due to the reason that the software requirement is not according to the need and does not conform; (3) staff burn-out when they leave or when areas of interest; and (4) small changes require huge effort. The following are the challenges we have identified through the literature; and their summary is presented in Table 2; and Table 3 presents a summary on RE challenges faced in large-scale agile development. Figure 4 shows the summary of this section.

5.1. Less Direct Communication with Clients/Stakeholders. Less direct communication causes requirement trackness issue [19]. It may occur due to many factors such as lack of time, distance factor, less customer representatives, and so on. From business perspective, customer involvement and availability directly affect the requirement change and validation. In decision making process, no user involvement causes requirement prioritization. Missing high-quality user interaction results in wrong requirements [68, 96].

How to Deal. To deal this issue, direct communication, surveys, and interviews should be conducted. While gathering requirement from client do not focus on collecting long-term requirements and do not go with long formal interview sessions [40]. Use surveys extensively to get feed back from customers.

5.2. Minimum Focus on Documentation. Lack of proper documentation may create some issues for development team [8] and cause requirement trackness issue [19]. Minimal documentation in requirement gathering process creates issues like no staff help and provides no support for reverse engineering practices [71, 72]. Minimal specification causes many issues such as difficulties for quality assurance team and lack of knowledge either domain or product [73]. This will create more impact on large projects.

How to Deal, Do some documentation, and more focus and consideration should be taken as this will help to track, mange, and test the requirements. Hence, there is need to standardize the agile process and RE activities.

5.3. Missing, Ambiguous, and Conflicting Requirements. The different levels of abstraction and inconsistency of user stories cause problems. So, the impact of ambiguous requirements on quality and schedule is very severe [16]. Missing interface between requirements causes more rework in next iteration [75].

How to Deal. The use of more formal ways for specification of requirements is needed to incorporate the missing, ambiguous, and conflicting requirements. Use more formal ways for specification of requirements to incorporate the changing requirements. RE-KOMBINE is a model that is used to specify requirements in formal way and is more flexible to handle change [87].

Table 2: Reported challenges, their cause, and potential solutions from the literature.

RE challenges	Resources	Suggestions
Direct communication with stakeholders	[16, 19, 74, 76]	Direct communication, surveys, and interviews
Lack of documentation	[6, 8, 71, 73]	Do some documentation, more consideration, and focus on documentation should be taken
Missing, ambiguous, and conflicting requirements	[16, 75]	Use more formal ways for specification of requirements to incorporate the missing, ambiguous, and conflicting requirements [87]
Problem of prototyping	[68]	Avoid much implementation in initial prototyping, use paper prototyping instead of wizard of Oz prototyping
Less preliminary planning, focus, and no initial team involvement	[16, 81]	Engage and involve all team members of the project from the very beginning such as development team
Less experienced and skilled team	[88–90]	Train practitioner [89], use supportive tools
Inability of customers in telling user stories	[76]	Train customer and involve them in decision making process
Tacit knowledge	[73]	Direct communication, observation, surveys, interviews, and the three factors: domain expert, communication process, and audience will help [40]
Changing requirements	[6, 16, 40, 71, 75]	Use agile requirement change management readiness model (ARCMRM) [91], adopt 21 progress and success factors of ARCM [92], automated techniques, and tools to manage the requirement issues, i.e., JIRA and RE-KOMBINE [40, 71, 87]
Requirement prioritization	[74, 77]	Four stages based frame has been proposed [93]
Negligence of nonfunctional requirements	[40]	Simulation tool NORMATIC for NFRs modeling [67], planning and visualization procedure, [94], NFR elicitation process [95]

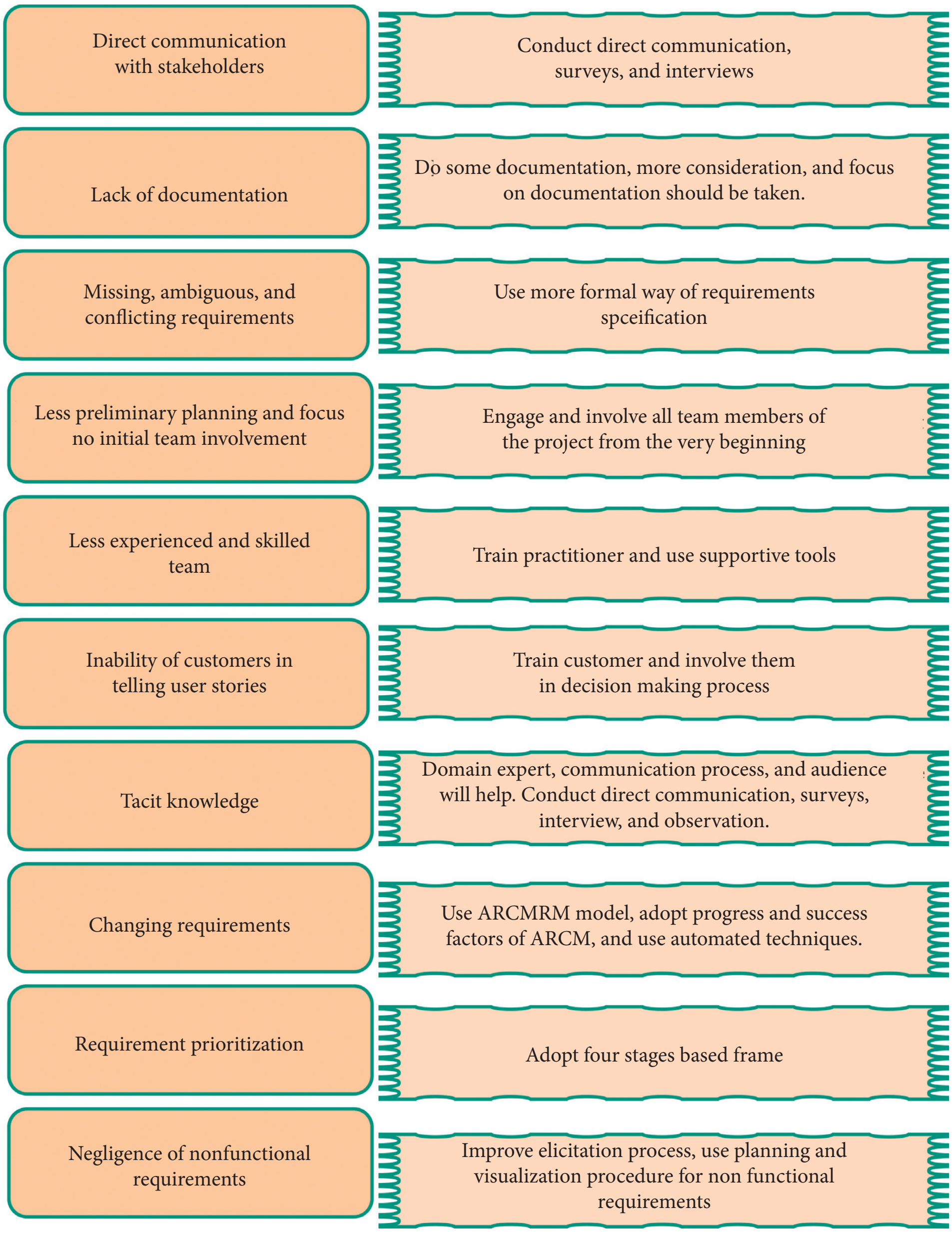
Table 3: RE challenges related to large-scale projects.

Authors	Type of project/source	Reported challenges
Dikert et al. [80]	Different cases of companies using different agile models in projects, i.e., banking, telecommunication, and distributed system	Missing high-level requirement management, refinement challenges, gap between short- and long-term planning, hard to create and estimate user stories
Mishra and Mishra [79]	Complex software development projects such as supply chain management having unstable and uncertain requirement	Long duration of RE due to complex decision making process, difficulty in creating and maintaining requirement priority list
Bjarnason et al. [81]	A case study of CASE company which develops embedded systems	The reported challenges using agile practices for large-scale software projects are planning, weak requirement in start, weak requirement prioritization and effort estimation, quality issues, customer proxy role, taking innovation, ensuring RE, validation and verification, motivation for requirement work, lack of requirement documentation
Rolland et al. [82]	SCRUM model-based large-scale project signed by GOV in order to modernize their existing legacy system	Scale product owners so that they manage to facilitate the requisite "translation" of user stories, which are usually multiple, including complex interdependencies, and relatively high-level development teams
Xu [83]	Discuss coordination challenges faced in large-scale agile projects	The reported challenges are lack of coordination in terms of decision making, communication, and control mode

5.4. Problem of Prototyping. According to Corral and Fronza [97], we can choose design thinking based on empathize, define, ideate, prototype, and testing than typical approaches. The RE agile development and the use of prototyping create the wrong understanding in the stakeholder's mind about development speed. Too much implementation in initial prototyping causes issues such as quality issues generated by the practice of reusing code in prototype and stakeholders may not accept other development cycles which are more scalable and robust [68].

How to Deal. Avoid much implementation in initial prototyping. It is better to use paper prototyping instead of wizard of Oz prototyping which actually reduces your effort and remove the misconception of user by seeing wizard prototype that you implemented all so fast. Paper prototyping in this case will help in many ways such as acts as a design test before you code, quickly changeable, and eliminates the specific variables of technology [98].

5.5. Less Preliminary Planning, Focus, and No Initial Team Involvement. In start of development, RE with less preliminary planning and focus can lead to rework; moreover, missing the scope of requirements in start causes the difficulty in project estimation of schedule and budget [81]. No team involvement in initial project estimation causes wrong estimation of project cost and schedule [16].



Direct communication with stakeholders	Conduct direct communication, surveys, and interviews
Lack of documentation	Do some documentation, more consideration, and focus on documentation should be taken.
Missing, ambiguous, and conflicting requirements	Use more formal way of requirements spceification
Less preliminary planning and focus no initial team involvement	Engage and involve all team members of the project from the very beginning
Less experienced and skilled team	Train practitioner and use supportive tools
Inability of customers in telling user stories	Train customer and involve them in decision making process
Tacit knowledge	Domain expert, communication process, and audience will help. Conduct direct communication, surveys, interview, and observation.
Changing requirements	Use ARCMRM model, adopt progress and success factors of ARCM, and use automated techniques.
Requirement prioritization	Adopt four stages based frame
Negligence of nonfunctional requirements	Improve elicitation process, use planning and visualization procedure for non functional requirements

Figure 4: Improvements and suggestions to deal with faced challenges.

How to Deal. It is better and helpful to engage and involve all team members of the project from the very beginning such as development team so that they know all about the project specification, their time to time changes, and technical details from the start. The one and important reason to engage and focus on all the teams members is that since there is minimum documentation, their preliminary planning, focus, and involvement are very beneficial. It is the responsibility of the project manager to monitor the presence of team members mentally by engaging them into multiple activities during questionnaires and interviews.

5.6. Less Experienced and Skilled Team. According to Hufmann and Lehner [88], requirement miss-interpretation, miss-communication, and inadequate requirement analysis are the result of less experienced and skilled team which in turn causes low rate of project success and other problems. According to Morales-Ramirez and Alva-Martinez [89], the persons/team involved in activities build wrong feature software products due to less experience and lack of knowledge [90].

How to Deal. Train practitioners to improve their skills by giving them materials, training sessions, and guidelines on how to perform on particular thing [89]. The training plan is applied on practitioners working at a Mexican SDO with grouped set of skills. Moodle platform is used as a supportive tool.

5.7. Inability of Customers in Telling User Stories. Inability of customer while describing user stories causes rework and creates load and extra charges because of incomplete domain knowledge among user groups [76] and hence creates issues in requirement prioritization.

How to Deal. To overcome this problem, we have to train the customer and involve them in decision making process throughout the process of requirement engineering.

5.8. Tacit Knowledge. The knowledge transferred by the business to the development team is a challenge because this information is not listed in specification [73]. Tacit knowledge is what we know but we cannot tell; it is what is known and acquired through personal experience, so sometimes it is difficult to articulate them [99]. According to [100], in tacit knowledge, there is risk associated with requirement misunderstanding and anticipation, and it is very difficult to involve the stake holders from every culture, field, and profession.

How to Deal. Direct communication, observation, surveys, and interviews frequently reduce the likelihood of tacit knowledge. According to [100], the three factors such as domain expert, communication process, and audience will help to encounter the issue of tacit knowledge. It is more appropriate to break the meeting into smaller sessions as it will help participants to digest knowledge between the session and they feel more comfort. The receiver of knowledge (audience in meeting) must have good listening skills [100]. Enable practice for knowledge sharing of positive experience as well as negative experience which helps team members and they will learn from it as sharing knowledge is a key success factor in development [101].

5.9. Changing Requirements. Sometimes due to changing need of the customer, the developed product in previous iteration causes interface compatibility issues [40]. Due to changing requirements, the agile project becomes expensive; thus, the project cost and its maintenance cost increased [71]. Failure towards managing the changing requirements causes system failure [102].

How to Deal. Kamal et al. [91] proposed ARCM-RM (agile requirement change management readiness model) which comprises of three main components such as maturity level, factor level, and assessment level. This model also supports global software development for measuring and improving ARCMRM. Kamal et al. [92] identified 21 key success and progress factor of ARCM, and the findings and results show that they are beneficial for practitioners for management of change in global software development context. Moreover, use automated techniques and tools to manage requirement issues, i.e., JIRA and RE-KOMBINE [40, 71, 87].

5.10. Requirement Prioritization. Requirement prioritization in agile approach after each iteration is challenging due to absence of any technique [77] as stakeholders are not involved in decision making process and customer is not readily available for daily meetings [74]. According to [103], there are almost 4 weeks or more time lapse between the delivery of each deliverable in incremental approaches. In this time period, change in market demand or requirement changing due to personal reason, these factors affect the process of requirement prioritization. There are many unseen and unobserved factors in requirement prioritization, and they have great impact and influence on the result of prioritization [104].

How to Deal. AbdElazim et al. [93] presented a framework to prioritize the requirements in agile development. The framework consists of four stages: identification, verification, estimation, and prioritization. The presented model will also help to handle change in any stage of the software development process [93].

5.11. Negligence of Nonfunctional Requirements. There is no rule or technique for gathering and evaluation of nonfunctional requirements (NFRs), for example, agile methods, extreme programming, and SCRUM are popular as they deliver quality functional requirements. The quality of the product being developed is checked by the user's feedback after each iteration is completed [40]. Negligence of nonfunctional requirements delays the project delivery as well as deeply affects the quality of the whole product.

How to Deal. Farid and Mitropoulos [67] proposed the java-based simulation tool NORMATIC for nonfunctional requirement modeling in agile development process. Planning and visualization procedure is presented to implement the NFRs in agile projects [94]. Younas et al. [95] discussed about the user story cards, their attributes, and agile NFR elicitation process. The authors [95] analyzed the process of elicitation of NFR by conducting a case study on master students of SED (software engineering department) and obtained the encouraging output.

6. Generic Guidelines to Improve the Agile Requirement Engineering Process

Many guidelines and techniques are presented by many researchers such as agile adoption techniques, agile principle, guidelines, and assumptions for enterprise or large distributed projects and others [82, 105, 106]. Some of them perform analysis and listed success factors [107]. In Figure 5, some generalized suggestions are shown to improve the process of requirement engineering which will help to incorporate challenges of agile development. The generic guidelines are shown in Figure 5.

6.1. Direct Communication, Surveys, and Interviews. While gathering requirements from client, do not focus on collecting long-term requirements and do not go with long formal interviews sessions [40]. Use surveys extensively to get feed back from customers. Frequent face-to-face communication from very beginning for identification of requirement is necessary and keeps the project from cost and schedule overrun [108]. According to [109], when we choose

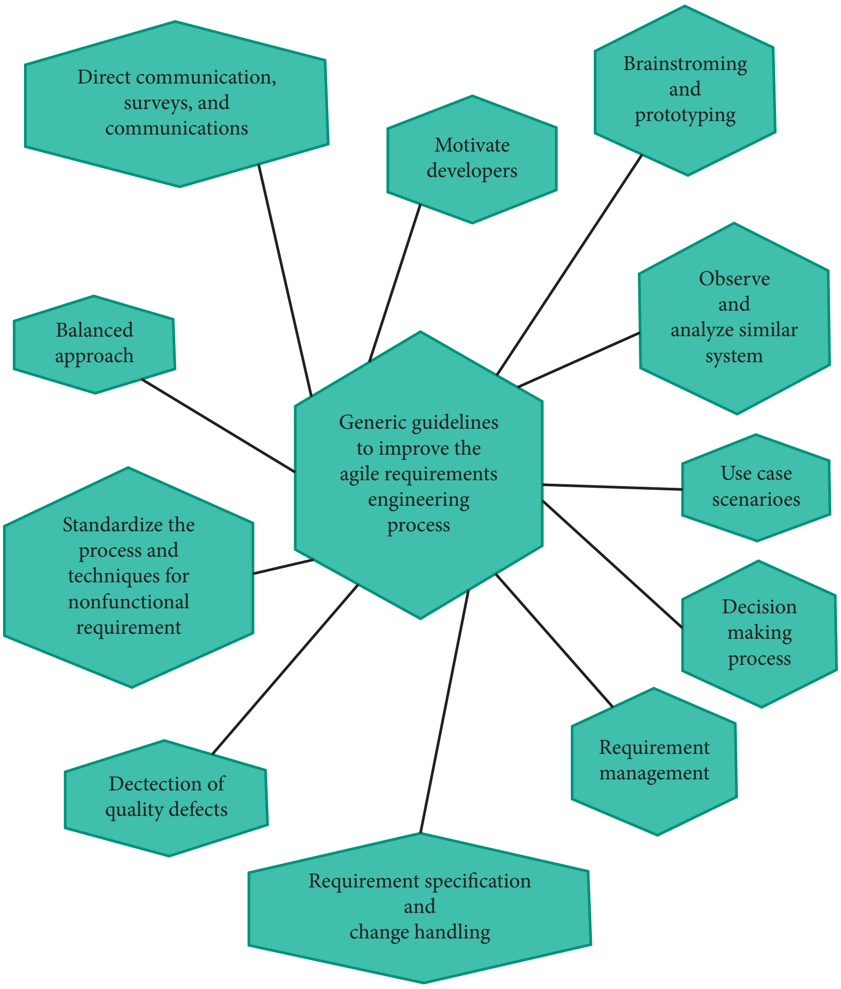

Direct communication, surveys, and communications
Motivate developers
Brainstroming and prototyping
Observe and analyze similar system
Balanced approach
Generic guidelines to improve the agile requirements engineering process
Use case scenarioes
Standardize the process and techniques for nonfunctional requirement
Decision making process
Requirement management
Dectection of quality defects
Requirement specification and change handling

Figure 5: Generic guidelines to improve the agile RE process.

and increase reliance on linear communication medium, accordingly the software defect rate will also increase. So, there is a need to pay more attention while choosing the communication medium when stakeholders are partially available.

6.2. Motivate Developers. Gather and share positive and successful stories and experience to motivate developers [16]. Motivation is considered as an important factor for the developers' productivity and is very difficult to quantify [110]. Highly motivated developers are more productive, and thus there are more chances of project success and vice versa. Hence, it can be concluded that the project outcome is correlated with motivation [110]; therefore, it is the responsibility of project manager to motivate the developers by giving them credits, appreciation, value, and regards. To motivate developers, Arai et al. [111] proposed a gamified tool to remove warning which helps in detection of bugs in early software development.

6.3. Brainstorming and Prototyping. To cater the issue of creative requirements from customer, individual and collaborative brainstorming techniques are used [40]. According to [96], the use of prototyping can solve many RE challenges as unambiguous documentation by developing executable prototypes, provide motivation of RE as updating prototype is easy rather than documenting, and improve communication among stakeholders as it serve as a visual medium and can capture real needs and results.

6.4. Observe and Analyze Similar Systems. To handle requirement change, observe and analyze the requirements of similar products that are already built because sometimes there is no enough time to brainstorm, gather, and analyze the requirements [40]. In software development industry, projects are related to one another to some extend by their nature or by features; hence, observation and analysis skill will help in many ways. If you have already experienced and developed some similar systems, then you have greater ability to tackle the issues, i.e., reusability of already built components and reusability of requirements and lot of others. Observing and analyzing similar systems will improve the ideas and the skill sets of persons.

6.5. Use Case Scenarios. Agile framework and methodologies use simple way of requirement gathering and specification in the form of user stories, which are easy to read but difficult when we talk about visual representation and the understanding of the system [112]. Wautelet et al. [112] bridged user stories to the use case (UC) model. Although it is very lengthy process, in agile, it can be very effective to use user stories and use case scenarios to capture requirements if customer is readily available [40].

6.6. Decision Making Process. Improve the decision making process and involve stakeholders in decision making process to avoid requirement prioritization issues. Use the requirement prioritization model and techniques in agile for better understanding and for ease of use which are scalable, save time, and customizeable [77].

6.7. Requirement Management. To ensure the integrity of the user information or requirements in agile, there should be proper mechanism or little documentation should be produced as in agile there is no focus on documentation and specifications. The process model for requirement change management should be used as it provides better communication, user understanding, process improvement, and management [113]. Requirement management metrics should be used [10], and tool support for large-scale agile projects should be used [114]. According to Hannay et al. [115], to handle and structure requirements, service-oriented capability taxonomies should be used.

Albuquerque et al. [116] stated that there are a lot of ways to enhance the requirement change management in agile system development. They further proposed a study which describes agile requirement change management (ARCM) process and activities associated to this process in order to manage the requirement change in agile modeling. A systematic mapping is used to control this process and identified 11 different agile practices to proceed the process. Process of ARCM has 3 main steps in which first is identification of change, second is analysis of identified change, and third is cost or effort estimation. These processes are further discussed in detail, and each process has distinct requirement engineering practices which help ARCM to enhance change in requirements.

6.8. Requirement Specification and Change Handling. Use more formal ways for specification of requirements to incorporate the changing requirements. RE-KOMBINE is a model that is used to specify requirements in formal way and is more flexible to handle change [87]. JIRA is also helpful to deal with agile requirements challenges and is more helpful for complex projects [117].

According to Albuquerque et al. [116], the process of agile requirement change management consists of requirement change identification, analysis, cost, and effort estimation to handle and manage change in requirements. AbdElazim et al. [93] presented a framework to prioritize the requirements in agile development which consists of four stages: identification, verification, estimation, and prioritization. The presented model will also help to handle change in any stage of the software development process [93].

6.9. Detection of Quality Defects in Requirements. AQUSA (Automatic Quality User Story Artisan) software tool based on natural language processing (NLP) should be used to find quality defects in requirements as it detects defect in requirements and suggests some possible solutions. Defect detection in early iterations will prevent the defects to appear in next iteration, and therefore we get better quality product [118].

6.10. Standardize the Process and Techniques for Nonfunctional Requirements. There is a need to standardize the agile process and RE activities and to provide a way to elicit and manage the nonfunctional requirements in order to improve the overall product quality in each iteration.

6.11. Use Some Balanced Approach. The reported solution to deal with common challenges faced in large-scale project is to create a balance between the agile and traditional approach. In large and complex distributed projects, agile without structure can cause chaos particularly where planning, control, and teamwork are crucial. Structure without agile concepts can result in rigidity, especially when a project needs extensive learning, innovation, and change [119]. According to [120], documentation in agile development is important to support the communication with stakeholders and it has more importance in distributed development and conversations. Hence, agile methods with zero documentation are not recommended. According to [83], in large agile projects, as the scale of the projects grows, more formal approaches are required such as centralization, vertical communication, impersonal communication, and formal control to remove coordination relates issues.

The common challenges associated with agile RE activities are requirement elicitation, clarification, analysis, prioritization, documentation, and decision [31]. According to [121], agile software development process also integrates and assures the security of the software. The authors proposed a method to develop acceptably secure software providing the assurance of security, and the proposed method claims partially mitigation of associated threats. Rodriguez et al. [70] described some findings related to requirement engineering activities of agile system development. These findings are limitations in success of agile development. The intention of this project is to develop a product, and for development of product, some previous details are available. A list of existing and dropped features of initial product as well as a list of some new functionalities is presented. There are many issues in conventional processes, so according to author, they can be more critical for agile development processes.

7. Limitations

Though the agile approach has preformed the traditional software development approaches and has numerous benefits, it still has some drawbacks which are highlighted in this section as follows:

No reusability of artifacts

Lack of support for distributed environment

Not suitable for development of safety critical systems

Development of large projects with implementation of agile approach is more difficult [122]

The analysis of agile requirement engineering with respect to different agile frameworks or methodologies is not discussed comprehensively. We do not claim that the presented study is applicable universally on all development phases and methodologies. Our choice is limited in terms of agile requirement engineering, its methodologies, challenges being faced in agile requirement, and generic suggestions.

8. Conclusion and Future Work

Requirement engineering is critical and vital phase of software development life cycle. There are numerous challenges related to requirement engineering in agile development process. Though agile requirement engineering process provides benefits over traditional development cycles due to its features such as minimal documentation, quick implementation strategies, incorporating changing requirements, and frequent feedback from customers, it also has challenges and limitations. These challenges create adverse effects on product being developed such that quality and schedule of the product are affected. In this article, we have identified the issues and challenges of RE in ASD by reviewing recent state-of-the-art from the literature.

The formulated challenges and suggestions based on this study can be used to enhance the skills and abilities of the development companies during agile software development. Our research can serve as a baseline for the researchers who intend to work in this direction. It is very beneficial for research contributors who want to work in this study area which require practical results and studies with many approaches of agile such that crystal and extreme programming. Conflicts and challenges arise when the process is not defined. Hence, the conclusion drawn from this study is that the agile domain is still immature, and the need for establishing guidelines and standardization of the process is crucial to improve outcomes. Thus, there is need of guideline and standardization of process.

Data Availability

The data used to support this study are included within the manuscript.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] V. Szalvay, "An introduction to agile software development," *Danube Technologies*, vol. 3, 2004.

[2] S. Balaji and M. S. Murugaiyan, "Waterfall vs. V-Model vs. Agile: a comparative study on SDLC," *International Journal of Information Technology and Business Management*, vol. 2, no. 1, pp. 26–30, 2012.

[3] M. Ochodek and S. Kopczyńska, "Perceived importance of agile requirements engineering practices - a survey," *Journal of Systems and Software*, vol. 143, pp. 29–43, 2018.

[4] X. Wang, L. Zhao, Y. Wang, and J. Sun, "The role of requirements engineering practices in agile development: an empirical study," in *Requirements Engineering*, pp. 195–209, Springer, Berlin, Germany, 2014.

[5] A. Sillitti and G. Succi, "Requirements engineering for agile methods," in *Engineering and Managing Software Requirements*, pp. 309–326, Springer, Berlin, Germany, 2005.

[6] Y. Sebega and E. Mnkandla, "Exploring issues in agile requirements engineering in the South African software industry," *The Electronic Journal of Information Systems in Developing Countries*, vol. 81, no. 1, pp. 1–18, 2017.

[7] T. E. Fægri and N. B. Moe, "Re-conceptualizing requirements engineering: findings from a large-scale, agile project," in *Proceedings of the Scientific Workshop Proceedings of the XP2015*, pp. 1–5, New York, NY, USA, May 2015.

[8] A. De Lucia and A. Qusef, "Requirements engineering in agile software development," *Journal of Emerging Technologies in Web Intelligence*, vol. 2, no. 3, pp. 212–220, 2010.

[9] A. Hussain, E. O. Mkpojiogu, and F. Mohmad Kamal, "The role of requirements in the success or failure of software projects," *International Review of Management and Marketing*, vol. 6, no. S7, pp. 306–311, 2016.

[10] S. A. Kumar and T. A. Kumar, "Study the impact of requirements management characteristics in global software development projects: an ontology based approach," *International Journal of Software Engineering & Applications*, vol. 2, no. 4, p. 107, 2011.

[11] P. Kettunen and M. Laanti, "Combining agile software projects and large-scale organizational agility," *Software Process: Improvement and Practice*, vol. 13, no. 2, pp. 183–193, 2008.

[12] S. Nerur, R. Mahapatra, and G. Mangalaraj, "Challenges of migrating to agile methodologies," *Communications of the ACM*, vol. 48, no. 5, pp. 72–78, 2005.

[13] T. Dingsøyr, N. B. Moe, T. E. Fægri, and E. A. Seim, "Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation," *Empirical Software Engineering*, vol. 23, no. 1, pp. 490–520, 2018.

[14] L. Lagerberg, T. Skude, P. Emanuelsson, K. Sandahl, and D. Ståhl, "The impact of agile principles and practices on large-scale software development projects: a multiple-case study of two projects at ericsson," in *Proceedings of the 2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, pp. 348–356, IEEE, Baltimore, ML, USA, October 2013.

[15] J. Eckstein, *Agile Software Development in the Large: Diving into the Deep*, Addison-Wesley, Boston, MA, USA, 2013.

[16] H. Hajjdiab and A. S. Al Shaima Taleb, "Adopting agile software development: issues and challenges," *International Journal of Managing Value and Supply Chains*, vol. 2, no. 3, pp. 1–10, 2011.

[17] H. Berger and P. Beynon-Davies, "The utility of rapid application development in large-scale, complex projects," *Information Systems Journal*, vol. 19, no. 6, pp. 549–570, 2009.

[18] B. Hobbs and Y. Petit, "Agile methods on large projects in large organizations," *Project Management Journal*, vol. 48, no. 3, pp. 3–19, 2017.

[19] S. Alam, S. A. A. Shah, S. N. Bhatti, and A. M. Jadi, "Impact and challenges of requirement engineering in agile methodologies: A systematic review," *International Journal of Advanced Computer Science and Applications*, vol. 8, 2017.

[20] D. Šmite, N. B. Moe, and P. J. Ågerfalk, "Agility across time and space: summing up and planning for the future," in *Agility Across Time and Space*, pp. 333–337, Springer, Berlin, Germany, 2010.

[21] M. Bano, S. Imtiaz, N. Ikram, M. Niazi, and M. Usman, "Causes of requirement change-a systematic literature review," in *Proceedings of the 16th International Conference on Evaluation & Assessment in Software Engineering (EASE 2012)*, Ciudad Real, Spain, May 2012.

[22] S. McGee and D. Greer, "Towards an understanding of the causes and effects of software requirements change: two case studies," *Requirements Engineering*, vol. 17, no. 2, pp. 133–155, 2012.

[23] N. Saher, F. Baharom, and O. Ghazali, "Requirement change taxonomy and categorization in agile software development," in *Proceedings of the 2017 6th International Conference on Electrical Engineering and Informatics (ICEEI)*, pp. 1–6, IEEE, Langkawi, Malaysia, November 2017.

[24] K. H. Rolland, "Desperately'seeking research on agile requirements in the context of large-scale agile projects," in *Proceedings of the Scientific Workshop XP2015*, pp. 1–6, New York, NY, USA, May 2015.

[25] D. H. Gobeli, H. F. Koenig, and I. Bechinger, "Managing conflict in software development teams: a multilevel analysis," *Journal of Product Innovation Management*, vol. 15, no. 5, pp. 423–435, 1998.

[26] A. Moniruzzaman and D. S. A. Hossain, "Comparative study on agile software development methodologies," 2013, http://arxiv.org/abs/13073356.

[27] R. Knaster and D. Leffingwell, *SAFe 4.0 Distilled: Applying the Scaled Agile Framework for Lean Software and Systems Engineering*, Addison-Wesley Professional, 2017.

[28] A. Putta, M. Paasivaara, and C. Lassenius, "Benefits and challenges of adopting the scaled agile framework (SAFe): preliminary results from a multivocal literature review," in *International Conference on Product-Focused Software Process Improvement*, pp. 334–351, Springer, Berlin, Germany, 2018.

[29] R. Brenner and S. Wunder, "Scaled agile framework: presentation and real world example," in *Proceedings of the 2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pp. 1-2, IEEE, Graz, Austria, April 2015.

[30] B. Ramesh, L. Cao, and R. Baskerville, "Agile requirements engineering practices and challenges: an empirical study," *Information Systems Journal*, vol. 20, no. 5, pp. 449–480, 2010.

[31] E.-M. Schön, D. Winter, M. J. Escalona, and J. Thomaschewski, "Key challenges in agile requirements engineering," in *Lecture Notes in Business Information Processing*, pp. 37–51, Springer, Berlin, Germany, 2017.

[32] K. Elghariani and N. Kama, "Review on Agile requirements engineering challenges," in *Proceedings of the 2016 3rd International Conference on Computer and Information Sciences (ICCOINS)*, pp. 507–512, IEEE, Kuala Lumpur, Malaysia, August 2016.

[33] T. Hafterson, "Incorporating agile methods into the development of large-scale systems," in *Proceedings of the UMM CSsci Senior Conference*, Moris, MN, USA, July 2011.

[34] K. Beck, M. Beedle, A. Van Bennekum et al., *Manifesto for Agile Software Development*, Agile Manifesto, 2001.

[35] J. Highsmith and A. Cockburn, "Agile software development: the business of innovation," *Computer*, vol. 34, no. 9, pp. 120–127, 2001.

[36] P. Ågerfalk and B. Fitzgerald, "Old petunias in new bowls?" *Communications of the ACM*, vol. 49, no. 10, p. 27, 2006.

[37] F. Dalpiaz and S. Brinkkemper, "Agile requirements engineering with user stories," in *Proceedings of the 2018 IEEE 26th International Requirements Engineering Conference (RE)*, pp. 506-507, IEEE, Banff, Alberta, Canada, August 2018.

[38] G. Lucassen, F. Dalpiaz, J. M. E. M. van der Werf, and S. Brinkkemper, "Improving agile requirements: the quality user story framework and tool," *Requirements Engineering*, vol. 21, no. 3, pp. 383–403, 2016.

[39] G. Lucassen, F. Dalpiaz, J. M. E. Van Der Werf, and S. Brinkkemper, "Forging high-quality user stories: towards a discipline for agile requirements," in *Proceedings of the 2015 IEEE 23rd International Requirements Engineering Conference (RE)*, pp. 126–135, IEEE, Ottawa, ON, Canada, August 2015.

[40] M. U. Malik, N. M. Chaudhry, and K. S. Malik, "Evaluation of efficient requirement engineering techniques in agile software development," *International Journal of Computer Applications*, vol. 83, no. 3, 2013.

[41] T. Jokela and P. Abrahamsson, "Usability assessment of an extreme programming project: close co-operation with the customer does not equal to good usability," in *Product Focused Software Process Improvement*, pp. 393–407, Springer, Berlin, Germany, 2004.

[42] E. Hossain, P. L. Bannerman, and D. R. Jeffery, "Scrum practices in global software development: a research framework," in *Product-Focused Software Process Improvement*, pp. 88–102, Springer, Berlin, Germany, 2011.

[43] M. Kumar, M. Shukla, and S. Agarwal, "A hybrid approach of requirement engineering in agile software development," in *Proceedings of the 2013 International Conference on Machine Intelligence and Research Advancement*, pp. 515–519, IEEE, Katra, JK, India, December 2013.

[44] T. Dingsøyr, T. E. Fægri, and J. Itkonen, "What is large in large-scale? A taxonomy of scale for agile software development," in *Proceedings of the International Conference on Product-Focused Software Process Improvement*, pp. 273–276, Springer, Rovaniemi, Finland, December 2014.

[45] K. Conboy and N. Carroll, "Implementing large-scale agile frameworks: challenges and recommendations," *IEEE Software*, vol. 36, no. 2, pp. 44–50, 2019.

[46] M. Laanti, O. Salo, and P. Abrahamsson, "Agile methods rapidly replacing traditional methods at Nokia: a survey of opinions on agile transformation," *Information and Software Technology*, vol. 53, no. 3, pp. 276–290, 2011.

[47] G. Papadopoulos, "Moving from traditional to agile software development methodologies also on large, distributed projects," *Procedia - Social and Behavioral Sciences*, vol. 175, no. 2, pp. 455–463, 2015.

[48] B. 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.

[49] J. S. Persson, L. Mathiassen, and I. Aaen, "Agile distributed software development: enacting control through media and context," *Information Systems Journal*, vol. 22, no. 6, pp. 411–433, 2012.

[50] P. Lous, P. Tell, C. B. Michelsen, Y. Dittrich, M. Kuhrmann, and A. Ebdrup, "Virtual by design: how a work environment can support agile distributed software development," in *Proceedings of the 2018 IEEE/ACM 13th International Conference on Global Software Engineering (ICGSE)*, pp. 97–106, IEEE, Gothenburg, Sweden, May 2018.

[51] R. Hoda and L. K. Murugesan, "Multi-level agile project management challenges: a self-organizing team perspective," *Journal of Systems and Software*, vol. 117, pp. 245–257, 2016.

[52] N. B. Moe and T. Dingsøyr, "Emerging research themes and updated research agenda for large-scale agile development: a summary of the 5th international workshop at XP2017," in *Proceedings of the XP2017 Scientific Workshops*, pp. 1–4, Cologne, Germany, May 2017.

[53] C. Fuchs and T. Hess, "Becoming agile in the digital transformation: the process of a large-scale agile transformation," in *Proceedings of the 39th International Conference on Information Systems (ICIS 2018)*, San Francisco, USA, December 2018.

[54] M. Jørgensen, "Do agile methods work for large software projects?" in *Proceedings of the International Conference on Agile Software Development*, pp. 179–190, Springer, Montréal, QC, Canada, May 2018.

[55] M. F. Abrar, M. S. Khan, S. Ali et al., "Motivators for large-scale agile adoption from management perspective: a systematic literature review," *IEEE Access*, vol. 7, pp. 22660–22674, 2019.

[56] A. Khalid, S. A. Butt, T. Jamal, and S. Gochhait, "Agile scrum issues at large-scale distributed projects," *International Journal of Software Innovation*, vol. 8, no. 2, pp. 85–94, 2020.

[57] M. Laanti, "Agile transformation model for large software development organizations," in *Proceedings of the XP2017 Scientific Workshops*, pp. 1–5, Cologne, Germany, May 2017.

[58] C. Fry and S. Greene, "Large scale agile transformation in an on-demand world," in *Proceedings of the Agile 2007 (AGILE 2007)*, pp. 136–142, IEEE, Washington, DC, USA, August 2007.

[59] S. Bick, A. Scheerer, and K. Spohrer, "Inter-team coordination in large agile software development settings: five ways of practicing agile at scale," in *Proceedings of the Scientific Workshop XP2016*, pp. 1–5, Scotland, UK, November 2016.

[60] O. Turetken, I. Stojanov, and J. J. Trienekens, "Assessing the adoption level of scaled agile development: a maturity model for Scaled Agile Framework," *Journal of Software: Evolution and Process*, vol. 29, no. 6, Article ID e1796, 2017.

[61] B. Kovitz, "Hidden skills that support phased and agile requirements engineering," *Requirements Engineering*, vol. 8, no. 2, pp. 135–141, 2003.

[62] J. A. Highsmith and J. Highsmith, *Agile Software Development Ecosystems*, Addison-Wesley Professional, Boston, MA, USA, 2002.

[63] P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta, "Agile software development methods: Review and analysis," 2017, http://arxiv.org/abs/170908439.

[64] T. Dingsøyr and N. B. Moe, "Towards principles of large-scale agile development," in *Proceedings of the International Conference on Agile Software Development*, pp. 1–8, Springer, Montreal, QC, Canada, May 2014.

[65] M. Laanti, "Characteristics and principles of scaled agile," in *Lecture Notes in Business Information Processing*, pp. 9–20, Springer, Berlin, Germany, 2014.

[66] Y. Zhu, *Requirements Engineering in an Agile Environment*, Uppsala University, Uppsala, Sweden, 2014.

[67] W. M. Farid and F. J. Mitropoulos, "NORMATIC: a visual tool for modeling non-functional requirements in agile processes," in *Proceedings of the IEEE Southeast*, pp. 1–8, IEEE, Orlando, FL, USA, March 2012.

[68] L. Cao and B. Ramesh, "Agile requirements engineering practices: an empirical study," *IEEE Software*, vol. 25, no. 1, pp. 60–67, 2008.

[69] N. N. B. Abdullah, S. Honiden, H. Sharp, B. Nuseibeh, and D. Notkin, "Communication patterns of agile requirements engineering," in *Proceedings of the 1st Workshop on Agile Requirements Engineering*, pp. 1–4, Lancaster, UK, June 2011.

[70] P. Rodríguez, A. Yagüe, P. P. Alarcón, and J. Garbajosa, "Some findings concerning requirements in agile methodologies," in *Lecture Notes in Business Information Processing*, pp. 171–184, Springer, Berlin, Germany, 2009.

[71] M. Amir, K. Khan, A. Khan, and M. Khan, "An appraisal of agile software development process," *International Journal of Advanced Science & Technology*, vol. 58, no. 56, p. 20, 2013.

[72] F. Paetsch, A. Eberlein, and F. Maurer, "Requirements engineering and agile software development," in *Proceedings of the Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 308–313, IEEE, Washington, DC, USA, June 2003.

[73] N. Ganesh and S. Thangasamy, "Issues identified in the software process due to barriers found during eliciting requirements on agile software projects: insights from India," *International Journal of Computer Applications*, vol. 16, no. 5, pp. 7–12, 2011.

[74] J. Cho, "Issues and Challenges of agile software development with SCRUM," *Issues in Information Systems*, vol. 9, no. 2, pp. 188–195, 2008.

[75] W. Helmy, A. Kamel, and O. Hegazy, "Requirements engineering methodology in agile environment," *International Journal of Computer Science Issues (IJCSI)*, vol. 9, no. 5, p. 293, 2012.

[76] M. Daneva, E. Van Der Veen, C. Amrit et al., "Agile requirements prioritization in large-scale outsourced system projects: an empirical study," *Journal of Systems and Software*, vol. 86, no. 5, pp. 1333–1353, 2013.

[77] M. S. Rahim, A. E. Chowdhury, and S. Das, "Rize: a proposed requirements prioritization technique for agile development," in *Proceedings of the 2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*, pp. 634–637, IEEE, Bangladesh, Asia, December 2017.

[78] M. Shameem, R. R. Kumar, C. Kumar, B. Chandra, and A. A. Khan, "Prioritizing challenges of agile process in distributed software development environment using analytic hierarchy process," *Journal of Software: Evolution and Process*, vol. 30, no. 11, Article ID e1979, 2018.

[79] D. Mishra and A. Mishra, "Complex software project development: agile methods adoption," *Journal of Software Maintenance and Evolution: Research and Practice*, vol. 23, no. 8, pp. 549–564, 2011.

[80] K. Dikert, M. Paasivaara, and C. Lassenius, "Challenges and success factors for large-scale agile transformations: a systematic literature review," *Journal of Systems and Software*, vol. 119, pp. 87–108, 2016.

[81] E. Bjarnason, K. Wnuk, and B. Regnell, "A case study on benefits and side-effects of agile practices in large-scale requirements engineering," in *Proceedings of the 1st Workshop on Agile Requirements Engineering*, pp. 1–5, Lancaster, UK, June 2011.

[82] K. Rolland, T. Dingsoyr, B. Fitzgerald, and K. J. Stol, "Problematizing agile in the large: alternative assumptions

for large-scale agile development," in *Proceedings of the 39th International Conference on Information Systems*, Association for Information Systems (AIS), Nysa, Poland, September 2016.

[83] P. Xu, "Coordination in large agile projects," *Review of Business Information Systems (RBIS)*, vol. 13, no. 4, 2009.

[84] R. Kasauli, G. Liebel, E. Knauss, S. Gopakumar, and B. Kanagwa, "Requirements engineering challenges in large-scale agile system development," in *Proceedings of the 2017 IEEE 25th International Requirements Engineering Conference (RE)*, pp. 352–361, IEEE, San Francisco, CA, USA, December 2017.

[85] M. Lindvall, D. Muthig, A. Dagnino et al., "Agile software development in large organizations," *Computer*, vol. 37, no. 12, pp. 26–34, 2004.

[86] G. Lee, W. DeLone, and J. A. Espinosa, "Ambidextrous coping strategies in globally distributed software development projects," *Communications of the ACM*, vol. 49, no. 10, pp. 35–40, 2006.

[87] N. A. Ernst, A. Borgida, I. J. Jureta, and J. Mylopoulos, "Agile requirements engineering via paraconsistent reasoning," *Information Systems*, vol. 43, pp. 100–116, 2014.

[88] H. Hufmann and F. Lehner, "Requirements engineering as a success factor in software projects," *IEEE software*, vol. 18, pp. 58–66, 2001.

[89] I. Morales-Ramirez and L. H. Alva-Martinez, "Requirements analysis skills: how to train practitioners?" in *Proceedings of the 2018 IEEE 8th International Workshop on Requirements Engineering Education and Training (REET)*, pp. 24–29, IEEE, Banff, Alberta, Canada, August 2018.

[90] D. M. Fernández, S. Wagner, M. Kalinowski et al., "Naming the pain in requirements engineering," *Empirical Software Engineering*, vol. 22, no. 5, pp. 2298–2338, 2017.

[91] T. Kamal, Q. Zhang, and M. A. Akbar, "Toward successful agile requirements change management process in global software development: a client–vendor analysis," *IET Software*, vol. 14, no. 3, pp. 265–274, 2019.

[92] T. Kamal, Q. Zhang, M. A. Akbar, M. Shafiq, A. Gumaei, and A. Alsanad, "Identification and prioritization of agile requirements change management success factors in the domain of global software development," *IEEE Access*, vol. 8, pp. 44714–44726, 2020.

[93] K. AbdElazim, R. Moawad, and E. Elfakharany, "A framework for requirements prioritization process in agile software development," *Journal of Physics*, vol. 1454, 2020.

[94] W. M. Farid and F. J. Mitropoulos, "Visualization and scheduling of non-functional requirements for agile processes," in *Proceedings of the 2013 IEEE Southeastcon*, pp. 1–8, IEEE, Jacksonville, FL, USA, April 2013.

[95] M. Younas, D. Jawawi, I. Ghani, and R. Kazmi, "Non-functional requirements elicitation guideline for agile methods," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 9, no. 3-4, pp. 137–142, 2017.

[96] M. Käpyaho and M. Kauppinen, "Agile requirements engineering with prototyping: a case study," in *Proceedings of the 2015 IEEE 23rd International Requirements Engineering Conference (RE)*, pp. 334–343, IEEE, Ottawa, ON, USA, August 2015.

[97] L. Corral and I. Fronza, "Design thinking and agile practices for software engineering: an opportunity for innovation," in *Proceedings of the 19th Annual SIG Conference on Information Technology Education*, pp. 26–31, Fort Lauderdale, FL, USA, October 2018.

[98] C. Snyder, *Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces*, Morgan Kaufmann, Burlington, MA, USA, 2003.

[99] R. McAdam, B. Mason, and J. McCrory, "Exploring the dichotomies within the tacit knowledge literature: towards a process of tacit knowing in organizations," *Journal of Knowledge Management*, 2007.

[100] V. Gervasi, R. Gacitua, M. Rouncefield et al., "Unpacking tacit knowledge for requirements engineering," in *Managing Requirements Knowledge*, pp. 23–47, Springer, Berlin, Germany, 2013.

[101] S. Ryan and R. V. O'Connor, "Acquiring and sharing tacit knowledge in software development teams: an empirical study," *Information and Software Technology*, vol. 55, no. 9, pp. 1614–1624, 2013.

[102] S. Ramzan and N. Ikram, "Requirement change management process models: activities, artifacts and roles," in *Proceedings of the 2006 IEEE International Multitopic Conference*, pp. 219–223, Islamabad, Pakistan, December 2006.

[103] A. R. Asghar, S. N. Bhatti, A. Tabassum, and S. Shah, "The impact of analytical assessment of requirements prioritization models: an empirical study," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 8, no. 2, pp. 303–313, 2017.

[104] S. A. Marjaie and V. Kulkarni, "Recognition of hidden factors in requirements prioritization using factor analysis," in *Proceedings of the 2010 International Conference on Computational Intelligence and Software Engineering*, pp. 1–5, IEEE, Wuhan, China, December 2010.

[105] K. Sureshchandra and J. Shrinivasavadhani, "Adopting agile in distributed development," in *Proceedings of the 2008 IEEE International Conference on Global Software Engineering*, pp. 217–221, IEEE, Bangalore, India, August 2008.

[106] S. Dorairaj, J. Noble, and P. Malik, "Knowledge management in distributed agile software development," in *Proceedings of the 2012 Agile Conference*, pp. 64–73, IEEE, Dallas, TX, USA, August 2012.

[107] D. Stankovic, V. Nikolic, M. Djordjevic, and D.-B. Cao, "A survey study of critical success factors in agile software projects in former Yugoslavia IT companies," *Journal of Systems and Software*, vol. 86, no. 6, pp. 1663–1678, 2013.

[108] I. Inayat, S. S. Salim, S. Marczak, M. Daneva, and S. Shamshirband, "A systematic literature review on agile requirements engineering practices and challenges," *Computers in Human Behavior*, vol. 51, pp. 915–929, 2015.

[109] M. Korkala, P. Abrahamsson, and P. Kyllonen, "A case study on the impact of customer communication on defects in agile software development," in *Proceedings of the AGILE 2006 (AGILE'06)*, p. 11, IEEE, Brazil, October 2006.

[110] J. M. Verner, M. A. Babar, N. Cerpa, T. Hall, and S. Beecham, "Factors that motivate software engineering teams: a four country empirical study," *Journal of Systems and Software*, vol. 92, pp. 115–127, 2014.

[111] S. Arai, K. Sakamoto, H. Washizaki, and Y. Fukazawa, "A gamified tool for motivating developers to remove warnings of bug pattern tools," in *Proceedings of the 2014 6th International Workshop on Empirical Software Engineering in Practice*, pp. 37–42, IEEE, Hong Kong, China, November 2014.

[112] Y. Wautelet, S. Heng, D. Hintea, M. Kolp, and S. Poelmans, "Bridging user story sets with the use case model," in *Lecture Notes in Computer Science*, pp. 127–138, Springer, Berlin, Germany, 2016.

[113] S. Ramzan and N. Ikram, “Requirement change management process models: activities, artifacts and roles,” in *Proceedings of the 2006 IEEE International Multitopic Conference*, pp. 219–223, IEEE, Islamabad, Pakistan, December 2006.

[114] E. Knauss, G. Liebel, J. Horkoff et al., “T-reqs: tool support for managing requirements in large-scale agile system development,” in *Proceedings of the 2018 IEEE 26th International Requirements Engineering Conference (RE)*, pp. 502-503, IEEE, Banff, Alberta, Canada, April 2018.

[115] J. E. Hannay, K. Brathen, and O. M. Mevassvik, “Agile requirements handling in a service-oriented taxonomy of capabilities,” *Requirements Engineering*, vol. 22, no. 2, pp. 289–314, 2017.

[116] D. Albuquerque, E. Guimaraes, M. Perkusich et al., *Defining Agile Requirements Change Management: A Mapping Study*, ACM, New York, NY, USA.

[117] H. M. Sarkan, T. P. S. Ahmad, and A. A. Bakar, “Using JIRA and Redmine in requirement development for agile methodology,” in *Proceedings of the 2011 Malaysian Conference in Software Engineering*, pp. 408–413, IEEE, Pahang, Malaysia, June 2011.

[118] R. Noor and M. Fahad Khan, “Defect management in agile software development,” *International Journal of Modern Education and Computer Science*, vol. 6, no. 3, pp. 55–60, 2014.

[119] D. Batra, W. Xia, D. E. VanderMeer, and K. Dutta, “Balancing agile and structured development approaches to successfully manage large distributed software projects: a case study from the cruise line industry,” *CAIS*, vol. 27, p. 21, 2010.

[120] E. Rubin and H. Rubin, “Supporting agile software development through active documentation,” *Requirements Engineering*, vol. 16, no. 2, pp. 117–132, 2011.

[121] L. b. Othmane, P. Angin, H. Weffers, and B. Bhargava, “Extending the agile development process to develop acceptably secure software,” *IEEE Transactions on Dependable and Secure Computing*, vol. 11, no. 6, pp. 497–509, 2014.

[122] D. Turk, R. France, and B. Rumpe, “Limitations of agile software processes,” 2014, http://arxiv.org/abs/14096600.