© IEEE. PREPRINT/EARLY VIEW. This is the author's version of the work.
It is posted here by permission of IEEE for your personal use. Not for redistribution.
The definitive version was published in the conference proceedings.

Is Scrum Fit for Global Software Engineering?

Pernille Lous
IT University of Copenhagen
Copenhagen, Denmark
pelo@itu.dk

Marco Kuhrmann
Clausthal University of Technology
Clausthal-Zellerfeld, Germany
marco.kuhrmann@tu-clausthal.de

Paolo Tell
IT University of Copenhagen
Copenhagen, Denmark
pate@itu.dk

***Abstract*—Distributed software engineering and agility are strongly pushing on today's software industry. Due to inherent incompatibilities, for years, studying Scrum and its application in distributed setups has been subject to theoretical and applied research, and an increasing body of knowledge reports insights into this combination. Through a systematic literature review, this paper contributes a collection of experiences on the application of Scrum to global software engineering (GSE). In total, we identified 40 challenges in 19 categories practitioners face when using Scrum in GSE. Among the challenges, scaling Scrum to GSE and adopting practices accordingly are the most frequently named. Our findings also show that most solution proposals aim at modifying elements of the Scrum core processes. We thus conclude that, even though Scrum allows for extensive modification, Scrum itself represents a barrier for global software engineering, and development teams have to customize Scrum properly to benefit from agile software development in GSE.**

***Index Terms*—global software engineering; agile software development; systematic literature review; systematic mapping study**

I. Introduction

Two strong trends shape modern software engineering: first, according to Herbsleb [1], companies are pushed strongly to embark in distributed software engineering endeavors to, among other reasons, attempt to lower production cost and increase the pool of available talented individuals. Second, agile and lean practices are increasingly applied in (distributed) projects to support a more dynamic handling of the products. Among the multitude of agile and lean practices, Scrum has taken a leading role [2]–[4]. Yet, these two trends seemingly create a contradiction, since distributed software engineering requires a number of rules and formalisms to coordinate the different teams spread across the globe, but agile software development is, on the other hand, strongly driven by immediate and direct communication and collaboration of people—quite often in small and co-located teams.

For years, studying Scrum (and agile software development in GSE in general [5]) and its application to distributed setups has been subject to theoretical and applied research, and an increasing body of knowledge reports insights into the delicate combination of the two trends mentioned before. However, the complexity of this research is further increased by the fact that practitioners have developed different interpretations and embodiments of Scrum [6], that is, similar to software processes in general [7], there is no "Silver Bullet" for implementing Scrum—neither in co-located nor in distributed setups.

A. Problem statement and objectives

A mostly problem- and situation-driven adaptation of Scrum makes it hard to identify a structure, which makes it difficult to obtain a comprehensive picture of the challenges faced and respective *effective* solutions. In 2011, Šmite et al. [8] already named this problem and put forward a call to (i) structure the literature on agile software development in global software engineering and (ii) define the current practices in terms of characteristics of the various agile methods used in GSE.

This paper aims to provide such a structure with a special emphasis on Scrum. The objective of the present research is to analyze and structure the available research as well as to provide a detailed breakdown of the successful practices and techniques applied for tackling the challenges arising from combining Scrum and GSE.

B. Context & Background

Scrum and GSE set the scene for the present research. For this, a previously conducted literature review by Ebert et al. [9] on the advances of the GSE community[1] serves as starting point (see Sect. II for further details). The second analysis tool for this research is given by the process model described in the *Scrum Guide* [10], which comprises ceremonies, roles, and artifacts. In particular, *ceremonies* relate to sprint planning, daily sprint, sprint review, and sprint retrospective. Product backlog and sprint backlog are the *artifacts* involved; while, the *roles* include the development team, the product owner, and the Scrum master (cf. Sect. II-D). The Scrum guide and the input data serve the stepwise analysis of the impact of challenges and solutions reported.

C. Contribution

This paper contributes a collection of experiences reported in the literature on the way practices and techniques were applied to the different elements of the Scrum process in GSE projects to mitigate challenges coming along with distributed software engineering. Utilizing the systematic literature review (SLR; [11]) instrument in combination with a research maturity index [9], this paper provides evidence from the industry on the interplay between GSE and Scrum.

[1]In the context of this paper, we refer to the community mostly represented by the IEEE International Conference on Global Software Engineering (ICGSE); see http://www.icgse.org

© IEEE. PREPRINT/EARLY VIEW. This is the author's version of the work.
It is posted here by permission of IEEE for your personal use. Not for redistribution.
The definitive version was published in the conference proceedings.

D. Outline

The remainder of the paper is structured as follows: Section II presents the research design, and Sect. III presents our findings, which are discussed in Sect. IV. Section V presents related work and positions this paper in the currently available body of knowledge, before we conclude the paper in Sect. VI.

II. Research Design

The 2007-guideline by Kitchenham and Charters [11] for conducting systematic reviews was used as base method. The respectively conducted steps (definition of research questions, search strategies, selection criteria, quality assessment procedures, and procedures for data extraction and synthesis) are explained in the subsequent subsections.

A. Research Questions

While working on the study's overall goals, this paper aims to answer the following research questions:

RQ_1	What challenging factors restrict the use of Scrum in globally distributed projects?
RQ_2	What strategies, suggestions, and practices are used to implement Scrum in distributed environments?

B. Search Strategy & Study Selection

We searched for relevant articles by applying *backward snowballing* [12] on the references of an initial set of articles. This technique was chosen as it yields result sets as adequate as those obtainable through other techniques [13] and provides an innovation worth pursuing compared to existing studies (Sect. V). The initial set of articles was taken from an updated dataset of a previously conducted study [9] that provides a categorized mapping of all ICGSE articles. Finally, the set of selection criteria and quality assessment procedures (Sect. II-C) designed for the entire study has been applied to filter the dataset for articles in scope of the current research. The overall procedure is illustrated in Figure 1.

Besides assessing criteria related to language, peer-reviewed content, and availability of the article, the initial screening process was designed to exclude papers that are (i) neither related to global software engineering (ii) nor to agile software development. This assessment was carried out by evaluating the titles and the abstracts, and—if necessary—reviewing the papers entirely. Finally, (iii) papers of which the results are not based on industrial involvement were also excluded.

C. Study Quality Assessment

To carry out the quality assessment and to identify those papers without sufficient industry involvement, we applied the maturity index as done in (Table I; [9]). The two parameters of the maturity index, which are similar to the rigor-relevance model as proposed by Ivarsson and Gorschek [14], have been used to assess the level of empirical investigation reported (*Strength of Evidence*, SoE) and the kind of industrial involvement (*Industry collaboration Pattern*, IcP). Table I provides details regarding the two scales to used assess *SoE* and *IcP*.

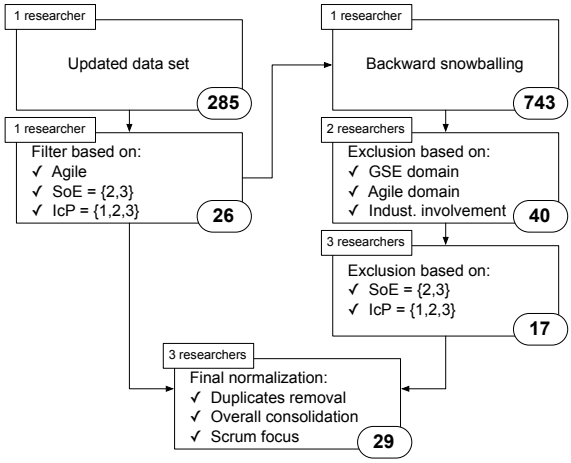

Fig. 1. Overview of the search strategy (starting point is an updated data set from [9]). Additionally, the figure shows the number of researchers involved in each stage, the selection criteria applied to the data set, and the number of papers resulting from each stage.

TABLE I
Maturity index to assess strength of evidence and industry collaboration pattern.

Score	Description
Strength of Evidence (scores: NA/NR=0,...,3)	
NA/NR	Not relevant (e.g., position paper) or based on experience only, e.g., "success stories" without empirical evidence
Weak	University lab with students only/experiment or simulation
Average	Industry is involved, but only one case, e.g., a demonstrator; or secondary studies, due to inherent publication bias
Strong	Multi-case, longitudinal or replication studies (can be lab-only or with industry involvement)
Industry Collaboration Pattern (scores: NA=0,...,3)	
NA	No involvement, e.g., university lab only or SLR/SMS
Interview	Practitioner interviews only
SingleCase	Single case (with/without complementing interviews)
Close	E.g., multiple studies in one company, or multiple companies in one study, or different research methods applied (mixed-method)

For an article to be included as a primary study (PS), the study had to be assessed having an *SoE* value of 2 or 3, and an *IcP* value between 1 and 3. That is, explicit industry collaboration was a key requirement for inclusion, and, among these articles, only those providing average to strong empirical evidence were considered.

D. Data Extraction & Synthesis

To extract data from the selected studies, we carried out different steps. We started with assessing the maturity index (i.e., SoE and IcP) used for in-/exclusion of the studies, and fetching demographic information, such as year of publication, venue, and so forth. Furthermore, context information and case-related details were extracted from the primary studies, which comprise for instance of projects size (number of people involved), number of sites involved, number of teams, and

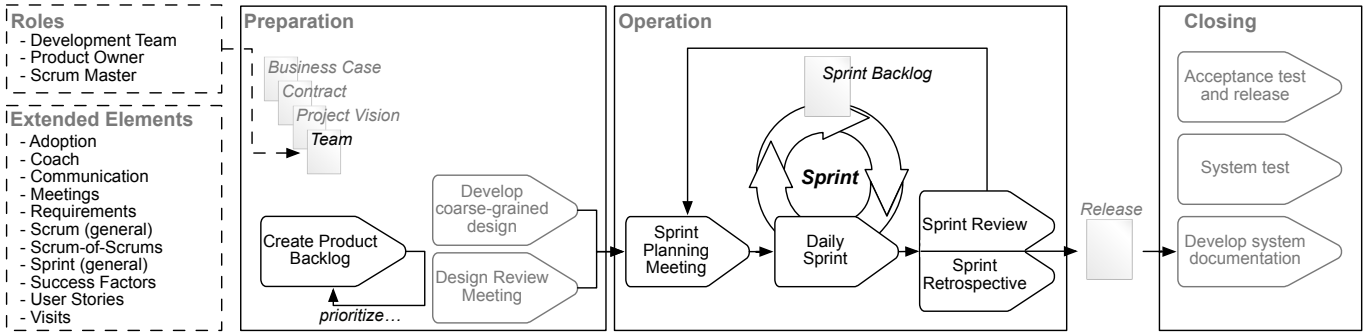

Fig. 2. Overview of the Scrum process used for the analysis (incl. all analysis elements from the core process and the extended practices).

target domains/industry sectors for which the studied products were developed.

For the in-depth investigation, the primary studies have been thoroughly analyzed to extract codes representing text snippets containing challenges, solutions, practices, and suggestions. To craft the codes, our approach borrowed from grounded theory [15], but used the process model described in the Scrum guide [10] as a reference and starting point for high-level concepts and categories—we refer to this set of categories to as the "core" process. In addition to these Scrum-based categories, while analyzing the articles, further categories were extracted to which we refer to as "extended practices". Figure 2 provides an overview of both the core process and the extended perspective that includes all other categories used for structuring and presenting our findings—in total, 19 categories. Both category sets provide us with a framework to present and discuss the qualitative data.

III. Results

This section presents the results of our study starting by presenting the result set obtained from implementing our search in Sect. III-A. Section III-B and Sect. III-C present the results addressing the research questions. An explicit discussion of the results is presented in Sect IV.

A. Result Set Overview & Demographics

Filtering the 285 papers provided by the updated dataset from [9], we isolated an initial data set of 26 papers that, after applying backward snowballing, resulted in 743 papers. Applying the selection procedures as described in Sect. II, 40 papers remained for which the maturity index was computed. After removing duplicates, consolidating the two sets, and enforcing the Scrum focus, eventually, **29** papers were selected for the in-depth analysis of which 14 report multi-case studies. In the paper at hand, we refer to the selected primary studies using a prefix "PS" and provide the references in a separate reference section.

1) Overview of the Primary Studies: Table II provides an overview of the selected primary studies in chronological order, and Figure 3 shows the primary studies' distribution over the years. The figure shows a growing number of publications after 2011. However, the publication frequency

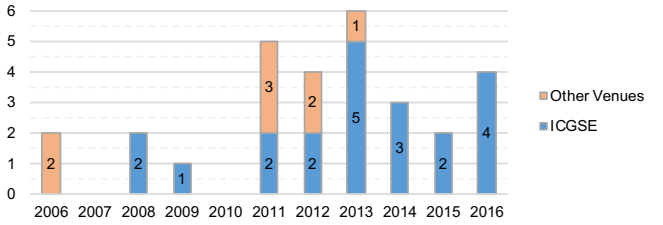

Fig. 3. Publication frequency of the primary studies (the chart distinguishes studies published at ICGSE from other venues, cf. Table I).

Fig. 4. Summary the result set focus (based on the papers' scope, cf. Table II).

of empirical studies on agile software development in GSE is rather irregular and, seemingly, authors mainly target the ICGSE conference series. Figure 4 visualizes the focus of the studies based on their respective scope (Table II): processes and practices (17 out of 29; 4 with an explicit focus on roles), general discussion on agile in GSE, knowledge management, and communication and coaching count each for 4 out of 29. Furthermore, among all focal points, *scaling for large projects* (13 out of 29) was identified as one of the most frequently mentioned one (see Sect. IV for a more detailed elaboration).

2) Characteristics of Distributed Scrum Projects: To analyze the primary studies, we use the characteristics presented in Table III, namely: application domain, project size (number of people involved), number of teams, number of sites, and temporal overlap. Regarding the application domain, our data shows a good distribution that supports the claim that GSE has become mainstream and, thus, companies have to adopt their processes to GSE.

The project characterization reveals GSE addressing companies and projects of different size and complexity. In particular, GSE has an impact on the project organization and management. Although many projects report a large temporal overlap (i.e., >2 hrs), our data also shows many projects

© IEEE. PREPRINT/EARLY VIEW. This is the author's version of the work.
It is posted here by permission of IEEE for your personal use. Not for redistribution.
The definitive version was published in the conference proceedings.

TABLE II
OVERVIEW OF PRIMARY STUDIES IN CHRONOLOGICAL ORDER.

PS	Year	Venue	SoE	IcP	Main focus
[PS27]	2006	cACM	3	3	agile in GSE, suggestions
[PS10]	2006	ISM	3	3	agile in GSE, benefits
[PS19]	2008	ICGSE	2	2	large project, process description
[PS4]	2008	ICGSE	3	3	global project, practice suggestions
[PS20]	2009	ICGSE	3	3	large project, process description
[PS17]	2011	ICGSE	3	3	coaching
[PS22]	2011	ESEM	2	2	scaling, practices
[PS8]	2011	ISEC	2	3	cultural distance
[PS15]	2011	ICGSE	2	2	communication
[PS26]	2011	ICSSP	2	2	coaching
[PS16]	2012	ICGSE	2	2	adoption, practice analysis
[PS21]	2012	ICGSE	3	3	scaling, roles (PO)
[PS24]	2012	ESEM	3	3	scaling, process (SoS)
[PS7]	2012	Agile	3	3	knowledge mgmt., practices
[PS12]	2013	AISeL	2	2	distributed agile, practices
[PS11]	2013	ICGSE	2	2	negotiating common ground
[PS25]	2013	ICGSE	3	2	scaling
[PS29]	2013	ICGSE	2	2	practice analysis
[PS1]	2013	ICGSE	3	3	scaling, roles (PO team)
[PS6]	2013	ICGSE	3	3	scaling, roles (senior mgmt support)
[PS9]	2014	ICGSE	3	3	agile vs. structured, practice desc.
[PS2]	2014	ICGSE	3	3	scaling, roles (SM)
[PS18]	2014	ICGSE	2	2	scaling, process
[PS13]	2015	ICGSE	2	2	scaling, process
[PS28]	2015	ICGSE	3	3	knowledge mgmt., analysis
[PS14]	2016	ICGSE	2	2	knowledge sharing
[PS5]	2016	ICGSE	2	2	communication
[PS23]	2016	ICGSE	2	2	scaling, practices
[PS3]	2016	ICGSE	3	3	knowledge sharing (architectural)

having a temporal overlap of $\leq$2 hrs. Hence, coordination and communication are two considerable challenges. Even though a majority of the projects report a two-site collaboration, the result set also contains projects implemented at four, five, and more sites. For instance, three primary studies report cases in which the projects are run by 20 teams at up to four sites.

That is, the reported data provides a good snapshot covering companies of all sizes and working in multiple industry sectors. All cases face the challenges coming along with GSE, i.e., communication and team collaboration, knowledge sharing, and applying and scaling agile methods to GSE (cf. Table II). However, the studies analyzed also include a number of cases that provide insufficient information for a thorough classification.

B. RQ1: Challenging Factors for Using Scrum

Given that the majority of the primary studies have been published at ICGSE, the main impacting factors identified are very often linked to the dimensions of distribution used within this community, i.e.: geographical, temporal, linguistic, and cultural [16]; or geographical, temporal, and socio-cultural [17]. Additionally, the scope of the primary studies analyzed concerns challenges related to scaling Scrum (explicated in 10 primary studies), which severely impacts the process as such and its practices. Most of the other aspects considered in the primary studies are studied in the context of the different scaling approaches. The complete collection of challenges

TABLE III
CHARACTERIZATION OF THE PROJECTS FROM THE PRIMARY STUDIES.

	Subcategory	#
Project size	0-15	41
	16-25	15
	$\geq$ 25	7
	Unclear	25
Number of teams	2	30
	3	8
	4-7	3
	>15	2
	Unclear	45
Number of sites	2	48
	3	16
	4	4
	$\geq$ 5	3
	Unclear	34
Temporal overlap	Yes (>2 hrs)	31
	Yes ($\leq$ 2 hrs)	28
	No	5
	Multi-sites	3
	Unclear	20

Application domain	#
Finance	7
Internet/Web	7
Telecom/Mobile	7
Service	4
Software Service	3
Retail	3
Energy and Oil	3
Online Trading	2
Industrial	2
Hardware	2
E-commerce	2
CRM	2
Transport/Logistics	2
Supply Chain	1
Recruitment	1
Publishing	1
Power Distribution	1
IT-Service	1
Unclear	38

Note: [PS6] has been excluded, for it reported data from 38 unspecified cases.

identified in the result set is provided in Table IV and Table V, these are further used to map the respective solution proposals. In the following, for space limitations, we only discuss selected challenges in more detail.

1) Scaling Scrum: Scaling Scrum to a (globally) distributed setup is regarded a central impediment to its adoption, since Scrum's core elements are designed to leverage physical co-location and extensive face-to-face interactions. In particular, it can be observed how the roles (e.g., understanding and implementation [PS17]), ceremonies (e.g., attending to meetings [PS17] or scaling meetings [PS24]), and artifacts (e.g., access to [PS4] and management of [PS20] distributed artifacts) are deeply affected by the absence of co-location, making the design of additional elements a necessity.

2) Knowledge Management: Knowledge management is a key topic, e.g., sharing architectural knowledge is regarded as a major concern [PS3], which is often related to the issues of managing distributed artifacts, notably in terms of introducing technical debt [PS18]. In addition, the general knowledge management within a project is considered challenging. For example, Moe et al. [PS14] focus on how to develop and maintain team knowledge in agile virtual teams. They conclude that even though processes and practices can be devised (i.e., annual localized gathering of the entire team or messenger channels dedicated to support knowledge sharing), maintaining team knowledge is expensive and it is crucial to consider which categories of knowledge should be favored. Razzak and Šmite [PS28] stress the importance of not underestimating the importance of knowledge management in a distributed setup, and Modi et al. [PS11] focus on the difficulties to find and negotiate common ground in distributed projects.

3) Communication and Coaching: Scrum highly relies on communication and people interaction. Communication in a

© IEEE. PREPRINT/EARLY VIEW. This is the author's version of the work.
It is posted here by permission of IEEE for your personal use. Not for redistribution.
The definitive version was published in the conference proceedings.

distributed setup is explicitly stressed by two studies [PS5], [PS15]. For instance, Niinimaki [PS15] quantitatively evaluates the impact of distribution on communication patterns and presents practices that have been adopted to circumvent the lack of face-to-face communication. Coaching is regarded a complication when it comes to a distributed setup. The major challenges named are: ensuring executive support [PS26], proper involvement of the management [PS17], [PS26], and the achievement of a continuous software process improvement [PS26].

C. RQ 2: Strategies, Suggestions, and Practices to Implement Scrum in Distributed Settings

In this section, we collect and structure challenges (cf. Sect. III-B) and respective strategies and practices to implement Scrum in distributed environments. For the result presentation, we use two perspectives, which will be later discussed in more detail. The first perspective in Table IV is given by the Scrum core process, and the second perspective in Table V uses a set of extended practices crafted by analyzing the selected primary studies as described in Sect. II-D.

1) The Scrum Core Process: The first perspective provides a detailed view of challenges, strategies, and practices used to implement Scrum in distributed environments. For the presentation and discussion of our findings regarding eight categories of the core process (i.e., those elements widely accepted as the basic assets of Scrum; cf. Figure 2), we distinguish between *ceremonies*, *artifacts*, and *roles* [10] for which we identified 21 challenges in total.

The main challenges identified for the Scrum ceremonies are organizing and holding meetings [PS17] and fixing process issues [PS23]. A particular issue identified is the applicability of Scrum to large setups, i.e., with 20+ teams [PS23]. The papers studied suggest a variety of solutions to overcome these issues, such as: improved organization of meetings (e.g., consecutive time slots, splitting meetings based on locations [PS11], [PS12]), refining roles (e.g., define proxies [PS21] for specific areas of responsibility [PS21], [PS23], [PS22] for product owner (groups), using multiple Scrum masters [PS21], [PS20]), or even implement higher levels of integration of the management as such, e.g., using Scrum-of-Scrums [PS19], [PS20], [PS23], [PS24] or complex agile frameworks like Large-scale Scrum (LeSS), Scaled Agile Framework (SAFe), or Disciplined Agile Delivery (DAD) [PS23].

A similar situation can be observed for the artifacts, in particular the product and sprint backlogs for which shared management [PS20] and technical debt as part of the evolution [PS18] are considered major challenges. As solution approaches, again, refinements of the role model are suggested (e.g., specialized roles [PS18] or proxy roles [PS21]). Yet, while [PS19] proposes individual, role- or team-specific backlogs to overcome artifact-related bottlenecks, [PS18] suggest using a shared backlog only—going as far as inviting customers to report bugs directly through the product backlog [PS19]. To address awareness concerns, [PS4], [PS12] suggest the use of a global Scrum board, which of course from being a physical tool becomes a virtual one. A need that is being more and more supported by commercial bug and issue tracking systems like Jira and Microsoft Team Foundation Server.

Although most challenges regarding the roles concern particular work practices, still, scaling issues can be found in the reported literature. In particular, (shared) knowledge management [PS11], [PS3] and defining/scaling the role of the product owner in large setups [PS1] are considered challenging. The solution proposals to address these challenges are in line with the aforementioned ones, e.g., refined/specialized roles or proxy roles [PS21], [PS23], [PS22], fostering collaborative development practices (e.g., "virtual" pair programming [PS29], [PS7] or code buddies [PS29]), and implementing trust- and team-building measures like improved communication, visits, annual gatherings, and team member rotation [PS19], [PS20], [PS27], [PS17]—even if only for a short kickoff period, given the financial and resource commitment required.

2) The Extended Scrum Practices: Complementing the challenges related to the Scrum core process, a number of further elements were considered challenging, which resulted in 11 extra categories for which we identified 19 challenges Table V. Remarkable challenges address the adoption of Scrum, communication and meetings, Scrum at the large scale, and team-related issues. For instance, to address the challenge of creating a shared understanding [PS25], appropriate measures suggested are: visiting team members, team member relocation [PS19], and creation of competence exchange programs [PS25]. These measures are are in line with a number of suggestions about the organization of such visits, e.g., [PS11], [PS25], [PS18], [PS7], [PS12]. However, Moe et al. [PS14] argue that cost and impact of visits should be considered as well. The problem field of general meetings shows similarities with the core process ceremonies. In particular, Paasivaara et al. suggest (formal) rules to be defined for holding meeting [PS19] and consider the technical meeting equipment critical [PS18]—especially concerning the technical challenges as for instance stated by Modi et al. [PS11]. Finally, scaling the entire process is considered challenging and, similar to the core process, different solutions are proposed, such as global Scrum boards [PS4], [PS12] or utilizing more compressive methods like a Scrum-of-Scrums [PS19], [PS20], [PS23], [PS24].

IV. Discussion

Our review findings in Table IV and Table V support an observation, which is already grounded in the collection of primary studies in Table II. In particular, papers published between 2006 and 2010 (Table II) mainly aim at proposing and discussing the general use of agile methods in GSE. Yet, starting with 2011, the number of papers discussing practical application and naming issues increases. Among the the most frequently mentioned focal points is *scaling* (Figure 4 and Sect. III-B). Seemingly, adopting agile methods to a GSE context requires scaling those approaches, e.g., regarding the general process and its management, roles, communication,

© IEEE. PREPRINT/EARLY VIEW. This is the author's version of the work.
It is posted here by permission of IEEE for your personal use. Not for redistribution.
The definitive version was published in the conference proceedings.

TABLE IV
SOLUTIONS, PRACTICES, AND SUGGESTIONS TO OVERCOME CHALLENGES IN USING SCRUM (CORE PROCESS) IN DISTRIBUTED ARRANGEMENTS.

Process Element	Challenges	Solutions, Practices, and Suggestions
Ceremonies		
Sprint Planning		[PS19], [PS20]: 3-staged meeting with 2 local and 1 distributed meeting; [PS23]: a joint 1-hour sprint planning meeting; [PS20]: visits are efficient for meetings, especially in the beginning (participants: Scrum master, architects/designers); [PS17]: if traveling between sites is an option, do it;
Daily Sprint	[PS4]: lack of attendance; [PS4]: too technical discussions; [PS17]: meetings at only on one site; [PS24]: scaling;	[PS19], [PS20]: for multiple teams, use fixed consecutive time slots, i.e., one team after the other in one room; [PS20]: at the beginning, write answers to 3 questions and every team member reads the others' answers; [PS21]: use a product owner proxy; [PS11], [PS12]: split meetings (per location, senior team members); [PS23]: do Scrum-of-Scrums meetings daily after the teams' sessions (1 person per team should participate);
Sprint Review	[PS21], [PS23]: scaling, e.g., more than 20 teams;	[PS19], [PS20]: team prepares agenda; [PS20]: combine review and planning; [PS21]: for many teams, arrange a common sprint review for all teams (representatives only briefly explain their achievements); [PS21]: demo results to product owner proxies; [PS29]: use a "war room" to think about the user stories from the end-users' perspective; [PS23]: project manager and product owner visit each team and demo achievements; [PS19], [PS20]: foster team building, e.g., applause if good quality is presented;
Sprint Retrospective	[PS23]: fixing the larger issues;	[PS20]: Scrum masters (on-/off-site) discuss improvements without the teams; [PS29]: make the Scrum master strong and present to build trust; [PS23]: use an "open space" in which anyone can suggest discussion topics; [PS23]: identify impediments, prioritize them, find root causes, and develop solutions step-by-step;
Artifacts		
Product Backlog/ Sprint Backlog	[PS4]: access to meeting minutes to clarify requirements; [PS20]: management of the items by different people (notably in projects without clear roles); [PS18]: technical debt;	[PS19]: individual, team-specific backlogs (development, maintenance, etc.) vs. [PS18]: establish a common backlog; [PS19]: customers have access to the backlog (e.g., to report bugs); [PS21]: product owner proxies take care of grooming till epics before involving the team for the user stories; [PS18]: introduce supporting roles (e.g., for subsystems or architects);
Roles		
Development Team	[PS4]: resistance to be a generalist rather than a specialist; [PS17]: using junior team members only; [PS11]: distributed practice implementation (e.g., finding common ground on practices to use); [PS11], [PS3]: sharing knowledge (tacit, architecture); [PS8] bridge cultural gaps; [PS27]: lacking team cohesion;	[PS4]: create multi-functional teams (e.g., developers and testers together); [PS20], [PS17]: do not split teams across sites (team cohesion); [PS17] empower team members of all sites and provide training; [PS17]: transfer with no hichup (off-site team had a team leader and a product area manager from the on-site); [PS16]: handle cross-functional teams with care (involvement and motivation vs. personal goals, also [PS18]); [PS11]: keep informal communication channels (e.g., instant messaging); [PS11], [PS7]: use pair programming to train and improve knowledge; *Proven fine-grained practices and aids:* [PS29]: code buddies, pair programming (also [PS7]), TDD, simultaneous coding (maximize code ownership and responsibility); [PS13]: use dedicated testers with low turnover, allow for self-management (e.g., work patterns, sites' autonomy), coordination by mutual adjustment (also [PS5]); [PS5]: use Jira to support handover activities, testers participate in formal meetings; [PS8]: use rotating team ambassadors (also [PS17]), manage barriers (e.g., language, culture), share work practices; [PS27]: build cohesive teams (also [PS20], [PS17]), implement distributed quality assurance, build trust, supplement informal communication with documentation;
Product Owner	[PS19]: misunderstood requirements across sites; [PS21]: communication with remotes (e.g., remote area product owner); [PS1]: scaling large enterprises; [PS1]: defining the product owner role; [PS22]: keep areas separated;	[PS19]: use extensive follow-up questions to spot communication misunderstandings (especially concerning requirements); [PS21]: have one "chief" product owner and proxies; [PS21], [PS23], [PS22]: define area product owners to scale; [PS21]: locate architects of the area product owner near the development team and create a virtual team with regular (e.g., bi-weekly) meeting to decide priorities; [PS1]: product owners have the nine functions groom, prioritize, release master, technical architect, governor, communicator, traveller, intermediary, and risk assessor; [PS18]: responsibilities must be clarified—setting up a product owner team;
Scrum Master	[PS4]: lacking political power (technical management and Scrum master), no shielding from Scrum master on last minute changes;	[PS19]: appoint a backup Scrum master; [PS4]: Scrum master needs to be a strong negotiator; [PS20]: for distributed teams, on-site Scrum master participates in off-site meetings; [PS2]: the Scrum master is a process anchor, stand-up facilitator, impediment remover, sprint planner, Scrum-of-Scrums facilitator, and an integration anchor;

and knowledge management. In particular, from the perspective of the core process (Table IV), most of the identified challenges and resulting solution proposals address intra-/inter-team communication and collaboration (e.g., organizing and holding meetings or sharing information in the backlog) and scaling the general process. Scaling the general process comprises different proposals to extend the Scrum role model by different refinements [PS21], [PS23], [PS22] or by introducing proxies [PS21]. Furthermore, scaling the general management and keeping the communication stable were considered challenging, and different approaches are suggested, like virtual teams, development practices customized for GSE, or utilizing more comprehensive management approaches like Scrum-of-Scrums [PS19], [PS20], [PS23], [PS24] or even entire frameworks, e.g., LeSS or SAFe that bring agile methods closer to traditional processes in terms of project organization and management [PS23]. Interestingly, the identified literature does not discuss Schwaber's Nexus framework [18], which appears to

© IEEE. PREPRINT/EARLY VIEW. This is the author's version of the work.
It is posted here by permission of IEEE for your personal use. Not for redistribution.
The definitive version was published in the conference proceedings.

TABLE V
SOLUTIONS, PRACTICES, AND SUGGESTIONS TO OVERCOME CHALLENGES IN USING SCRUM (EXTENDED PRACTICES) IN DISTRIBUTED ARRANGEMENTS.

Process Element	Challenges	Solutions, Practices, and Suggestions
Adoption	[PS25]: create shared understanding of a change, enable for end-to-end development, bridge cultural gaps, create transparency among sites; [PS18]: resistance towards change;	[PS19], [PS20]: review a Scrum document describing the practice during sprint planning, and use the sprint planning for training; [PS19]: have a visiting engineer or a Scrum master in the first iteration, and relocate system experts in the teams; [PS4]: document meeting minutes for lessons learned and successful practices; [PS25]: involve all sites early and broadly at all levels; [PS25]: create a competence exchange program (regular communication, cross-site visits, joint infrastructures); [PS18]: build a leadership team with an Agile and Lean mindset; [PS26]: the team "owns" the change and thus should have a change agent;
Coach	[PS26]: achieving continuous SPI	[PS20], [PS17]: provide proper training to all sites is fundamental; [PS17], [PS26]: ensure executive support at all sites; [PS17]: regular meetings with senior management and coaches, permanently assigned coaches in the first period (incl. traveling between sites, e.g., every 1–2 months); [PS18]: set up a *Coaching Community of Practice*, and increase number of coaches and streamline the coaching; [PS26]: consider coaching a long-term relationship; [PS26]: involve everybody from the case project team; [PS26]: do not give solution, but teach teams to reason from the beginning, do not push changes; [PS26]: prefer small improvement steps to a big bang; [PS26]: avoid resistance by explaining the need properly;
Communication	[PS19], [PS20]: cultural differences in communication (e.g., Europeans and Asians); [PS11]: tech. barriers to set up videoconferencing; [PS27]: required formal documentation;	[PS20]: use multiple communication modes (e.g., code buddies as communication proxies; [PS29]); [PS17]: establish rituals (e.g., penalties for delayed communication); [PS25]: maximize communication (e.g., frequency); [PS27], [PS12]: keep informal communication but use official (formal) channels;
Meetings	[PS19], [PS20]: quality in video-conferencing; [PS19], [PS20]: communication distance (understanding, problem explanation); [PS11]: keep meetings jointly;	[PS19]: define rules to guide meetings; [PS17]: consider Kaizen workshops (start Kanban rather than Scrum); [PS17]: hold distributed team leader meetings; [PS18]: provide high quality video-conference equipment; [PS19]: hold "unofficial" meetings after the daily sprint with people interested to discuss a topic (e.g., discussing new feature); [PS19]: provide one room per team (ease co-located communication);
Requirements	[PS4]: meeting/discussion with little documentation, resistance for sign off (waterfall practice); [PS16]: involvement of developers and testers;	[PS16]: set up on virtual Scrum team in which requirements are discussed and the client is involved; [PS27]: plan iterations to finalize requirements and develop designs, document requirements at different levels of formality;
Scrum (general)	[PS16]: different delivery speeds and outdated SLAs; [PS22], [PS21], [PS24], [PS25], [PS1], [PS6], [PS2], [PS18], [PS13], [PS23]: scaling;	[PS17]: develop a vision to motivate continuous improvement, use Scrumban; [PS6]: ensure senior management support; [PS4], [PS12]: set up a global Scrum board; [PS27], [PS7]: maintain product/process repository to facilitate knowledge sharing; Large-scale Scrum (LeSS), Scaled Agile Framework (SaFE), or Disciplined Agile Delivery (DAD) [PS23];
Scrum-of-Scrums	[PS24]: scaling the process;	[PS19], [PS20]: meet once a week with one rotating member per team and all Scrum masters; [PS24]: talk about impediments only;
Sprint (general)	[PS4]: time pressure lead to reduced QA instead of de-scoping;	[PS19]: keep 20% free capacities to handle fast track issues
Success Factors	[PS18]: lacking continuous integration and test automation;	[PS19]: an experienced company and competent people increase success probability; [PS19]: Scrum supports more frequent communication, which reduces risk of misunderstanding with (positive) impact on quality and motivation; [PS28]: knowledge management must not be underestimated; [PS19]: use nightly builds and automated testing, rule: the team that breaks the build fixes it; [PS18]: develop staff and infrastructure for continuous integration;
User Stories	[PS4] use cases separated from user stories make navigation difficult; [PS11]: keep the practice in distributed cases;	[PS4] integrate use cases with user stories; [PS21]: arrange workshops for new user stories with architects and/or design, developers; [PS11]: clarify picked stories with BA and QA; [PS12]: development should demonstrate compliance before testing;
Visits	[PS14]: cost, impact on development and maintenance of team knowledge;	[PS11], [PS25], [PS18], [PS7], [PS12]: organize regular visits; [PS19], [PS20], [PS27]: frequent visits to remote team (2–4 weeks; more frequently for planning) to consolidate and work effectively together (co-located kick-off and knowledge transfer period; [PS17]), and organize an annual gathering (e.g., a long weekend including leisure activities for team building, trust); [PS11]: implement (continuous) staff rotation to improve team cohesion (e.g., on-site domain experts, [PS20]; developer exchange, [PS29]) and knowledge transfer (also in [PS7], [PS12]); [PS14]: analyze invest/cost of exchange for knowledge categories of interest, organize focused meetings (i.e., task-, team-, process-, and goal-related);

be in line with the agile principles and does not introduce such heavy management infrastructures as suggested, e.g., by SAFe. The analysis of the extended set of practices and challenges in Table V draws a similar picture. For instance, a number of authors discuss ways of organizing face-to-face meetings and visits (e.g., [PS11], [PS25], [PS18], [PS7], [PS12]) to overcome distance and to develop team spirit. Scrum in particular (and agile methods in general) fundamentally relies on interaction. However, distance (geographical, temporal, and cultural [16], [17]) makes such interactions challenging. That is, GSE puts pressure on one of the most basic principles of agile software development, and a considerable share of the analyzed articles aims to propose strategies to overcome this issue, e.g., by discussing virtual teams, refining the Scrum roles, or presenting technical solutions to support good communication and collaboration environments.

From our analysis we thus conclude that besides the people-centric challenges of GSE (culture or language [16]) and the

© IEEE. PREPRINT/EARLY VIEW. This is the author's version of the work.
It is posted here by permission of IEEE for your personal use. Not for redistribution.
The definitive version was published in the conference proceedings.

TABLE VI
OVERVIEW OF SYSTEMATIC SECONDARY (SMS AND SLR) AND TERTIARY (SSLR) STUDIES.

Ref.	Year	Focus	PS	Overlap	Overlap details (Ref ID of original study and ID of study at hand)
[19]	2009	SLR: Scrum in GSD	20	3	S1 [PS19], S7 [PS4], S8 [PS10]
[20]	2010	SMS: Agile practices in GSE	77	5	PS22 [PS19], PS5 [PS4], PS23 [PS20], PS11 [PS10], PS30 [PS27]
[21]	2011	SSLR: Agile trends in GSE	12	0	–
[22]	2016	SLR: Communication in GDAD	21	2	S21 [PS29], S7 [PS10]

physical constraints (global and temporal distance [17]), the methods of project organization and management constitute another set of significant challenges. Especially agile methods like Scrum are prone to conflicts between agile software development principles and GSE constraints. Most of the challenges and solution proposals analyzed indicate that scaling Scrum properly is one of the most challenging problems in the context of GSE. A number of solution approaches exist (occasionally even contradicting ones like team-specific vs. common/shared backlogs [PS19], [PS18]) that shows applying Scrum to global software engineering to be everything but a solved problem.

V. RELATED WORK

Since its inception in the early 1990's [10], Scrum has been a success in terms of adoption. Current literature reports on numerous software engineering teams that have been adopting Scrum (mostly as hybrid engineering approach in combination with other approaches) to guide their processes [2]–[4], [6]. Over the years, a large amount of literature was published on agile software development [5], and several secondary studies (i.e., reviews of the available literature) have been presented. During the execution of this systematic review, some of these studies have been identified and retrieved; they range from ad-hoc reviews [23] to systematic mapping studies [20] and systematic reviews [19], [22], and even tertiary studies [21].

Table VI provides an overview of the publications among those that were conducted following a systematic procedure. In 2009, Hossain et al. [19] studied the challenges related to using Scrum in a globally distributed scenario. Their review identified a significant amount of challenges and practices that should be utilized to mitigate the impact of those challenges, e.g., visits and team gatherings to foster trust among team members and facilitate knowledge sharing. Besides the Scrum-of-Scrums, no other practice was identified to support the scaling of Scrum. In their large-scale systematic mapping study, Jalali and Wohlin [20] found evidence supporting the observation that many companies customize agile practices to address the specific characteristics of projects (also supported by [4], [6]). In their tertiary study from 2011, Hanssen et al. [21] confirmed "*both globalization and "agilization" of software companies are stable trends for the future*", but state an impact analysis of the challenges of distribution on the agile principles still missing. Finally, Alzoubi et al. [22] reviewed empirical studies reporting on communication challenges in distributed agile development. They provided a large set of challenges impacting communication and of respective techniques for mitigation, which were grouped into strategies, tools, and agile practices.

The present study differs in two aspects from the reviews outlined above: First, we retrieved primary studies using the backward snowballing technique [12], which gave us access to a set of literature not biased by the tokens of the search strings used in reviews based on an automatic search process through digital libraries. Looking at the more than 100 primary studies identified by the aforementioned reviews, the overlap between these sets and our 29 primary studies is minimal, i.e., five at most (Table VI). Second, we filtered the retrieved publications by using a quality criterion based on two parameters of a maturity index [9] (Table I), which allowed us to quantify the relevance and maturity of the empirical evidence presented.

VI. CONCLUSION

We presented a systematic review providing an in-depth analysis of the available body of knowledge discussing Scrum in (globally) distributed software engineering. The study is based on a dataset [9], which investigates the advances of the GSE community and, via backward snowballing, eventually identified 29 primary studies reporting empirical evidence on applying Scrum to GSE. We found 45 challenges in 19 categories in total, and we identified reported solutions, practices, and suggestions, which we structured and categorized using two perspectives: the first perspective is given by the Scrum "core" process (8 categories, 21 challenges) and the second perspective uses an extended set of practices crafted from analyzing the primary studies (11 categories, 19 challenges). Among the challenges identified, scaling Scrum in general, and adopting processes and practices in particular were the most frequently mentioned. By structuring and mapping the different solution proposals to the challenges identified, our study contributes a large amount of practices and strategies to help scaling Scrum to the distributed context.

Our findings indicate a trend in which "everybody" wants to use Scrum for different reasons, yet, all adopters need to (extensively) adjust the core assets of Scrum. That is, the core Scrum framework has to be considered a serious impediment to the deployment of agility in a (globally) distributed setup, but, due to Scrum's flexibility, assets that require extension can be modified accordingly. Thus, from our findings, we conclude that Scrum as such represents a serious barrier for (globally) distributed projects, and projects need to spend effort for preparing Scrum properly before deploying it to a distributed project.

© IEEE. PREPRINT/EARLY VIEW. This is the author's version of the work.
It is posted here by permission of IEEE for your personal use. Not for redistribution.
The definitive version was published in the conference proceedings.

A. Threats to Validity & Limitations

Even though this study has been conducted following an established and rigorous research instrument [11], it has limitations (most arising from particular design decisions taken) that we discuss in the following.

The *internal validity* is threatened by our study selection approach: First, the initial dataset [9] represents an idiosyncratic bias that might have affected the primary studies identified by the search strategy. Second, backward snowballing is an effective technique to retrieve related work [13], yet, it retrieves only older publications, and it is susceptible to the circular reference problem potentially causing very relevant articles from different communities to not be found. Third, the instrument suffers from subjective bias during the data curation process, and human errors might have been introduced in the data set. If the first issue has been ignored by design, we updated the initial dataset to the latest available material to address the second. Nonetheless, the update process was performed within the focus chosen for this study, i.e., the proceedings of the ICGSE conference series only. To address the third threat, more researchers were allocated to crucial steps (Figure 1), and consensus meetings were held in case of disagreements to either align the direction taken by the study or to discuss doubtful publications/data. Finally, both secondary studies and surveys as well as lessons learned and industrial experience reports have been excluded by design (Sect. II-B). If a secondary study or a survey was found, it was kept for discussion purposes, while lessons learned and industrial experience papers were excluded as they are unlikely grounded in rigorously documented empirical evidence.

The *external validity* is threatened by our result set, which potentially misses important articles. For instance, the decision to exclude lessons learned and industrial experience papers might affect the completeness of our result set. However, such material was often hard to evaluate, e.g., regarding root causes of reported challenges and the measurable consequences of presented solutions and practices. We therefore cannot claim to have provided the complete overview of the interplay between global software engineering and agile software engineering, which also limits the generalizability of our results. In fact, by excluding these publications when not following a strictly documented research method, it remains unknown whether our results properly reflect the reality in practice.

Finally, in addition to the external validity, the *conclusion validity* is also affected by uncertainty introduced by the result set. Even though we applied a rigorous selection procedure, a number of cases did not provide sufficient details to be fully categorized (which lead to the exclusion of [PS6]; Table III). Our conclusions thus suffer from partially poor reporting (see also [24] for publications in GSE). Therefore, to limit the bias in our conclusions, we framed the results solely via the Scrum process model as we could not provide a more fine grained presentation accounting for, e.g., the distribution type or the company size. These concerns motivate further investigation to improve the depth of our findings.

B. Future Work

Our selected approach leaves room for extension. First, due to our research design, results validity was preferred over their completeness. Due to this decision, relevant publications were left out based on either their year or since they were not referenced within the publications we analyzed (e.g., [25], [26]). Second, this study was focused on Scrum as representative of agile software development only. This might cause a limitation concerning the amount of practices, solutions, and suggestions identified. Nonetheless, the present paper provides a baseline as well as a framing instrument that has been designed and tested. The present paper thus lays the foundation for future studies to obtain a refined and more comprehensive picture.

ACKNOWLEDGEMENTS

This research was supported in parts by the Danish Agency for Science, Technology and Innovation under the project "Next Generation Technology for Global Software Development", Grant No. #10-092313.

REFERENCES

[1] J. D. Herbsleb, "Global software engineering: The future of socio-technical coordination," in *Future of Software Engineering*, ser. FOSE. IEEE Computer Society, 2007, pp. 188–198.

[2] Version One, "The 10th annual state of agile report," Online: http://stateofagile.versionone.com, 2016.

[3] L. R. Vijayasarathy and C. W. Butler, "Choice of software development methodologies: Do organizational, project, and team characteristics matter?" *IEEE Software*, vol. 33, no. 5, pp. 86–94, Sept 2016.

[4] G. Theocharis, M. Kuhrmann, J. Münch, and P. Diebold, "Is Water-Scrum-Fall reality? on the use of agile and traditional development practices," in *Proccedings of International Conference on Product Focused Software Development and Process Improvement*, ser. LNCS. Berlin, Heidelberg: Springer, December 2015, vol. 9459, pp. 149–166.

[5] T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Information and Software Technology*, vol. 50, no. 9–10, pp. 833 – 859, 2008.

[6] P. Diebold, J.-P. Ostberg, S. Wagner, and U. Zendler, "What do practitioners vary in using scrum?" in *International Conference XP*. Springer, 2015, pp. 40–51.

[7] F. P. Brooks, "No silver bullet essence and accidents of software engineering," *IEEE Computer*, vol. 20, no. 4, pp. 10–19, 1987.

[8] G. K. Hanssen, D. Šmite, and N. B. Moe, "Signs of agile trends in global software engineering research: A tertiary study," in *6th International Conference on Global Software Engineering Workshops*, ser. ICGSE-W. IEEE, Aug 2011, pp. 17–23.

[9] C. Ebert, M. Kuhrmann, and R. Prikladnicki, "Global software engineering: Evolution and trends," in *11th International Conference on Global Software Engineering*, ser. ICGSE. IEEE, Aug 2016, pp. 144–153.

[10] J. Sutherland and K. Schwaber, "The Scrum guide. the definitive guide to Scrum: The rules of the game," Availble from Scrum.org, oct 2013.

[11] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," Keele University, Tech. Rep. EBSE-2007-01, 2007.

[12] T. Greenhalgh and R. Peacock, "Effectiveness and efficiency of search methods in systematic reviews of complex evidence: audit of primary sources," *BMJ*, vol. 331, no. 7524, pp. 1064–1065, November 2005.

[13] D. Badampudi, C. Wohlin, and K. Petersen, "Experiences from using snowballing and database searches in systematic literature studies," in *International Conference on Evaluation and Assessment in Software Engineering*, ser. EASE. ACM, 2015, pp. 17:1–17:10.

[14] M. Ivarsson and T. Gorschek, "A method for evaluating rigor and industrial relevance of technology evaluations," *Empirical Software Engineering*, vol. 16, no. 3, pp. 365–395, 2011.

[15] A. Strauss and J. Corbin, *Basics of qualitative research: Techniques and procedures for developing grounded theory* . Sage Publications, 1998.

© IEEE. PREPRINT/EARLY VIEW. This is the author's version of the work.
It is posted here by permission of IEEE for your personal use. Not for redistribution.
The definitive version was published in the conference proceedings.

[16] J. Noll, S. Beecham, and I. Richardson, "Global software development and collaboration: Barriers and solutions," *ACM Inroads*, vol. 1, no. 3, pp. 66–78, Sep. 2011.

[17] P. J. Ågerfalk and B. Fitzgerald, "Flexible and distributed software processes: Old petunias in new bowl?" *Communications of the ACM*, vol. 49, no. 10, pp. 27–34, 2006.

[18] K. Schwaber, "Nexus guide, version 1.1," 2015. [Online]. Available: https://www.scrum.org/Portals/0/NexusGuide%20v1.1.pdf

[19] E. Hossain, M. A. Babar, and H. y. Paik, "Using scrum in global software development: A systematic literature review," in *4th International Conference on Global Software Engineering*, ser. ICGSE. IEEE, July 2009, pp. 175–184.

[20] S. Jalali and C. Wohlin, "Agile practices in global software engineering - a systematic map," in *5th IEEE International Conference on Global Software Engineering*, ser. ICGSE. IEEE, Aug 2010, pp. 45–54.

[21] G. K. Hanssen, D. Šmite, and N. B. Moe, "Signs of agile trends in global software engineering research: A tertiary study," in *6th International Conference on Global Software Engineering Workshops*, ser. ICGSE-W. IEEE, Aug 2011, pp. 17–23.

[22] Y. I. Alzoubi, A. Q. Gill, and A. Al-Ani, "Empirical studies of geographically distributed agile development communication challenges: A systematic review," *Inform. & Manag.*, vol. 53, no. 1, pp. 22–37, 2016.

[23] S. V. Shrivastava and H. Date, "Distributed agile software development: A review," *Comput. Sci. and Engin.*, vol. 1, no. 1, pp. 10–17, May 2010.

[24] A. R. D. R. Techio, R. Prikladnicki, and S. Marczak, "Reporting empirical evidence in distributed software development: An extended taxonomy," in *10th International Conference on Global Software Engineering*, ser. ICGSE. IEEE, July 2015, pp. 71–80.

[25] J. M. Bass, "How product owner teams scale agile methods to large distributed enterprises," *Empirical Software Engineering*, vol. 20, no. 6, pp. 1525–1557, 2015.

[26] P. Bjørn, M. Esbensen, R. E. Jensen, and S. Matthiesen, "Does distance still matter? revisiting the cscw fundamentals on distributed collaboration," *ACM Trans. Comput.-Hum. Interact.*, vol. 21, no. 5, pp. 27:1–27:26, Nov. 2014.

PRIMARY STUDIES

[PS1] J. M. Bass. Agile method tailoring in distributed enterprises: Product owner teams. In *8th International Conference on Global Software Engineering*, ICGSE, pages 154–163. IEEE, Aug 2013.

[PS2] J. M. Bass. Scrum master activities: Process tailoring in large enterprise projects. In *9th International Conference on Global Software Engineering*, ICGSE, pages 6–15. IEEE, Aug 2014.

[PS3] G. Borrego, A. L. Morán, R. Palacio, and O. M. Rodríguez. Understanding architectural knowledge sharing in agsd teams: An empirical study. In *11th International Conference on Global Software Engineering*, ICGSE, pages 109–118. IEEE, Aug 2016.

[PS4] M. Cristal, D. Wildt, and R. Prikladnicki. Usage of scrum practices within a global company. In *3rd International Conference on Global Software Engineering*, ICGSE, pages 222–226. IEEE, Aug 2008.

[PS5] D. S. Cruzes, N. B. Moe, and T. Dybå. Communication between developers and testers in distributed continuous agile testing. In *11th International Conference on Global Software Engineering*, ICGSE, pages 59–68. IEEE, Aug 2016.

[PS6] S. Dorairaj, J. Noble, and G. Allan. Agile software development with distributed teams: Senior management support. In *8th International Conference on Global Software Engineering*, ICGSE, pages 197–205. IEEE, Aug 2013.

[PS7] S. Dorairaj, J. Noble, and P. Malik. Knowledge management in distributed agile software development. In *Agile Conference*, AGILE, pages 64–73. IEEE, Aug 2012.

[PS8] Siva Dorairaj, James Noble, and Petra Malik. Bridging cultural differences: A grounded theory perspective. In *Proceedings of the 4th India Software Engineering Conference*, ISEC, pages 3–10, New York, NY, USA, 2011. ACM.

[PS9] Hans-Christian Estler, Martin Nordio, Carlo A. Furia, Bertrand Meyer, and Johannes Schneider. Agile vs. structured distributed software development: A case study. *Empirical Software Engineering*, 19(5):1197–1224, 2014.

[PS10] Helena Holmström, Brian Fitzgerald, Pär J. Ågerfalk, and Eoin Ó. Conchúir. Agile practices reduce distance in global software development. *Information Systems Management*, 23(3):7–18, 2006.

[PS11] S. Modi, P. Abbott, and S. Counsell. Negotiating common ground in distributed agile development: A case study perspective. In *8th International Conference on Global Software Engineering*, ICGSE, pages 80–89. IEEE, Aug 2013.

[PS12] Sunila Modi, Pamela Abbott, and Steve Counsell. Raising awareness in distrbuted agile development: A case study perspective. In *UK Academy for Information Systems Conference Proceedings*, pages 26:1–26:23. Association for Information Systems, 2013.

[PS13] N. B. Moe, D. Cruzes, T. Dybå, and E. Mikkelsen. Continuous software testing in a globally distributed project. In *10th International Conference on Global Software Engineering*, ICGSE, pages 130–134. IEEE, July 2015.

[PS14] N. B. Moe, T. E. Fægri, D. S. Cruzes, and J. E. Faugstad. Enabling knowledge sharing in agile virtual teams. In *11th International Conference on Global Software Engineering*, ICGSE, pages 29–33. IEEE, Aug 2016.

[PS15] T. Niinimaki. Face-to-face, email and instant messaging in distributed agile software development project. In *6th International Conference on Global Software Engineering Workshops*, ICGSE-W, pages 78–84. IEEE, Aug 2011.

[PS16] R. Noordeloos, C. Manteli, and H. V. Vliet. From rup to scrum in global software development: A case study. In *7th International Conference on Global Software Engineering*, ICGSE, pages 31–40. IEEE, Aug 2012.

[PS17] M. Paasivaara. Coaching global software development projects. In *6th International Conference on Global Software Engineering*, ICGSE, pages 84–93. IEEE, Aug 2011.

[PS18] M. Paasivaara, B. Behm, C. Lassenius, and M. Hallikainen. Towards rapid releases in large-scale xaas development at ericsson: A case study. In *9th International Conference on Global Software Engineering*, ICGSE, pages 16–25. IEEE, Aug 2014.

[PS19] M. Paasivaara, S. Durasiewicz, and C. Lassenius. Distributed agile development: Using scrum in a large project. In *3rd International Conference on Global Software Engineering*, ICGSE, pages 87–95. IEEE, Aug 2008.

[PS20] M. Paasivaara, S. Durasiewicz, and C. Lassenius. Using scrum in distributed agile development: A multiple case study. In *4th International Conference on Global Software Engineering*, ICGSE, pages 195–204. IEEE, July 2009.

[PS21] M. Paasivaara, V. T. Heikkilä, and C. Lassenius. Experiences in scaling the product owner role in large-scale globally distributed scrum. In *7th International Conference on Global Software Engineering*, ICGSE, pages 174–178. IEEE, Aug 2012.

[PS22] M. Paasivaara and C. Lassenius. Scaling scrum in a large distributed project. In *International Symposium on Empirical Software Engineering and Measurement*, ESEM, pages 363–367. IEEE, Sept 2011.

[PS23] M. Paasivaara and C. Lassenius. Scaling scrum in a large globally distributed organization: A case study. In *11th International Conference on Global Software Engineering*, ICGSE, pages 74–83. IEEE, Aug 2016.

[PS24] M. Paasivaara, C. Lassenius, and V. T. Heikkilä. Inter-team coordination in large-scale globally distributed scrum: Do scrum-of-scrums really work? In *Proceedings of the ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM, pages 235–238. ACM, Sept 2012.

[PS25] M. Paasivaara, C. Lassenius, V. T. Heikkilä, K. Dikert, and C. Engblom. Integrating global sites into the lean and agile transformation at ericsson. In *8th International Conference on Global Software Engineering*, ICGSE, pages 134–143. IEEE, Aug 2013.

[PS26] Maria Paasivaara and Casper Lassenius. How does an agile coaching team work?: A case study. In *Proc. of International Conference on Software and Systems Process*, ICSSP, pages 101–109. ACM, 2011.

[PS27] Balasubramaniam Ramesh, Lan Cao, Kannan Mohan, and Peng Xu. Can distributed software development be agile? *Commun. ACM*, 49(10):41–46, October 2006.

[PS28] M. A. Razzak and D. Šmite. Knowledge management in globally distributed agile projects – lesson learned. In *10th International Conference on Global Software Engineering*, ICGSE, pages 81–89. IEEE, July 2015.

[PS29] F. Zieris and S. Salinger. Doing scrum rather than being agile: A case study on actual nearshoring practices. In *8th International Conference on Global Software Engineering*, ICGSE, pages 144–153. IEEE, Aug 2013.