The Journal of Systems & Software 183 (2022) 111110

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

Why and how is Scrum being adapted in practice: A systematic review☆

Michal Hron [a,*], Nikolaus Obwegeser [b]

[a] Aarhus University, Fuglesangs Allé 4, 8210 Aarhus V, Denmark
[b] Bern University of Applied Sciences, Brückenstr. 73, 3005, Bern, Switzerland

ARTICLE INFO

Article history:
Received 9 September 2020
Received in revised form 30 September 2021
Accepted 4 October 2021
Available online 9 October 2021

Keywords:
Scrum
Agile development
Systematic literature review
Information systems development

ABSTRACT

Scrum, recognized today as the most popular agile development methodology, has been used in a wide range of settings and for varying purposes, in- and outside of the traditional software development context. The use of Scrum in non-traditional settings and for different needs led to a considerable corpus of academic literature that investigates, presents, and discusses modifications to the original method, aimed to make it fit such novel forms of application. Based on a large-scale review of extant literature, this study systematically analyses why and how Scrum was reportedly modified in different instances and contributes with a synthesis that can serve as a basis for a more systematic approach to future research and practice. We explicate nine common modification objectives for change (e.g., attaining high performance, non-standard contexts, distributed development) mapped against seven generic modification strategies (e.g., method guidance, new procedures, or artifacts). Building on our extensive literature analysis we highlight research gaps and identify promising areas for future research.

1. Introduction

Two decades after the publication of the Manifesto of Agile Software development (Beck et al., 2001b), agile methodologies have gained widespread acceptance (Dingsør et al., 2012), and Scrum has emerged as the most popular method from the agile family (VersionOne, 2020). The original design of Scrum best accommodates small, co-located groups of developers with diverse skills, working on software for a client that is actively involved in the development process (Schwaber, 1995). Despite that, due to "its perceived simplicity and 'lightweight' approach" (Masood et al., 2020, p. 1), Scrum has today been adopted by a range of institutions in a variety of contexts, ranging from management outside of software engineering to managing entire companies (Cloke, 2007; Greening, 2010). Many of those circumstances require adaptations or augmentations of different aspects of the method, which led to a substantial and growing body of research studies discussing modifications to Scrum (Diebold et al., 2015; Forbes Insights, & Scrum Alliance, 2018).

While this widespread attraction of Scrum to different contexts outside the software engineering domain can be considered a success and confirmation for the developers of the method, it also led to a vast variety of different adaptations (Fitzgerald et al., 2002a,b). Method customization and tailoring are commonplace and well-recognized phenomenon within the software engineering domain since the complexity and contingencies of different real-world contexts can hardly be reflected in a unified methodological approach (Goldkuhl and Karlsson, 2020). Yet, some Scrum evangelists strongly advocate following the outlined practices "by the book" to avoid introducing the same inefficiencies and dysfunctional elements that one set out to eliminate by introducing Scrum in the first place (Schwaber, 2007).

The motivation to adapt a method to a specific context is arguably even more accentuated when moving outside its originally intended discipline, i.e., outside the software engineering domain. However, to the detriment of cumulative knowledge building, such modifications have largely been reported in a fragmented manner with a focus on individual empirical case studies. Many research results are presented in regional conferences and have not been effectively assimilated towards a unified body of knowledge. Subsets of this literature, typically corresponding to a specific aspect (i.e., distributed development), have been reviewed in earlier studies (Kitchenham et al., 2010). However, a comprehensive mapping of the literature on the adaptation of Scrum that would map the reasons for *why* it is being modified against *how* it has been modified, has not been conducted. This comes at the expense of future researchers, seeking to build on prior findings rather than repeating what others already studied and practitioners who would benefit from informed guidance to support their context-specific needs. Bringing together the

☆ Editor: Kelly Blincoe.

* Corresponding author.
Michal Hron (M. Hron), Nikolaus Obwegeser (N. Obwegeser).
E-mail addresses: hron@mgmt.au.dk (M. Hron), nikolaus.obwegeser@bfh.ch (N. Obwegeser).

https://doi.org/10.1016/j.jss.2021.111110
0164-1212/

Why and How of context-specific method adaptations is thus important to enable both effective learning from past results and a systematic approach towards future research.

This study aims to address this research gap through by synthesizing extant research on modifications proposed to Scrum. We conducted a large-scale, systematic literature review (SLR) of documented cases in the academic literature to accomplish that. Our study is guided and structured by the following research question: *Why* and *how* is Scrum adapted to different contexts?

The paper first discusses related works before briefly introducing the reader to the basic properties of the original Scrum method. After outlining the research methodology in Section 4, we discuss the results of our literature review and present the descriptive results outlining the overall shape of the literature in Section 5. In Section 6, we synthesize extant knowledge by extracting and mapping seven modification strategies against nine modification objectives reported in the academic literature. The paper concludes with an interpretation and discussion of the findings with emphasis on potential future research areas as well as implications for practitioners.

2. Related works

The primary literature mapping adjustments to agile methods has been regularly summarized by a number of systematic reviews. So much so that a review of those reviews is available (Hoda et al., 2017). Some reviews document the spread of the paradigm itself (Dingsør et al., 2012) or various uses ranging from global software engineering (Dreesen et al., 2016) to the adaptation of agile methods to conform to standards like CMMI (Palomino et al., 2017). This paper contributes to the literature on agile development and Scrum by providing an exhaustive overview of the reported adaptations with emphasis on the methodological approach used.

Our work differs from other recent reviews on agile software development through two main aspects. We focus on a specific methodology (Scrum) instead of agile methods as a whole, and we aimed to inclusively evaluate and review all potentially relevant studies, including both empirical and conceptual/theoretical study designs, instead of using hard cut-off criteria in our search strategy. Concerning the focus on a single method, it can be observed that most reviews single out a specific development situation and then survey the whole range of agile development methods on this topic. Recent examples include global systems engineering (Hossain et al., 2009; Jalali and Wohlin, 2010) or user-centered practices (Duechting et al., 2007). We pursued the opposite approach: we focused on a specific method and the explored circumstances to which it been introduced, and how. Pertaining to our filter and search strategy, we find that while research questions similar to ours have been addressed by prior studies (Ashraf and Aftab, 2017; Diebold et al., 2015), their methodologies differ markedly from our study in terms of relying on a non-systematic sampling of literature or generalizing from a small sample of practitioners. Related works are found in industry reports (Forbes Insights, & Scrum Alliance, 2018; VersionOne, 2020). While those reports typically provide data on individual methods, they are not methodologically transparent and lack detail in their results.

This paper expands on a previous conference paper discussing preliminary results of the same research project (Hron and Obwegeser, 2018). We engaged in a broader literature search and filter process, including more academic databases (three instead of one) and no filtering based on citation count or year of publication. Consequently, our sample increased from 31 to 925 studies, supporting a substantially deeper analysis and discussion of the phenomenon of Scrum adaptations.

3. A brief review of scrum

Scrum was originally proposed by Schwaber (1995) as a simple development methodology that embodies iterative and incremental development principles. It comprises so-called ceremonies (also referred to as rituals or procedures), artifacts, and roles. As an actively used methodology, it has been regularly updated by its creators. The most recent version of the official Scrum Guide was released in November 2020 (Schwaber and Sutherland, 2020). The Scrum Primer provides a concise alternative overview of the method (Deemer et al., 2012).

Development under Scrum is split into so-called sprints, which are typically between two and four weeks long. Sprint planning occurs at the beginning of each sprint and is used to define what can be delivered in the next sprint and what needs to be done to achieve that. At the end of each sprint, a sprint retrospective is held to support continuous learning and improvement in the team. In addition, a sprint review is conducted to demonstrate the outcome of the sprint to the customer and gather feedback and relevant information for the next working increment. Scrum places weight on developer autonomy but not at the expense of discipline. Developers start their working days with stand-up meetings to update each other on progress and tasks ahead. In the spirit of self-organization, instead of being led by a project manager, a Scrum team is facilitated by a Scrum Master. The main point of contact is the product owner, who also manages the backlog of work to be done. In successful Scrum development, the customer regularly participates by contributing ideas for new features (that are recorded as user stories and maintained in a backlog) and by signing off on completed features at the end of each sprint. Lastly, Scrum offers tools to track the productivity of the team by measuring task velocity that can be plotted on a so-called burn-down chart. The team estimates how time-consuming particular development tasks will be (expressed in story points) and self-assigns work for a given sprint. The number of backlog items, expressed in story-points, realized over time gives us a measure of team productivity or "velocity" in Scrum parlance. Measuring velocity not only motivates teams by seeing their productivity and progress on the burn-down chart, but it also helps in planning future work.

Scrum was developed for small, co-located teams of diverse specializations (Schwaber, 1995) and – like many other agile development methods – is built on certain assumptions (Ramesh et al., 2017; Turk et al., 2005). This includes, for example, the assumption of availability of the customer for frequent interactions, which can be difficult to achieve in practice.

4. Research method

The objective of this study to understand why and how Scrum was modified, as reported in the academic literature. Agile development with Scrum has been the subject of a growing number of academic studies, and therefore, a systematic review that allows identification, systematization, and evaluation of extant contributions is in order (Kitchenham, 2007).

Due to the variety of applications of Scrum inside and outside of software engineering, the research questions call for an inherently transdisciplinary review that spans across various literature streams, e.g., software engineering, computer science, information systems, management (Besson and Rowe, 2012).

To ensure transparency and replicability of our search, filter, and analysis process, we opted to follow a systematic approach to scope and review the existing literature (Paré et al., 2015; Webster and Watson, 2002) in conjunction with specific recommendations for conducting systematic literature reviews (SLRs) in the software engineering domain (Kitchenham, 2007). Fig. 1 gives an overview of our research design comprising literature search, filter, coding, and analysis.

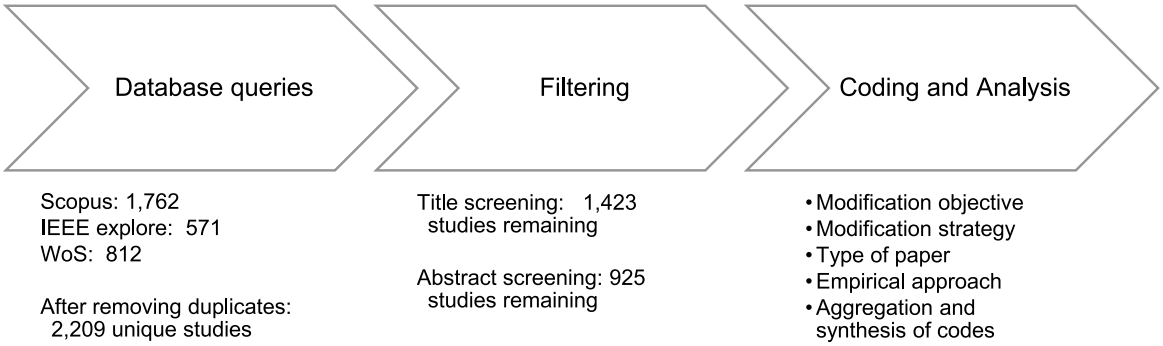

Fig. 1. Step-by-step literature search and analysis strategy.

Table 1
Structured search query.

Positive terms	Negative terms	Neutral terms	Topic specifiers
exten*	limit*	demand*	Scrum,
accommodat*	drawback*	requirement*	software,
improve*	shortcoming*	need*	agile,
adapt*	issue	suit*	excluding Rugby,
widen*	challeng*	modif*	excluding sport
focus*	downside*	alter*	
enhance*	reduc*	fit*	
expand*		revis*	
		chang*, tailor*,	
		scop*	

4.1. Database queries

The literature sample was collected using a structured query from three major academic databases. Due to the interdisciplinary nature of our study, we opted to search for relevant literature both in broad databases that cover a wide range of different disciplines, and specific databases that focus on software engineering research. To this end, Scopus and Web of Science databases were selected as general-purpose databases with wide coverage. Additionally, IEEE Explore was consulted as a domain-specific database that also provides higher coverage of conferences.

The query was developed over multiple iterations of exploring different search strategies and subsequent discussion among the authors. The final query string specifies the method in question (Scrum) and a set of terms that operationalize our focus on changes to the Scrum method. We opted to include a thesaurus of words to specify the change aspect, which can be represented by either positive (e.g., improve, enhance), negative (e.g., limit, drawback), or neutral (e.g., modify, change) terms. Lastly, we used an exclusion clause for sports literature using the same terms (i.e., Scrum as a part of the Rugby gameplay). Consequently, our Scopus search string was given as follows: TITLE-ABS-KEY (“Scrum”) AND ALL (“software” AND “agile”) AND ALL ([positive terms] OR [negative terms] OR [neutral terms]) AND NOT ALL (“rugby” AND “sport”). Wherever possible, we used the placeholder “*” to allow for word variations. Analogical queries were developed for the other two databases. A complete list of our search terms and thesaurus can be found in Table 1.

The query executed in mid-August 2020 returned 3145 papers since 2002, which formed the basis for our three-step screening and filter process.

We merged the results from the three databases first by matching on the title and then by manual checking. Scopus provided the largest number of articles, so we checked for duplicates against the results of the Scopus query. With each of the remaining articles, we performed manual checking for duplicates. Even though the same organization might have been studied more than once, multiple studies reported on different subsets of the organization. For example, Yahoo! was used as a case in a study on integrating process framework (Scotland and Boutin, 2008) and how to approached distributed development (Lee and Yong, 2010).

Conversely, we treated studies reporting on multiple cases as one record, as we were focused on what the authors considered the most significant alteration to Scrum. This is a lesser concern as only 8% of our sample presented multiple cases. The initial sample after elimination of duplicates consisted of 2209 articles which were subsequently subject to title and abstract screening.

4.2. Filtering

To conduct the screening process, we defined a clear set of criteria to be applied in the process. We first screened all articles for titles that indicated modifications of Scrum and only removed those that focused on other topics. If in doubt, the article was included for later abstract screening.

4.2.1. Title screening

We engaged in a check-coding process, where both authors screened the titles of the same 100 papers separately to strengthen the reliability of the coding process (Knudsen, 1975). The degree of inter-reviewer reliability was measured using Cohen’s kappa ($\varkappa$) (Spitzer et al., 1967), resulting in $\varkappa = 0,96$, which indicates a high degree of agreement among the authors. Cohen’s kappa was calculated as follows:

Kappa coefficient

$$\varkappa = \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)} = \frac{0.98 - 0.50}{1 - 0.50} = 0.96 \quad (1)$$

Pr(a) is the relative observed agreement among the researchers, and Pr(e) is the hypothetical probability of chance agreement.

After establishing clarity and agreement concerning the selection criteria, we split the remaining papers between both authors and continued to screen the titles for relevance separately. Out of the initial 2209 articles, we found 1423 as potentially relevant based on titles alone.

4.2.2. Abstract screening

We separately screened the remaining papers’ abstracts (after exchanging the sets of articles to review) according to the same criteria. This reduced our sample to 1054. Some papers were eliminated in full-text screening (including previous literature reviews), resulting in the final sample of 925 publications. Papers were included if they fulfilled two mandatory conditions:

Condition 1: the paper introduces a modification or extension of the Scrum methodology

Condition 2: the paper states a reason or motivation for introducing such a change

While the body of this article presents a wealth of papers in the final sample, the inclusion criteria are best explained by presenting examples that were excluded. For example, the first condition was not met by a survey that aims to study the effects of Scrum on productivity without modifying it (Hanslo et al., 2020). The second condition was not fulfilled, for example, by a case study by Moe and Aurum (2008), which derives several pieces of advice for decision making in Scrum, but it is driven by an objective to understand decision making in Scrum teams and not by a desire to modify Scrum motivated by an explicit objective.

In line with the mandatory conditions for inclusion described above, we generally excluded papers about learning how to use Scrum (i.e., the process of adopting the method) as well as papers that investigated how to teach Scrum at universities, if they did not also describe modifications to the method (e.g., Santos et al., 2015). We also disregarded papers on the (psychological or sociological) dynamics of Scrum teams, which offered academic descriptions of phenomena like team maturity but little prescriptive potential for actual modifications of the method (e.g., Hasnain and Hall, 2009).

4.3. Coding and analysis

Considering our research questions, the main goal of our coding efforts was to extract categories along two major dimensions from each paper in the final sample: *why* a modification was intended, i.e., the modification objective, and *how* Scrum was to be modified, i.e., the modification strategy. To ensure a systematic approach, our coding and analysis efforts took inspiration from relevant method literature on the synthesis of prior literature (Cruzes and Dybå, 2011). The coding process was performed by both authors in a similar fashion as the screening and filter process. First, we coded a set of the same 50 articles separately, inductively developing and extracting coding categories. For example, several papers motivated their method adaptation by emphasizing the need for managerial control and embedding the software development process in the organization's overall strategy, which we coded as "Managerial Extension". Similarly, in terms of modification strategies, we found that many papers focused on explicating how to properly use and apply the Scrum method; thus we created the modification strategy "Method Guidance".

After the initial coding, we compared and consolidated the coding categories until an agreement was reached. Thereafter, we coded another set of 50 articles separately, keeping in mind the preliminary codes already established but also adding new codes when necessary. For example, a category for "Documentation" was used initially, but it was subsequently merged into "Performance" or "Managerial Extensions" as the number of papers dedicated to practices of documenting software did not saturate a category on par with the other clusters. Subsequently, we again compared and consolidated our codes before splitting the remaining articles between the authors and coded separately. Lastly, we compared and discussed our full set of codes to reach an agreement on the final categorization scheme. The full coding table is available as an online Appendix.

As our main analytical dimensions, we coded for modification objective (primary and secondary) and modification strategy (primary and secondary). The primary objective refers to the main goal stated in the study that triggered the method adaptation, while secondary objectives were also coded for when mentioned. For example, Vlietland and Van Vliet (2015) motivate their study by the need to scale Scrum to multiple teams in enterprise settings, and address the managerial extensions necessary to govern the collaboration across teams. Similarly, primary modification strategy refers to the main adaptation approach used, which was in some cases accompanied by a secondary strategy. For example, De Melo et al. (2019) propose an adaptation of Scrum for small companies that build on the introduction of a new artifact (a point-based heuristic) but also describe new tools (based on the Balanced Scorecard technique). We also coded for several descriptive elements in our sample (see Table 2). To categorize the methodological approach of papers in our sample, we distinguished between case studies (single, multiple) experiments, conceptual proposals (i.e., theoretical developments with either none or limited empirical validation), simulations, and survey-based studies. Inspired by Dikert et al. (2016), we also evaluated and coded for the quality of the papers on a scale from 1–5, with the lowest score applied for studies with unclear research objectives, apparent flaws, or in transparent methodological section and the highest score for papers that are well anchored in prior literature, identifying a clear research gap and objective, applying a systematic and rigorous method and critically reflecting on their findings. However, we decided not to use the quality dimension as a cut-off or filter criterion to ensure our sample represents the full breadth of the current research landscape.

5. Results

In our sample period, we observed a steadily rising volume of academic output, suggesting growing interest in the topic. Fig. 2 visualizes the growth of the relevant academic research since 2002, indicating both the number of studies per year as well as the accumulated growth of overall citations in the sampling period. Due to the differences in citation count across the various databases, we opted to use a single source for citations – Google Scholar – regardless of which database the paper was originally found in. Google Scholar reportedly provides more accurate citation results than Scopus or Web of Science, especially for Social Sciences, Arts and Humanities, and Engineering due it is better coverage of books, conference proceedings, and a wider range of journals (Harzing and van der Wal, 2008).

In terms of publication outlets, we observed that, on average, about 80% of the papers were presented at conferences, further evidencing the pragmatism of the field. The majority of studies reside in the long tail of the distribution: over 200 papers were published in the proceedings of conferences, colloquia, or workshops, where each event generated one or two papers in our sample. Among the most frequently used outlets are conference proceeding series like Springer's *Lecture Notes in Computer Science* (46) and *Lecture Notes in Business Information Processing* (44), as well as journals like *Information and Software Technology* (11) and the *Journal of Systems and Software* (11).

From a methodological perspective, the largest part of our sample followed a single case study design (477). Apart from that, we also counted experiments (36), multiple case studies (86), experiments (36 studies), conceptual proposals (227), simulations (11), and survey-based studies (88).

Regardless of their methodological approach, we found that less than 10 percent of studies in our sample contain an explicit theoretical underpinning; that is, they have a dedicated description of a theoretical lens or embed their research into related studies and theories. The Manifesto for Agile Software Development principles, the CMMI framework, or the 4 + 1 Software Safety principles (Doss and Kelly, 2016a), are sometimes referred to, in place of a fully developed theory (Conboy, 2009; Lee and Xia, 2010). Notable exceptions to the "theory-light" majority of studies in our sample include, for example, a study exploring agility as organizational learning (Lyytinen and Rose, 2006) or another one discussing the "base patterns" of agile approaches

Table 2
Coding categories.

Coding category	Range of values
Primary Motivation Secondary Motivation	Inductive
Primary Solution	Inductive
Empirical	Binary: yes or no
Method	Single case study, multiple case study, experiment, conceptual proposal, simulation, survey-based
Theoretical framing	Theory or model applied to frame research
Citation count	Citations from Google Scholar
Quality appraisal	Evaluation of quality (range from 1 to 5)

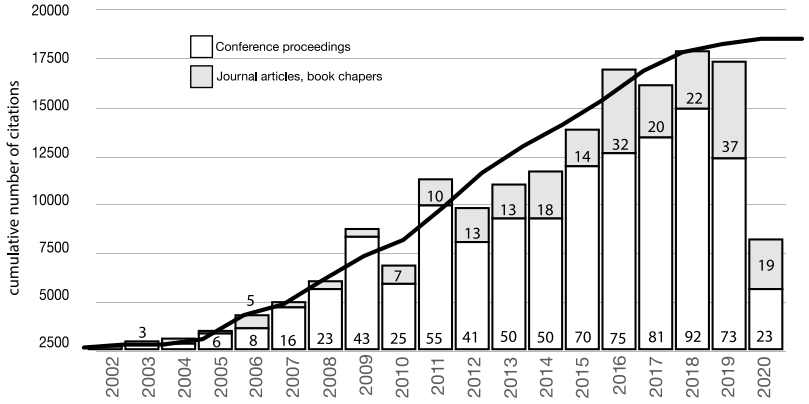

Fig. 2. Publications and citations over time.

Table 3
Methods, quality appraisals, and examples.

Method	Experiment	Multiple case	Conceptual proposal	Simulation	Single case	Survey
Empirical	*1*	*1*	*0*	*0*	*1*	*1*
Quality	3.00	3.15	2.51	2.85	2.58	2.66
mean (St. dev)	(1.01)	(0.89)	(1.17)	(1.46)	(1.01)	(2.20)
Example	Harvie and Agah (2016)	Kettunen (2009)	Rahman et al. (2018)	Griffith et al. (2014)	Gary et al. (2011)	Lehnen et al. (2016)
Count	36	86	227	11	477	88

(Greening, 2016). Table 3 accompanies the analysis with information on empirics, quality appraisal ranging from one (worst) to five (best), and examples (Table 3).

Relying on Google Scholar as a standardized source of citations across databases, we calculated average citations per year for each paper. That means that for a paper from 2010, we divided the total number of citations by ten because it was published ten years ago. This adjustment allows for comparing old and new papers.

We uncovered that the average paper garners just 2.51 citations per year with a standard deviation 4.50. Minor variations can be found across the different methodological approaches (see Fig. 3). Conference publications are cited less (1.96 citations per year, St. dev = 2.93) than journal publications (4.36 citations per year, St. dev = 7.40).

6. Modification objectives and strategies

Our research identified nine generic modification objectives and seven common modification strategies. Before discussing each modification objective and associated strategies in detail here, we briefly introduce the main categories to guide the reader through our findings.

In response to the question: *Why is Scrum being modified*? Our review identified nine generic modification objectives. The most common was a simple desire to attain high *Performance*, typically in line with the project management triad (higher quality, lower cost, reduced time). The second most common was the objective of accommodating a specific *Context*. Other papers addressed the need to combine Scrum with another framework. We labeled such modification objectives *Juxtaposition*. *Architecting* refers to studies where the stated objective was to specify high-level development outcomes in release plans, technical specifications, or similar. *Distributed development* subsumes all efforts aimed at using Scrum in non-collocated but geographically distributed setups. The objective of *User Experience* captures those efforts that specifically emphasized the role of user experience when motivating their method adaptations. *Managerial extensions* refer to the linkage of the software development process to the business strategy of the organizations. Studies that focused on making Scrum work for large-scale projects that would exceed the traditional setting of Scrum are captured in the category *Size scaling*. Finally, some studies motivate their adaptations to Scrum through increased demand for *Security*, i.e., regulatory or industry-specific constraints or demands.

In response to the question: *How is Scrum being modified?* We have derived seven common modification strategies from our sample. *Method Guidance* includes appeals to "by the book applications" of the selected method, or reminders of the Manifesto for Agile Software Development principles. For example, Eloranta et al. (2016) identify a set of harmful Scrum practices and propose mitigating strategies. Specific advice may concern the daily stand-up meeting (Dick, 2016; Stray et al., 2013) or distributed teams (Lous et al., 2017). *Procedures* refer to introducing novel activities into the method that can be re-occurring (e.g. testing; Tuomikoski and Tervonen, 2009) or conducted on a one-off basis (e.g. during hand-over; Anwar et al., 2014). *Infusion* describes an in-depth reconsideration of multiple elements of Scrum, which

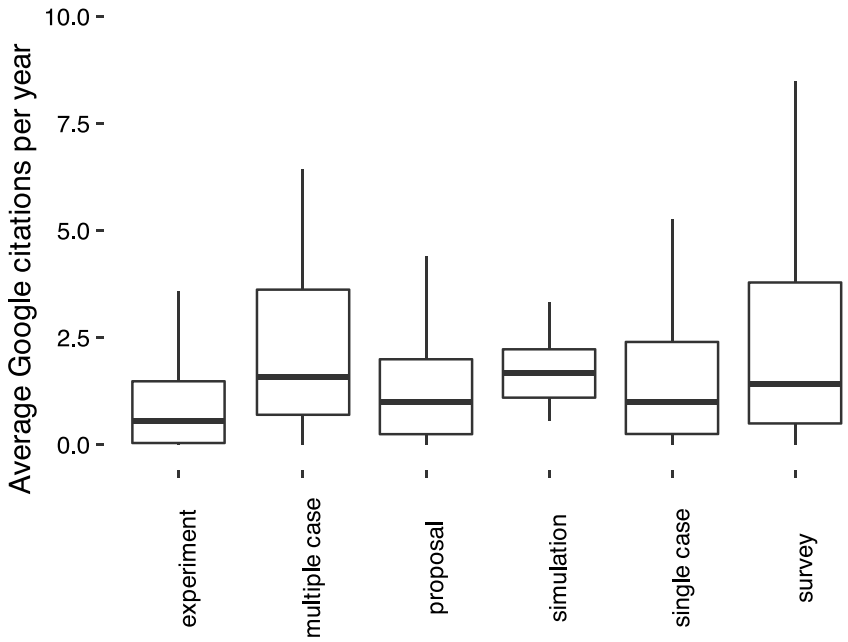

Fig. 3. Average google scholar citations of studies across methodological approaches.

either has a basis on other existing processes/methods such as CMMI (Jakobsen and Johnson, 2008; Sutherland et al., 2007), Lean (Sikamani and Raj Dharmapal, 2016; Wang et al., 2012) or is done to address new contexts, such as new product development (Cooper and Sommer, 2016b; Lehnen et al., 2016). *Tools* refer to proposals of tools that do not directly modify Scrum but help accomplish a certain task. Such tools can often be seen as a form of "plug-ins" to the original method that may be applied without directly changing the method in its working. New metrics are typical examples of this category, as they are small additions that can enrich Scrum without significantly changing the method (Greening, 2015; Padmini et al., 2015). *Artifacts* include the presentation of new or altered artifacts for Scrum, e.g., the use of digital displays for co-located teams (Schwarzer et al., 2016) or distributed settings (Esbensen et al., 2015). *Pre-development* refers to introducing additional processes, artifacts, and sometimes roles that specifically deal with tasks such as the definition of technical architecture, articulating a product vision, or creating milestones for development, before the development itself is initiated. Multiple perspectives can be adopted in formulating pre-development modifications. Designers sometimes call for personas to understand the task at hand (Sedeño et al., 2017); management can be concerned with release planning (Heikkila et al., 2015a; Heikkilä et al., 2015b). Technical requirements may need to be included, which are often difficult to address under the agile paradigm (Bourimi and Tesoriero, 2014). *Multiplicity* describes the multiplication of an artifact, a role, or the entire team. This was only coded when it was explicitly introduced as a novel amendment. The multiplicity of the team is inevitably present under circumstances of distributed or large-scale developments (Paasivaara and Lassenius, 2011). Teams may be multiplied according to specialization: e.g., design team and a development team (Ferreira et al., 2011). The multiplication of artifacts can refer to a separate backlog, dedicated only to testing (Aamir and Khan, 2017).

Table 4 provides a complete overview of modification objectives (rows) mapped against modification strategies (columns), including the number of papers in which the given modification strategies were invoked in response to which modification objective. The shading of each cell corresponds to the number it displays, i.e., darker shadings indicate a higher number of studies for a specific combination of objective and strategy. In the following subsections, we will present each modification objective following a structured approach. We will first describe the category pertaining to our research context and then discuss the most common modification strategies and common examples as given in the summary tables at the beginning of each subsection. Finally, we will point to dedicated literature or reviews that focus on this particular aspect of Scrum or related agile software development practices.

6.1. Performance

The most frequent objective for the modification of Scrum is to attain high performance under standard development circumstances (co-located, with a small team) or with limited deviation from such a setting. The objective is to deliver results in line with the logic of the project management triad (time, quality, cost). Performance is a broad category that houses numerous, typically practice-derived, approaches that do not necessarily result in deep, structural changes to the method. Rather, these papers frequently codify minor alterations and additions, often found through usage and experimentation (see Table 5)..

The most common modification strategies are proposals of new procedures, which are often paired with Method Guidance as a secondary modification strategy. Papers also propose the development of tools with procedures as secondary modification strategies and method guidance. For example, one paper proposes a novel process for testing (Tuomikoski and Tervonen, 2009). Other studies develop metrics for prioritizing backlog elements (Kayes et al., 2016) or tools for alternative progress measurements (Miranda and Bourque, 2010). A typical example of using method guidance as a strategy for improved performance is Eloranta et al. (2016), in which the authors identify 14 anti-patterns and provide guidance on how to address them. Artifacts are less common to achieve general performance enhancements, but an example of a "flexible cooperative task board" to enhance daily meetings illustrates how they can be used (Rubart and Freykamp, 2009). In some cases, performance boosts are sought by large-scale rethinking of Scrum (Alqudah and Razali, 2018).

Table 4
Number of papers for modification objectives and modification strategy.

Modification Objective	Methodological Approach	*Method Guidance*	*Procedures*	*Infusion*	*Tools*	*Artifacts*	*Pre-development*	*Multiplicity*	*Totals*
Performance	*Non-empirical*	7	22	3	22	4	0	0	237
	Empirical	58	64	4	38	13	2	0	
Context	*Non-empirical*	6	13	20	1	0	0	0	173
	Empirical	54	22	48	1	5	1	2	
Architecting	*Non-empirical*	7	7	3	7	4	8	1	115
	Empirical	8	30	5	11	7	17	0	
Juxtaposition	*Non-empirical*	8	0	26	0	0	0	1	90
	Empirical	7	2	43	1	2	0	0	
Distributed Development	*Non-empirical*	43	12	4	5	4	0	7	87
	Empirical	7	2	1	2	0	0	0	
Managerial Extensions	*Non-empirical*	2	7	0	11	1	0	1	66
	Empirical	8	16	7	6	4	0	3	
User Experience	*Non-empirical*	0	8	2	1	1	0	0	65
	Empirical	12	16	8	1	3	9	4	
Size Scaling	*Non-empirical*	1	1	1	0	0	0	1	53
	Empirical	20	14	4	0	1	0	10	
Security	*Non-empirical*	6	6	2	4	0	0	0	39
	Empirical	6	6	4	1	2	0	2	
Totals		*260*	*248*	*185*	*112*	*51*	*37*	*32*	925

Some recurrent pre-development procedures are proposed, for instance, to prevent risks.

The most frequent secondary modification objectives to Performance are found in the Managerial extensions and related Architecting category. This hints at the desire of practitioners to imbue Scrum with more rigidity for increased performance.

Table 5
Primary and secondary modification strategies for Performance.

Modification strategy	Primary use	Secondary use	Example of primary use
Method Guidance	65	43	Eloranta et al. (2016)
Procedures	86	33	Tuomikoski and Tervonen (2009)
Infusion	7	2	Alqudah and Razali (2018)
Tools	60	20	Kayes et al. (2016)
Artifacts	17	5	Rubart and Freykamp (2009)
Pre-development	2	2	Ali et al. (2016)
Multiplicity	0	2	

The attention given to this category speaks highly of the quality of the original Scrum method. It is robust enough to fit a variety of contexts and designed to address an enduring problem. The prevalence of method guidance reports is indicative of the practitioner-grounded nature of the field.

6.2. Context

The second most common modification objective to modify Scrum is connected to specific development contexts that Scrum is introduced into. Scrum offers a notably simple approach to working, and it can be transposed to a number of different settings. Some of the encountered contexts include regulated environments (Cawley et al., 2010), maintenance operations (Heeager and Rose, 2015), and data warehousing projects (Goede, 2011). Outside of the software development domain, Scrum is seen as a promising method for innovation management (Cooper and Sommer, 2016a). It is also reported to be used in the construction industry (Streule et al., 2016) or animated film production (Sharma and Wherry, 2009) (see Table 6).

Table 6
Primary and secondary modification strategies for Context.

Modification strategy	Primary use	Secondary use	Example of primary use
Method Guidance	60	50	Heeager and Rose (2015)
Procedures	35	19	Wagner (2014)
Infusion	68	8	Cooper and Sommer (2016a)
Tools	2	1	Mendonça et al. (2014)
Artifacts	5	8	Streule et al. (2016)
Pre-development	1	2	Hanschke et al. (2015)
Multiplicity	2	2	Aamir and Khan (2017)

The contexts listed include software development but also applications far outside of the context of software altogether. The context can motivate changes, but it can inspire how those changes should be implemented. While it is widely recognized that contextualization plays a major role in the practice of agile methods (Fitzgerald et al., 2002a,b; MacCormack and Verganti, 2003), the objective of papers in this group is not really to recognize the fungibility of the method as much as it is to document a catalogue of ready-made modifications that fit particular circumstances such as those listed above.

The generic modification strategy Infusion is used here to denote markedly altered Scrum with novel elements, and Scrum imbued with elements from specific contexts (Cooper and Sommer, 2016a). Method guidance is frequently invoked to address practical considerations stemming from the transposition of Scrum to a new environment (Heeager and Rose, 2015). It is also offered as a secondary modification strategy accompanying Infusion or Procedures. Smaller-scale alterations to Scrum most commonly add Procedures (Wagner, 2014). New artifacts are rare (Streule et al., 2016) as are Tools (Mendonça et al., 2014).

Many of the contexts Scrum is introduced into already have their codified methods and processes. Hence, we observe Juxtaposition as the most frequent co-occurring modification objective. Less commonly, we noted explicit mentions of modification objectives to boost the overall development performance with the Performance category.

6.3. Architecting

The Architecting category aggregates the common desire to specify the outcome of the development process, at least on a

Table 7
Primary and secondary modification strategies for Architecting.

Modification strategy	Primary use	Secondary use	Example of primary use
Method Guidance	15	20	Angelov et al. (2016)
Procedures	37	36	Gayer et al. (2016)
Infusion	8	3	Harvie and Agah (2016)
Tools	18	8	Farid (2012)
Artifacts	11	13	Alperowitz et al. (2017)
Pre-development	25	2	Pastrana et al. (2017)
Multiplicity	1	1	Griffith et al. (2014)

Table 8
Primary and secondary modification strategies for Juxtaposition.

Modification strategy	Primary use	Secondary use	Example of primary use
Method Guidance	15	23	Schar et al. (2016)
Procedures	2	8	McMahon (2016)
Infusion	69	3	Sutherland et al. (2007)
Tools	1	1	Ionica et al. (2017)
Artifacts	2	3	Santos et al. (2016a)
Pre-development	0	2	
Multiplicity	1	0	Stålhane et al. (2012)

high level. To that aim, studies commonly invoke modification strategies around new artifacts, accompanied by procedures as a secondary modification strategy. Specific pre-development activities are sometimes highlighted with artifacts accompanying them as a secondary modification strategy. While Scrum originally suggests that high-level planning and software architecture are conducted in a sprint that precedes actual development (Schwaber, 1995), many practitioners see a need to attend to those processes more carefully (see Table 7).

Because technical architects are not specified in the original Scrum, their role is not always clear (Angelov et al., 2016). Further complicating matters, the task of "Architecting" can be articulated in different ways depending on the chosen background. A technically-minded team member may call for "traceability procedures" (Gayer et al., 2016) or a better approach to the pre-development phase and improving requirements elicitation (Pastrana et al., 2017) which can be supported with tools (Farid, 2012). Particularly, non-functional requirements seem to pose a challenge (Farid, 2012). A User-Experience specialist may address this modification objective by proposing artifacts such as "visual backlogs" (Alperowitz et al., 2017). A business-minded member of the team may be concerned with release planning (Li et al., 2010) on top of requirements elicitation. The results of a simulation study advise maintaining a parallel backlog for technical debt issues, therefore calling for multiplicity (Griffith et al., 2014). Despite the fragmentation by domains, high-level reconsiderations (Infusion) of Scrum to achieve greater architectural discipline have been proposed (Harvie and Agah, 2016). Taken together, these suggestions have a commonality and address the unease resulting from the agile principle of "responding to change" (Beck et al., 2001a) and give rise to a tension between emergence and control.

Architecting modifications are often introduced with a secondary modification objective of increasing the performance of the development process. The multitude of perspectives on what we define as the "architecting task" is reflected in the variety of other co-occurring modification objectives. Managerial extensions reflect the business perspective. The usability perspective is reflected in the User Experience category.

Outside the scope of our literature sample, further reading on the intersection of Scrum with Architecting in our cross-domain view can be found in the conceptual work by Madison (2010), as well as in reviews concerning specific domain-centered facets of architecting, most notably requirements engineering (Heikkila et al., 2015a; Heikkilä et al., 2015b).

6.4. Juxtaposition

The fourth most frequent modification objective occurs when Scrum is required to coexist with other codified frameworks, such as lean development (Wang et al., 2012) or CMMI (Sutherland et al., 2007). This juxtaposition of two codified standards can either be due to institutional requirements, motivated by the desire to reap benefits of both standards, or purely exploratory. Often, the juxtaposition of two methods is driven by the desire to signal quality. The result of such juxtaposition is usually a substantial rework or infusion of the original Scrum method with one or more other methods. Such rework is often accompanied with practical advice, or Method Guidance, as a secondary modification strategy (see Table 8).

By institutional requirements, we mean both top-down mandates, such as the Swiss standard Hermes (Schar et al., 2016), but also standards voluntarily adopted processes for quality assurance and signaling purposes, such as CMMI (McMahon, 2016) or industry standards (Stålhane et al., 2012). Some papers explore combinations of different methodologies, such as the Rational Unified Process (Silva et al., 2017). The primary literature also gives attention to the coexistence of agile and traditional methodologies. For example, the combination of the linear (waterfall) approach with the iterative approach of Scrum has deserved the name "Water-Scrum-fall" (Schlauderer and Overhage, 2015). Scrum can also be juxtaposed with techniques that provide tools like Quality Function Deployment (Ionica et al., 2017) or artifacts like UML models (Santos et al., 2016b).

A large share of papers in this category lack empirical insights (35 of 90). It is, however, interesting to note the order in which the different methods are adopted. In a common scenario, Scrum is introduced to an environment governed by a different standard. In that case, Scrum is reported to accelerate the speed of the development process and increase customer and developer satisfaction (Morales Trujillo et al., 2011). In another scenario, an organization already using Scrum chooses to combine it with a second method. In that case, Scrum is reported to imbue the development process with robustness and control, equipping developers with procedures for architecting.

Concerning co-occurring modification objectives, Juxtaposition was most commonly accompanied with Context because Scrum is often juxtaposed with a standard that originates in a particular setting. The combined methods often enable Scrum to perform better along traditional product management dimensions (on time, on budget, meeting requirements). Hence, Performance has been identified as a co-occurring modification objective. Lastly, because elements of other methods infused into Scrum often provide the method with additional rigidity and a more systematic approach, we note Architecting and Managerial Extensions as co-occurring modification objectives.

6.5. Distributed development

When development is not co-located, the development process must be amended to support distributed settings. Distributed teams do not necessarily have to be separated by several time zones; they can be distributed within a single country. However, offshoring and (partial) outsourcing are common reasons for why many development teams today are distributed. In addition to reduced communication bandwidth, difficulties can emanate from scheduling across time zones, language barriers, and cross-cultural communication issues. As a result, Method Guidance is the dominant modification strategy to overcome these challenges.

Table 9
Primary and secondary modification strategies for Distributed development.

Modification strategy	Primary use	Secondary use	Example of primary use
Method Guidance	50	17	Paasivaara et al. (2009)
Procedures	14	13	Szoke (2010)
Infusion	5	2	Banijamali et al. (2017)
Tools	7	1	Sungkur and Ramasawmy (2014)
Artifacts	4	2	Esbensen et al. (2015)
Pre-development	0	0	
Multiplicity	7	6	Lee and Yong (2010)

Interestingly, specific Procedures are more often mentioned as a secondary modification strategy, accompanying Method Guidance. Numerous studies discuss how new procedures or particular forms of multiplicity can support Scrum in a distributed environment. Overall, most studies recommend the multiplication of teams and sometimes procedures, while the main artifacts of Scrum (such as the backlog) are generally advised to be shared across teams (see Table 9).

For example, Paasivaara et al. (2009) offer a list of concrete supporting activities to facilitate distributed Scrum, such as reducing the independence of the sites and synchronization of work hours across sites. The communication challenge can be countered by artifacts such as a digital Scrum board (Esbensen et al., 2015) or knowledge management tools (Sungkur and Ramasawmy, 2014). Lee and Yong (2010) present a more comprehensive modification using the case of Yahoo!, including a multiplicity of procedures, new roles, and artifacts. In distributed environments, procedures for distributing work across teams are proposed (Szoke, 2010). Juxtapositions of methods can also be employed for distributed settings (Banijamali et al., 2017).

Papers addressing distributed development form a relatively mature and coherent body of work; therefore, only a few secondary modification objectives could be extracted. Most commonly, "performance", particularly in studies that go beyond the fundamental problem of establishing a distributed team but attempt to "fine-tune" work performance in distributed settings.

Outside the scope of our research and sample, dedicated literature reviews are available about distributed development, focusing, for example, on agile practices in global software development (Hossain et al., 2009; Vallon et al., 2018) or global software engineering (Jalali and Wohlin, 2010).

6.6. Managerial extensions

The "Managerial Extensions" objective can be split into two sub-parts: the addition of higher levels of software product management (such as road mapping and establishing a connection to a firm's overall strategy) on the one hand, and the other hand the integration of managerial tools for coordination, reporting, documentation, etc. Those modifications do not aim to close the gap between development and the strategic processes of the enterprise but rather equip the development process with more accountability and possibly an oversight. Modifications generally stay within the confines of the development process and improve aspects of coordination. Managerial Extensions are most frequently addressed by new procedures (Barton, 2009) with Method Guidance as a secondary strategy. Sometimes, Artifacts and accompanying procedures, as a secondary strategy, are offered to the existing roles to accomplish new managerial tasks. Method guidance is the third most common solution category (see Table 10).

In the first group, we find studies like one by Heikkilä et al. (2017), who demonstrate how the requirements flow from strategy to release in large-scale agile developments calls for various Scrum extensions, such as specialist roles, to manage the significant coordination effort. Other studies describe the need for managerial extensions in small business settings (Suwanya and Kurutach, 2009). In the second group, we find proposals for handling documentation (Duraisamy and Atan, 2013), tools like metrics (Greening, 2015).

Managerial Extensions are often accompanied by the introduction of Scrum to a new context. Besides the management function is a context of its own, we noted contexts such as a "services organization" (Barton and Campbell, 2007) or innovation management (Barton, 2009). The secondary modification objective Performance acknowledges explicit goals to improve the performance of the Scrum development, for instance, by overcoming the short-term focus (Dinakar, 2009) inherent in agile development. Co-occurrence of architecting is manifested in papers that modify Scrum by pre-development activities or procedures for documentation.

6.7. User experience

User experience is a critical factor for the acceptance of many software products today. Agile development methods, and Scrum, in particular, aim to accomplish high levels of usability by involving the user in the development process, as well as specifying user acceptance as the final condition that needs to be passed for a backlog item to be considered "done" (James, 2010). Nevertheless, practitioners concerned with the practical design and usability of software offer their work practices that can be absorbed into Scrum. Such practices most commonly take the form of procedures or experience-derived method guidance (Hodgetts, 2005). Sometimes, a codified framework addressing usability or formalized approach from the user experience field is juxtaposed with Scrum, resulting in a significant overhaul of Scrum or infusion (de Carvalho et al., 2016). In those cases, Method Guidance often accompanies the presentation of a radically overhauled Scrum specification. User Experience professionals are often involved at the beginning of the development process (Higuchi and Nakano, 2017), facilitating requirements elicitation (categorized as pre-development). Tools are also offered, albeit rarely (Peixoto, 2009). As a notable tension, some authors advocate for multiplying the development team and introducing multiplicity in the form of a separate team for designers (Budwig et al., 2009), while

Table 10
Primary and secondary modification strategies for Managerial extensions.

Modification strategy	Primary use	Secondary use	Example of primary use
Method Guidance	10	18	Heikkila et al. (2013)
Procedures	23	8	Barton (2009)
Infusion	7	0	Haidar et al. (2017)
Tools	17	6	Greening (2015)
Artifacts	5	7	Duraisamy and Atan (2013)
Pre-development	0	2	
Multiplicity	4	0	Hoda and Murugesan (2016)

Table 11
Primary and secondary modification strategies for User experience.

Modification strategy	Primary use	Secondary use	Example of primary use
Method Guidance	12	16	Hodgetts (2005)
Procedures	24	13	Cavichi de Freitas et al. (2016)
Infusion	10	3	de Carvalho et al. (2016)
Tools	2	3	Peixoto (2009)
Artifacts	4	5	Wautelet et al. (2016).
Pre-development	9	1	Higuchi and Nakano (2017)
Multiplicity	4	1	Budwig et al. (2009)

others claim that developers should be taught UX tasks (Øvad et al., 2015) (see Table 11).

The cluster of disciplines that includes User Experience, Usability, and Human–Computer Interactions, has developed a collection of field-specific methods and standards. In our sample, we can often see such techniques being juxtaposed with codified software development methods. Since the UX field is chiefly concerned with working towards a high level of fit between users and the developed product, these modifications often fall into the Architecting category (e.g., persona development, requirements elicitation).

Outside the scope of our research and sample, a recent systematic review summarizes approaches to incorporating UX in Scrum (Kikitamara and Noviyanti, 2018).

6.8. Size scaling

Today, Scrum is often applied in development efforts that outgrow the capability of a single, typical Scrum team (3–6 people; Schwaber, 1997, p. 16). In industrial projects to develop large software artifacts (e.g., Operating Systems), many teams usually contribute to developing the overall product. Increasing the size of the development team puts a strain on coordination, which calls for method guidance. Therefore, a common strategy is to split the developers into several Scrum teams (i.e., utilizing multiplicity). These teams are then arranged in structures such as "Scrum of Scrums" (Sutherland, 2005), which requires additional supporting infrastructure in the form of new procedures and roles. A tension then emerges with the core agile principle of self-organization (Beck et al., 2001a). The communication challenges are among those for which Method Guidance is provided as a secondary modification. In other cases, advice is offered as a primary strategy with procedures supporting it as a secondary strategy. Product line engineering can provide a mechanism of attaining control of the constellation of Scrum teams (Hofman et al., 2012). Size scaling can also be achieved by infusion with a different method such as the Rational Unified Process (Tanveer, 2016) (see Table 12).

As an example for method guidance in this context, Heikkila et al. (2013) present a decision framework for choosing between agile, hybrid, and plan-driven methods in large organizations. Other authors propose governance frameworks for multiple Scrum teams (Paasivaara et al., 2012; Vlietland et al., 2016). In addition to new roles and associated procedures, new artifacts can also prove useful to assist large-scale development (Bass, 2016).

Naturally, size scaling is often addressed jointly with distributed development settings (Bass, 2016). As teams grow in size, the supporting managerial infrastructure is sometimes extended to account for effective coordination among teams and to address alignment between the development work and strategic direction; hence we found "Managerial Extensions" as co-occurring motivation (Hofman et al., 2012). Agile development methodologies are sometimes forced to co-exist with plan-based methods in enterprise settings, resulting in juxtaposition as a co-occurring motivation.

Outside the scope of our research and sample, a recent literature review addressed scaling agile practices to large organizations (Kalenda et al., 2018).

Table 12
Primary and secondary modification strategies for Size scaling.

Modification strategy	Primary use	Secondary use	Example of primary use
Method Guidance	21	16	Heikkila et al. (2013)
Procedures	15	9	Hofman et al. (2012)
Infusion	5	3	Tanveer (2016)
Tools	0	1	
Artifacts	1	3	Bass (2016)
Pre-development	0	2	
Multiplicity	11	3	Sutherland (2005)

6.9. Security

Scrum is not explicitly concerned with the safety of the resulting software. It does not include procedures or roles for running tests and does not adhere to a specific standard of safety. However, there are numerous applications in which safety is of utmost importance. Papers in this category enable systematic ways of addressing security in Scrum (see Table 13).

Thirty-nine of the reviewed papers were primarily motivated by ensuring or improving safety-related aspects, and eight other papers mentioned security as a secondary driver for modifying Scrum. Software security presents a challenge because "established secure software development methodologies are mostly based on linear models… making them unsuitable for an agile environment" (Maier et al., 2017). In this limited sample, the most common modification strategies were introductions of new procedures like those for recurring safety analysis (Wang and Wagner, 2016) and Method Guidance, for example, in the form of principles (Doss and Kelly, 2016b). In our sample, explicit solutions to security were often connected with a specific context. In two instances, We noted that security practices from other formalized frameworks (i.e., juxtaposition) were introduced (Maier et al., 2017). Technical tools can help make testing more efficient (Ali and Ben Othmane, 2016), but even disciplined development of documentation, as additional artifacts, can be helpful (Wang et al., 2017). It is even possible to employ a multiplicity of procedures and dedicate a sprint specifically to security assurance (Koc et al., 2019). Secondary motivations largely mirror the primary ones.

The Security category is recognized as the smallest but still sufficiently distinctive. The growing number of papers that come from later years of the selection (earliest from 2013) may be a sign of an emerging research topic.

Table 13
Primary and secondary modification strategies for Security.

Modification strategy	Primary use	Secondary use	Example of primary use
Method Guidance	12	6	Doss and Kelly (2016b)
Procedures	12	8	Wang and Wagner (2016)
Infusion	6	4	Maier et al. (2017)
Tools	5	0	Ali and Ben Othmane (2016)
Artifacts	2	3	Wang et al. (2017)
Pre-development	0	0	
Multiplicity	2	0	Koc et al. (2019)

7. Discussion

Our study shows that Scrum is an incredibly versatile method used far beyond its original setting and purpose. As a result of its popularity, Scrum is used as a platform for adaptations to countless different contexts and objectives or as a building block for other methods. All this versatility has been documented in a rich corpus of literature. However, this highly dynamic research field has not been comprehensively integrated with a method-centric approach which stands in the way of cumulative knowledge generation. Such situation is common in emerging and highly active research areas (Keen, 1980). The strong practitioner interest acts as a further driver to the dissemination – and consequently adaptation – of the Scrum method to different contexts (Dybå,

2013; VersionOne, 2018). To this end, we hope that our review can help towards more systematic future research designs.

Our research is – to the best of our knowledge – the first that provides a structured overview of the modification objectives and adaptations to Scrum. Some modification objectives, such as *Performance* or adaptation to *Context,* attract much interest, while others like *Security* may see increased interest with the ongoing digitalization of our society and businesses. We use this discussion to first provide more details on our findings' empirical validity, before turning to implications and strategies for future research. Before concluding, we also acknowledge the limitations of our study.

7.1. Methodological diversity

For researchers and especially for practitioners, the validity of our findings is heavily dependent on the extent of their empirical grounding. It matters whether the reported proposals are just ideas or have been empirically tested in a case company. To inform our readers about the empirical grounding of each combination of modification objective and strategy (e.g., objective: Juxtaposition, strategy: Infusion), we investigated two specific questions for each combination: (i) what is the proportion of empirical studies vs. conceptual studies and (ii) how varied are the empirical approaches employed (e.g., single case studies, surveys) in any given combination. Table 14 gives an overview of our results.

For each combination of modification objective and strategy, the cell holds three lines of information. The first line addresses two proportions of empirical papers (i). Lines two and three address the methodological diversity (ii).

The number of empirical cases is given in the first line, followed by the absolute number of studies in brackets. For example, 43 (76) means that we found 43 empirical studies out of 76 total studies for this combination. These numbers on the first line of each cell already enable evaluating the empirical strength of a certain combination. While we are not imposing our scale or evaluation principle by categorizing papers (e.g., into low or strong empirics), we hope to support readers in their assessment by providing this overview.

Beyond empirical strengths, we employed the Gini coefficient as a measure for methodological pluralism, i.e., the concentration or dispersion or variety of different types of empirical approaches that have been used to study a certain combination (Roth, 2019). Again, we do not argue that one empirical design, e.g., a survey, may be more valuable than a single case study. On the contrary, there is common agreement among scholars that methodological pluralism is particularly beneficial to fields that study contemporary and complex phenomena, as it may offset the limitations of one method by the strength of another (Chamberlain et al., 2011; Frost et al., 2014). In the words of (Barnes et al., 2014), methodological plurality "produces more complex, richer understandings of the topic under investigation". (p. 35).

In each cell's second line, we show a measure of the concentration of empirical approaches for each combination, differentiating between single case studies, multiple case studies, survey designs, and experiments. To calculate the concentration, we relied on the widely accepted Gini coefficient for measuring statistical concentration (Gini, 1912). Specifically, we provide a normalized form of the Gini coefficient, which gives values from zero to one and enables easy interpretation. The highest values of the normalized Gini coefficient mean that all empirical studies in the cell are conducted using a single method. Zero denotes equal distribution across all methods. The calculation of the Gini coefficient was conducted using the *R* software package REAT from Wieland (2019, p. R4).

In the third line of each cell, we report the single most used – or dominant – empirical research design applied for this combination. If the Gini coefficient scores below 0.7, it indicates that a multitude of various empirical approaches has been applied with no single dominant design, in which case we noted the combination to be methodologically "diverse".

7.2. Implications for research and practice

As highlighted in Tables 4 and 14, the insights provided in this study, have direct implications for the advancement of the field in research and practice. For practitioners, our research helps to easily identify relevant studies that can guide situations where similar modification objectives arise and avoid unnecessary and costly trial and error experimentation.

For academics, our findings serve to reveal areas for promising future research activities. We show the relevance of certain combinations, as indicated by the number of studies that are pursuing method adaptations for a certain purpose, as well as strengths or weaknesses in terms of rigor, as indicated through the share of empirical studies and the breadth of empirical methods applied.

For example, practitioners may find it reassuring that "method guidance" as a means for the objective of "Managerial Extensions" has been studied empirically in eight out of ten studies, and that these studies applied a wide range of empirical approaches to look at this phenomenon (Gini = 0.50). On the other hand, when looking at the use of "Tools" for achieving increased "Security", only a single empirical study has been conducted out of five total studies, indicating a lack of information regarding the practical applicability of this modification approach.

The overwhelming focus on single case studies as a dominant empirical method without theoretical framing leads to the current state of "Dustbowl empiricism" (APA Dictionary of Psychology, 2018). More and more proposals addressing similar situations are being added to the corpus of literature. Our analysis shows that the literature lacks a cumulative research tradition, failing to link findings to previous research or benefit from relevant literature reviews. About 78% of the reviewed papers were presented at conferences, where the shorter format does not typically allow for an extended review of related literature. References to published literature reviews could be used in place of dedicated reviews in papers, but not a common practice. The current situation seemingly satisfies earlier calls for contextualized research in the software engineering domain (Clarke and O'Connor, 2012) which urges for "research to operate in clearly defined contexts, enabling us to identify realistic working assumptions and identify important, well-defined problems, as well as create opportunities for realistic evaluations" (Briand et al., 2017). However, throughout the literature, similar modifications introduced in response to near-identical modification objectives are reported repeatedly, and new publications motivated by previously addressed problems ignore previous literature. Such approach to research is concerning as it stands in the way of cumulative knowledge generation and risks the ongoing "reinvention of the wheel" (Hassan and Mathiassen, 2018; Keen, 1980).

For example, integrating User Experience professionals in Scrum, both proposals for integrating designers in the development team (Øvad et al., 2015) and keeping them separate (Singh, 2008) are documented as functional. Cross-comparison of those approaches, ideally with backing, in theory, would be interesting and highly relevant to the advancement of the field.

Table 14
Methodological pluralism across modification objectives and strategies.

	Method guidance	Procedures	Infusion	Tools	Artifacts	Pre-development	Multiplicity
Performance	58 (65) Gini: 0.48 diverse	64 (86) Gini: 0.53 diverse	4 (7) Gini: 0.5 diverse	38 (61) Gini: 0.81 single case	13 (17) Gini: 0.59 diverse	2 (2) Gini: 0.67 diverse	0 (0)
Context	54 (60) Gini: 0.86 single case	22 (35) Gini: 0.70 diverse	48 (68) Gini: 0.90 single case	1 (2) Gini: 1.00 single case	5 (5) Gini: 0.87 single case	1 (1) Gini: 1.00 survey	2 (2) Gini: 0.67 diverse
Architecting	8 (15) Gini: 0.50 diverse	30 (37) Gini: 0.53 diverse	5 (8) Gini: 0.87 single case	11 (18) Gini: 0.45 diverse	7 (11) Gini: 0.71 single case	17 (25) Gini: 0.8 single case	0 (1)
Juxtaposition	7 (15) Gini: 0.71 single case	2 (2) Gini: 0.67 diverse	43 (69) Gini: 0.75 single case	1 (1) Gini: 1.00 single case	2 (2) Gini: 1.00 single case	0 (0)	0 (1)
Distributed	43 (56) Gini: 0.72 single case	12 (15) Gini: 0.78 single case	4 (5) Gini: 1.00 single case	5 (7) Gini: 0.60 diverse	4 (4) Gini: 0.50 diverse	0 (0)	7 (7) Gini: 1.00 single case
Managerial extensions	8 (10) Gini: 0.50 diverse	16 (23) Gini: 0.42 diverse	7 (7) Gini: 0.90 single case	6 (17) Gini: 0.67 diverse	4 (5) Gini: 0.67 diverse	0 (0)	3 (4) Gini: 0.78 single case
User experience	12 (12) Gini: 0.61 diverse	16 (24) Gini: 0.67 diverse	8 (10) Gini: 0.67 diverse	1 (2) Gini: 1.00 multi. case	3 (4) Gini: 1.00 single case	9 (9) Gini: 0.48 diverse	4 (4) Gini: 1.00 single case
Size scaling	20 (21) Gini: 0.57 diverse	14 (15) Gini: 0.95 single case	4 (5) Gini: 0.83 single case	0 (0)	1 (1) Gini: 1.00 survey	0 (0)	10 (11) Gini: 0.80 single case
Security	6 (12) Gini: 0.78 single case	6 (12) Gini: 0.89 single case	4 (6) Gini: 1.00 single case	1 (4) Gini: 1.00 single case	2 (2) Gini: 1.00 single case	0 (0)	2 (2) Gini: 0.67 diverse

7.3. Strategies for future research

Based on our findings, we see two broad strategies that can help to substantially improve the systematic advancement of the field: improving cross-citation and theoretical framing.

Cross-citation requires future researchers to build on prior studies, and thus, to easily access and situate their research endeavor within the existent body of knowledge. The systematization offered in this paper is a step intended towards encouraging and enabling such efforts. Authors of future empirical work may use the presented synthesis of modification objectives and strategies to categorize their research plans against similar research and further follow the listed references to identify relevant prior studies easily. A variety of narrower, focused reviews is available, which have been referenced throughout this manuscript. This way, a systematic approach to assess the novelty of the intended or proposed activities is supported which may support the emergence of a more cumulative research tradition.

The second broad strategy to overcome the apparent lack of cumulative tradition is anchoring empirical research studies in theoretical framings. Without explicitly linking novel modifications to the underlying principles of agile approaches, it is difficult to assess which modifications will undermine it and which will enforce it. The lack of theoretical underpinning becomes particularly acute as the ideas of agile approaches have traveled from software development to generic project management and even to fields like organizational design (Gerster et al., 2020). Without theoretical consolidation and synthesis of key characteristics and mechanisms of the phenomenon, the literature stream is at risk of being reduced to a set of techniques and methodologies without facilitating real understanding (Cram and Newell, 2016). This concern echoes earlier calls for "the need for a better understanding of what constitutes 'agility'" (Abrahamsson et al., 2009; Conboy, 2009).

7.4. Limitations

We acknowledge that this study has certain limitations. We consistently aimed for the highest level of rigor and transparency in the literature search and filter process, yet there remains a risk of researcher bias resulting in having overlooked relevant studies that did not match our keyword thesaurus (Feldt and Magazinius, 2010). This risk was mitigated by ongoing discussion between researchers to design as comprehensive a filter as possible.

Many studies in the literature are written from the standpoint of advocating agile development and, hence the sample is skewed towards presenting positive results. This is accentuated by the fact that many of the publications are experience reports where the authors participated in the development. The reliance on experience reports further potential for bias in the sample. We, however, circumvented the absence of negative results by constructing our review around the effort of systematizing particular contexts and strategies where Scrum is reported to be successfully adopted.

Our coding process could have been affected by subjective bias. We attempted to counter this threat to validity by the ongoing discussion of the coding process and supplying supporting literature for our claims in the manuscript itself. Although alternative views of the data could be reached by different analysts (as is always the case when working with qualitative data), both authors agreed on the categorization scheme as robust and leading to the insightful discussion. We further addressed this potential limitation by discussing working versions of the manuscript with knowledgeable colleagues.

Our coding process was also limited concerning assessing validity threats and mitigation strategies (Feldt and Magazinius, 2010). Lacking a detailed analysis of validity for each study in our sample, we instead opted for a less granular quality assessment to evaluate primary literature, as described in Section 4.3.

Moreover, we limited our literature search to academic journal publications and conference proceedings but did not include relevant practitioner literature on the topic—often published in online

blogs or non-indexed industry magazines. While the large sample size of our review and the repeated occurrence of topics across studies in our sample provide ample evidence for our findings, we encourage future research to expand our insights, e.g., through the inclusion of or comparison with practitioner outlets.

8. Conclusion

We conducted a large-scale systematic literature review investigating *why* and *how* Scrum is being modified in practice. By reviewing a final sample of 925 studies, we developed nine categories of modification objectives connected to seven generic modification strategies. Our review confirms the applicability of Scrum to a variety of contexts (from standard small, co-located teams to large-scale distributed development) but also highlights the need for good practices to ensure that the benefits of using an agile method remain in place. The main contribution of this work is the synthesis of a fragmented body of literature that lacks a cumulative tradition. Through this synthesis, this study provides an exhaustive overview and discussion of extant knowledge, which at the same time can serve as a blueprint for future research.

CRediT authorship contribution statement

Michal Hron: Conceptualization, Methodology, Software, Formal analysis, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Nikolaus Obwegeser:** Methodology, Validation, Formal analysis, Writing – original draft, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We would like to thank Keld Pedersen for his comments to an earlier draft of this paper as well as the two anonymous reviewers.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jss.2021.111110.

References

Aamir, M., Khan, M.N.A., 2017. Incorporating quality control activities in scrum in relation to the concept of test backlog. Sadhana - Acad. Proc. Eng. Sci. http://dx.doi.org/10.1007/s12046-017-0688-7.

Abrahamsson, P., Conboy, K., Wang, X., 2009. Lots done, more to do: The current state of agile systems development research. Eur. J. Inf. Syst. 281–284. http://dx.doi.org/10.1057/ejis.2009.27.

Ali, A., Ben Othmane, L., 2016. Towards efficient security assurance for incremental software development the case of zen cart application. In: Proceedings - 2016 11th International Conference on Availability, Reliability and Security, ARES 2016. http://dx.doi.org/10.1109/ARES.2016.86.

Ali, M., Hafeez, Y., Hamid, B., 2016. An empirical study and a framework for effective risk management in scrum. Proc. Pak. Acad. Sci..

Alperowitz, L., Scheuermann, C., Von Frankenberg, N., 2017. From storyboards to code: Visual product backlogs in agile project courses. In: CEUR Workshop Proceedings, Vol. 1790. pp. 69–72.

Alqudah, M., Razali, R., 2018. An empirical study of scrumban formation based on the Selection of Scrum and Kanban Practices. Int. J. Adv. Sci. Eng. Inf. Technol. http://dx.doi.org/10.18517/ijaseit.8.6.6566.

Angelov, S., Meesters, M., Galster, M., 2016. Architects in scrum: What challenges do they face?. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), http://dx.doi.org/10.1007/978-3-319-48992-6_17.

Anwar, S.A., Hafeez, Y., Asghar, S., Shabbir Hassan, M., Hamid, B., 2014. Towards a framework for scrum handover process. Proc. Pak. Acad. Sci. 51 (4), 265–275.

APA Dictionary of Psychology, 2018. Dustbowl Empiricism. APA Dictionary of Psychology, https://dictionary.apa.org/dustbowl-empiricism.

Ashraf, S., Aftab, S., 2017. Latest Transformations in Scrum : A State of the Latest Transformations in Scrum : A State of the Art Review. I.J. Modern Education and Computer Science, http://dx.doi.org/10.5815/ijmecs.2017.07.02, July.

Banijamali, A., Dawadi, R., Ahmad, M.O., Similä, J., Oivo, M., Liukkunen, K., 2017. Empirical investigation of scrumban in global software development. Commun. Comput. Inf. Sci. http://dx.doi.org/10.1007/978-3-319-66302-9_12.

Barnes, J., Caddick, N., Clarke, N.J., Cromby, J., McDermott, H., Willis, M., Wiltshire, G., 2014. Methodological pluralism in qualitative research: Reflections on a meta-study. QMiP Bull. 17 (Spring).

Barton, B., 2009. All-out organizational scrum as an innovation value chain. In: Proceedings of the 42nd Annual Hawaii International Conference on System Sciences, HICSS. http://dx.doi.org/10.1109/HICSS.2009.57.

Barton, B., Campbell, E., 2007. Implementing a professional services organization using type c scrum. In: Proceedings of the Annual Hawaii International Conference on System Sciences. http://dx.doi.org/10.1109/HICSS.2007.265.

Bass, J.M., 2016. Artefacts and agile method tailoring in large-scale offshore software development programmes. Inf. Softw. Technol. 75, 1–16. http://dx.doi.org/10.1016/j.infsof.2016.03.001.

Beck, K., Beedle, M., Bennekum, A., Van Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D., 2001a. Manifesto for Agile Software Development. The Agile Alliance.

Beck, Kent, Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D., 2001b. Agile Manifesto. Software Development.

Besson, P., Rowe, F., 2012. Strategizing information systems-enabled organizational transformation: A transdisciplinary review and new directions. J. Strateg. Inf. Syst. http://dx.doi.org/10.1016/j.jsis.2012.05.001.

Bourimi, M., Tesoriero, R., 2014. Non-functional requirements for distributable user interfaces in agile processes. In: Proceedings of the 2014 Workshop on Distributed User Interfaces and Multimodal Interaction - DUI '14. http://dx.doi.org/10.1145/2677356.2677668.

Briand, L., Bianculli, D., Nejati, S., Pastore, F., Sabetzadeh, M., 2017. The case for context-driven software engineering research: Generalizability is overrated. IEEE Softw. 34 (5), 72–75. http://dx.doi.org/10.1109/MS.2017.3571562.

Budwig, M., Jeong, S., Kelkar, K., 2009. When user experience met agile. In: Proceedings of the 27th International Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '09, 3075. http://dx.doi.org/10.1145/1520340.1520434.

Cavichi de Freitas, R., Rodrigues, L.A., Marques da Cunha, A., 2016. AGILUS: A method for integrating usability evaluations on agile software development. In: International Conference on Human-Computer Interaction. Springer, pp. 545–552. http://dx.doi.org/10.1007/978-3-319-39510-4_50.

Cawley, O., Wang, X., Richardson, I., 2010. Lean/agile software development methodologies in regulated environments - State of the art. In: Lecture Notes in Business Information Processing, 65 LNBIP. pp. 31–36. http://dx.doi.org/10.1007/978-3-642-16416-3_4.

Chamberlain, K., Cain, T., Sheridan, J., Dupuis, A., 2011. Pluralisms in qualitative research: From multiple methods to integrated methods. Qual. Res. Psychol. 8 (2), 151–169. http://dx.doi.org/10.1080/14780887.2011.572730.

Clarke, P., O'Connor, R.V., 2012. The situational factors that affect the software development process: Towards a comprehensive reference framework. Inf. Softw. Technol. 54 (5), 433–447. http://dx.doi.org/10.1016/j.infsof.2011.12.003.

Cloke, G., 2007. Get your agile freak on! agile adoption at yahoo! music. In: Proceedings - AGILE 2007. http://dx.doi.org/10.1109/AGILE.2007.30.

Conboy, K., 2009. Agility from first principles: Reconstructing the concept of agility in information systems development. Inf. Syst. Res. 20 (3), 329–354. http://dx.doi.org/10.1287/isre.1090.0236.

Cooper, R.G., Sommer, A.F., 2016a. Agile-stage-gate: New idea-to-launch method for manufactured new products is faster, more responsive. Ind. Mark. Manage. 59, 167–180. http://dx.doi.org/10.1016/j.indmarman.2016.10.006.

Cooper, R.G., Sommer, A.F., 2016b. The agile–stage-gate hybrid model: A promising new approach and a new research opportunity. J. Prod. Innov. Manage. http://dx.doi.org/10.1111/jpim.12314.

Cram, W.A., Newell, S., 2016. Mindful revolution or mindless trend? Examining agile development as a management fashion. Eur. J. Inf. Syst. 25, 154–169. http://dx.doi.org/10.1057/ejis.2015.13.

Cruzes, D.S., Dybå, T., 2011. Recommended steps for thematic synthesis in software engineering. In: Proceedings of the 2011 International Symposium on Empirical Software Engineering and Measurement. In: ESEM '11, IEEE Computer Society, http://dx.doi.org/10.1109/esem.2011.36.

de Carvalho, J.M.I., da Silva, T.S., Silveira, M.S., 2016. Agile and UCD integration based on pre-development usability evaluations: An experience report. In: International Conference on Human-Computer Interaction. pp. 586–597. http://dx.doi.org/10.1007/978-3-319-39510-4_54.

De Melo, I.F., Mendes, G.A.R., Gelmetti, S.A., 2019. Non-scrum implementation: A methodological approach for small companies. In: Proceedings of the European Conference on Research Methods in Business and Management Studies, 2019-*June*. http://dx.doi.org/10.34190/RM.19.027.

Deemer, P., Benefield, G., Larman, C., Vodde, B., 2012. The Scrum Primer: A Lightweight Guide to the Theory and Practice of Scrum. http://www.infoq.com/minibooks/Scrum_Primer.

Dick, C., 2016. From the trenches | improving the scrum daily standup meeting. CrossTalk: J. Def. Softw. Eng..

Diebold, P., Ostberg, J.P., Wagner, S., Zendler, U., 2015. What do practitioners vary in using scrum? Lect. Notes Bus. Inf. Process. 212, 40–51. http://dx.doi.org/10.1007/978-3-319-18612-2_4.

Dikert, K., Paasivaara, M., Lassenius, C., 2016. Challenges and success factors for large-scale agile transformations: A systematic literature review. J. Syst. Softw. 119, 87–108. http://dx.doi.org/10.1016/j.jss.2016.06.013.

Dinakar, K., 2009. Agile development: overcoming a short-term focus in implementing best practices. In: Proceeding of the 24th ACM SIGPLAN Conference Companion on Object Oriented Programming Systems Languages and Applications - OOPSLA '09. p. 579. http://dx.doi.org/10.1145/1639950.1639952.

Dingsør, T., Nerur, S., Balijepally, V., Moe, N.B., 2012. A decade of agile methodologies: Towards explaining agile software development. J. Syst. Softw. 85 (6), 1213–1221. http://dx.doi.org/10.1016/j.jss.2012.02.033.

Doss, O., Kelly, T., 2016a. Addressing the 4+1 software safety assurance principles within scrum. In: ACM International Conference Proceeding Series. http://dx.doi.org/10.1145/2962695.2962712.

Doss, O., Kelly, T., 2016b. The 4+1 principles of software safety assurance and their implications for scrum. In: Lecture Notes in Business Information Processing. http://dx.doi.org/10.1007/978-3-319-33515-5_27.

Dreesen, T., Linden, R., Meures, C., Schmidt, N., Rosenkranz, C., 2016. Beyond the border: A comparative literature review on communication practices for agile global outsourced software development projects. In: Proceedings of the Annual Hawaii International Conference on System Sciences. pp. 4932–4941. http://dx.doi.org/10.1109/HICSS.2016.612.

Duechting, M., Zimmermann, D., Nebe, K., 2007. Incorporating user centered requirement engineering into agile software development. Proc. HCII 2007, 58–67.

Duraisamy, G., Atan, R., 2013. Requirement traceability matrix through documentation for SCRUM methodology. J. Theor. Appl. Inf. Technol. 52 (2), 154–159.

Dybå, T., 2013. Contextualizing empirical evidence. IEEE Softw. 30 (1), 81–83. http://dx.doi.org/10.1109/MS.2013.4.

Eloranta, V.P., Koskimies, K., Mikkonen, T., 2016. Exploring ScrumBut - An empirical study of scrum anti-patterns. Inf. Softw. Technol. 74, 194–203. http://dx.doi.org/10.1016/j.infsof.2015.12.003.

Esbensen, M., Tell, P., Cholewa, J.B., Pedersen, M.K., Bardram, J., 2015. The dBoard: A digital scrum board for distributed software development. In: Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces. http://dx.doi.org/10.1145/2817721.2817746.

Farid, W.M., 2012. The Normap methodology: Lightweight engineering of non-functional requirements for agile processes. In: Proceedings - Asia-Pacific Software Engineering Conference, APSEC, Vol. 1. pp. 322–325. http://dx.doi.org/10.1109/APSEC.2012.23.

Feldt, R., Magazinius, A., 2010. Validity threats in empirical software engineering research - An initial survey. In: SEKE 2010 - Proceedings of the 22nd International Conference on Software Engineering and Knowledge Engineering. pp. 374–379.

Ferreira, J., Sharp, H., Robinson, H., 2011. User experience design and agile development: Managing cooperation through articulation work. Softw. - Pract. Exp. 41 (9), 963–974. http://dx.doi.org/10.1002/spe.1012.

Fitzgerald, B., Russo, N.L., Stolterman, E., 2002a. Information Systems Development: Methods in Action. McGraw-Hill Education.

Fitzgerald, Brian, Russo, N., Stolterman, E., 2002b. Information Systems Development: Methods in Action.

Forbes Insights, & Scrum Alliance, 2018. THE ELUSIVE AGILE ENTERPRISE: How the Right Leadership Mindset. Workforce and Culture Can Transform Your Organization, https://www.forbes.com/forbes-insights/our-work/elusive-agile-enterprise/.

Frost, N., Nolas, S.M., Brooks-Gordon, B., Esin, C., Holt, A., Mehdizadeh, L., Shinebourne, P., 2014. Pluralism in qualitative research: The impact of different researchers and qualitative approaches on the analysis of qualitative data. Qual. Res. 10 (4), 441–460. http://dx.doi.org/10.1177/1468794110366802.

Gary, K., Enquobahrie, A., Ibanez, L., Cheng, P., Yaniv, Z., Cleary, K., Kokoori, S., Muffih, B., Heidenreich, J., 2011. Agile methods for open source safety-critical software. Softw. - Pract. Exp. 41 (9), 945–962. http://dx.doi.org/10.1002/spe.1075.

Gayer, S., Herrmann, A., Keuler, T., Riebisch, M., Antonino, P.O., 2016. Lightweight traceability for the agile architect. Computer 49 (5), 64–71. http://dx.doi.org/10.1109/MC.2016.150.

Gerster, D., Dremel, C., Brenner, W., Kelker, P., 2020. How enterprises adopt agile forms of organizational design: A multiple-case study. Data Base Adv. Inf. Syst. http://dx.doi.org/10.1145/3380799.3380807.

Gini, C., 1912. Variabilità e mutabilità. In: Memorie Di Metodologica Statistica. https://doi.org/Cited By (since 1996) 41\rExport Date 21 2011.

Goede, R., 2011. Agile data warehousing : The suitability of scrum as development methodology. In: Mccsis 2011.

Goldkuhl, G., Karlsson, F., 2020. Method engineering as design science. J. Assoc. Inf. Syst. 21 (5), 1237–1278. http://dx.doi.org/10.17705/1jais.00636.

Greening, D.R., 2010. Enterprise scrum: Scaling scrum to the executive level. In: Proceedings of the Annual Hawaii International Conference on System Sciences. http://dx.doi.org/10.1109/HICSS.2010.186.

Greening, D.R., 2015. Agile enterprise metrics. In: Proceedings of the Annual Hawaii International Conference on System Sciences. http://dx.doi.org/10.1109/HICSS.2015.597.

Greening, D.R., 2016. Agile base patterns in the agile canon. In: Proceedings of the Annual Hawaii International Conference on System Sciences, 2016-*March*. pp. 5368–5377. http://dx.doi.org/10.1109/HICSS.2016.664.

Griffith, I., Taffahi, H., Izurieta, C., Claudio, D., 2014. A simulation study of practical methods for technical debt management in agile software development. In: Proceedings - Winter Simulation Conference. http://dx.doi.org/10.1109/WSC.2014.7019961.

Haidar, H., Kolp, M., Wautelet, Y., 2017. Agile Product Line Engineering: The AgiFPL Method. http://dx.doi.org/10.5220/0006423902750285.

Hanschke, S., Ernsting, J., Kuchen, H., 2015. Integrating agile software development and enterprise architecture management. In: Proceedings of the Annual Hawaii International Conference on System Sciences. http://dx.doi.org/10.1109/HICSS.2015.492.

Hanslo, R., Vahed, A., Mnkandla, E., 2020. Quantitative analysis of the scrum framework. In: Lecture Notes in Business Information Processing, 376 LNBIP, pp. 82–107. http://dx.doi.org/10.1007/978-3-030-37534-8_5.

Harvie, D.P., Agah, A., 2016. Targeted scrum: Applying mission command to agile software development. IEEE Trans. Softw. Eng. 42 (5), 476–489. http://dx.doi.org/10.1109/TSE.2015.2489654.

Harzing, A.W.K., van der Wal, R., 2008. Google scholar as a new source for citation analysis. Ethics Sci. Environ. Politics 8 (1), 61–73. http://dx.doi.org/10.3354/esep00076.

Hasnain, E., Hall, T., 2009. Introduction to stand-up meetings in agile methods. In: AIP Conference Proceedings. http://dx.doi.org/10.1063/1.3146182.

Hassan, N.R., Mathiassen, L., 2018. Distilling a body of knowledge for information systems development. Inf. Syst. J. 28 (1), 175–226. http://dx.doi.org/10.1111/isj.12126.

Heeager, L.T., Rose, J., 2015. Optimising agile development practices for the maintenance operation: nine heuristics. Empir. Softw. Eng. 20 (6), 1762–1784. http://dx.doi.org/10.1007/s10664-014-9335-7.

Heikkila, V.T., Damian, D., Lassenius, C., Paasivaara, M., 2015a. A mapping study on requirements engineering in agile software development. In: Proceedings - 41st Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2015. pp. 199–207. http://dx.doi.org/10.1109/SEAA.2015.70.

Heikkila, V.T., Paasivaara, M., Lassenius, C., 2013. ScrumBut, but does it matter? A mixed-method study of the planning process of a multi-team scrum organization. In: International Symposium on Empirical Software Engineering and Measurement. pp. 85–94. http://dx.doi.org/10.1109/ESEM.2013.27.

Heikkilä, V.T., Paasivaara, M., Lasssenius, C., Damian, D., Engblom, C., 2017. Managing the requirements flow from strategy to release in large-scale agile development: a case study at Ericsson. Empir. Softw. Eng. 1–45. http://dx.doi.org/10.1007/s10664-016-9491-z.

Heikkilä, V.T., Paasivaara, M., Rautiainen, K., Lassenius, C., Toivola, T., Järvinen, J., 2015b. Operational release planning in large-scale scrum with multiple stakeholders - A longitudinal case study at F-secure corporation. Inf. Softw. Technol. http://dx.doi.org/10.1016/j.infsof.2014.09.005.

Higuchi, M.M., Nakano, D.N., 2017. Agile design: A combined model based on design thinking and agile methodologies for digital games projects. In: Revista de Gestão E Projetos. http://dx.doi.org/10.5585/gep.v8i2.528.

Hoda, R., Murugesan, L.K., 2016. Multi-level agile project management challenges: A self-organizing team perspective. J. Syst. Softw. http://dx.doi.org/10.1016/j.jss.2016.02.049.

Hoda, R., Salleh, N., Grundy, J., Tee, H.M., 2017. Systematic literature reviews in agile software development: A tertiary study. Inf. Softw. Technol. 85, 60–70. http://dx.doi.org/10.1016/j.infsof.2017.01.007.

Hodgetts, P., 2005. Experiences integrating sophisticated user experience design practices into agile processes. In: Proceedings - AGILE Confernce 2005. http://dx.doi.org/10.1109/ADC.2005.24.

Hofman, P., Stenzel, T., Pohley, T., Kircher, M., Bermann, A., 2012. Domain specific feature modeling for software product lines. In: Proceedings of the 16th International Software Product Line Conference on - SPLC '12 -Volume 1. http://dx.doi.org/10.1145/2362536.2362568.

Hossain, E., Babar, M.A., Paik, H., Verner, J., 2009. Risk identification and mitigation processes for using scrum in global software development: A conceptual framework. In: 2009 16th Asia-Pacific Software Engineering Conference. pp. 457–464. http://dx.doi.org/10.1109/APSEC.2009.56.

Hron, M., Obwegeser, N., 2018. Scrum in practice: an overview of scrum adaptations. In: Proceedings of the 51st Hawaii International Conference on System Sciences. pp. 5445–5454, http://pure.au.dk/portal/files/120140100/paper0681.pdf.

Ionica, A., Leba, M., Dovleac, R., 2017. A QFD based model integration in Agile software development. In: Iberian Conference on Information Systems and Technologies, CISTI. http://dx.doi.org/10.23919/CISTI.2017.7975995.

Jakobsen, C.R., Johnson, K.A., 2008. Mature agile with a twist of CMMI. In: Proceedings - Agile 2008 Conference. pp. 212–217. http://dx.doi.org/10.1109/Agile.2008.10.

Jalali, S., Wohlin, C., 2010. Agile practices in global software engineering - A systematic map. In: Proceedings - 5th International Conference on Global Software Engineering, ICGSE 2010. pp. 45–54. http://dx.doi.org/10.1109/ICGSE.2010.14.

James, M., 2010. Scrum Reference Card. New Society.

Kalenda, M., Hyna, P., Rossi, B., 2018. Scaling agile in large organizations: Practices, challenges, and success factors. J. Softw.: Evol. Process 30 (10), e1954. http://dx.doi.org/10.1002/smr.1954.

Kayes, I., Sarker, M., Chakareski, J., 2016. Product backlog rating: a case study on measuring test quality in scrum. Innov. Syst. Softw. Eng. 12 (4), 303–317. http://dx.doi.org/10.1007/s11334-016-0271-0.

Keen, P.G.W., 1980. MIS research: reference disciplines and a cumulative tradition. In: International Conference on Information Systems.

Kettunen, P., 2009. Adopting key lessons from agile manufacturing to agile software product development-a comparative study. Technovation 29 (6–7), 408–422. http://dx.doi.org/10.1016/j.technovation.2008.10.003.

Kikitamara, S., Noviyanti, A.A., 2018. A conceptual model of user experience in scrum practice. In: 2018 10th International Conference on Information Technology and Electrical Engineering. pp. 581–586.

Kitchenham, B.A., 2007. Guidelines for Performing Systematic Literature Reviews in Software Engineering. Keele University.

Kitchenham, B., Pretorius, R., Budgen, D., Brereton, O.P., Turner, M., Niazi, M., Linkman, S., 2010. Systematic literature reviews in software engineering-A tertiary study. Inf. Softw. Technol. 8, 60–70. http://dx.doi.org/10.1016/j.infsof.2010.03.006.

Knudsen, E.I., 1975. Qualitative Data Analysis: A Sourcebook of New Methods (Vol. 51, Issue 2). http://dx.doi.org/10.1016/0300-9629(75)90395-3.

Koc, G., Aydos, M., Tekerek, M., 2019. Evaluation of trustworthy scrum employment for agile software development based on the views of software developers. In: UBMK 2019 - Proceedings, 4th International Conference on Computer Science and Engineering. http://dx.doi.org/10.1109/UBMK.2019.8907213.

Lee, G., Xia, W., 2010. Toward agile: An integrated analysis of quantitative and qualitative field data on software development agility. MIS Q.: Manage. Inf. Syst. 34 (1), 87–114.

Lee, S., Yong, H.-S., 2010. Distributed agile: project management in a global environment. Empir. Softw. Eng. 15 (2), 204–217. http://dx.doi.org/10.1007/s10664-009-9119-7.

Lehnen, J., Schmidt, T.S., Herstatt, C., 2016. Bringing agile project management into lead user projects. Int. J. Prod. Dev. 21 (2–3), 212–232. http://dx.doi.org/10.1504/IJPD.2016.078867.

Li, C., van den Akker, M., Brinkkemper, S., Diepen, G., 2010. An integrated approach for requirement selection and scheduling in software release planning. Requir. Eng. 15 (4), 375–396. http://dx.doi.org/10.1007/s00766-010-0104-x.

Lous, P., Kuhrmann, M., Tell, P., 2017. Is scrum fit for global software engineering? In: Proceedings - 2017 IEEE 12th International Conference on Global Software Engineering, ICGSE 2017. http://dx.doi.org/10.1109/ICGSE.2017.13.

Lyytinen, K., Rose, G.M., 2006. Information system development agility as organizational learning. Eur. J. Inf. Syst. 15 (2), 183–199. http://dx.doi.org/10.1057/palgrave.ejis.3000604.

MacCormack, A., Verganti, R., 2003. Managing the sources of uncertainty: Matching process and context in software development. J. Prod. Innov. Manage. 20, 217–232. http://dx.doi.org/10.1111/1540-5885.2003004.

Madison, J., 2010. Agile and architecture agile–architecture interactions. IEEE Softw. 27 (2), 41–48. http://dx.doi.org/10.1109/MS.2010.27.

Maier, P., Ma, Z., Bloem, R., 2017. Towards a secure SCRUM process for agile web application development. In: Proceedings of the 12th International Conference on Availability, Reliability and Security - ARES '17. pp. 1–8. http://dx.doi.org/10.1145/3098954.3103171.

Masood, Z., Hoda, R., Blincoe, K., 2020. Real world scrum a grounded theory of variations in practice. IEEE Trans. Softw. Eng. http://dx.doi.org/10.1109/tse.2020.3025317.

McMahon, P.E., 2016. CMMI the agile way in constrained and regulated environments. In: CrossTalk.

Mendonça, Letícia Thaís, et al., 2014. A scrum support system integrated to a web project management environment. In: 29th International Conference on Computers and their Applications, CATA 2014.

Miranda, E., Bourque, P., 2010. Agile monitoring using the line of balance. J. Syst. Softw. 83 (7), 1205–1215. http://dx.doi.org/10.1016/j.jss.2010.01.043.

Moe, N.B., Aurum, A., 2008. Understanding decision-making in agile software development: A case-study. In: EUROMICRO 2008 - Proceedings of the 34th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2008. http://dx.doi.org/10.1109/SEAA.2008.55.

Morales Trujillo, M., Oktaba, H., Pino, F.J., Orozco, M.J., 2011. Applying agile and lean practices in a software development project into a CMMI organization. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), http://dx.doi.org/10.1007/978-3-642-21843-9_4.

Øvad, T., Bornoe, N., Larsen, L.B., Stage, J., 2015. Teaching Software Developers to Perform UX Tasks. In: Proceedings of the Annual Meeting of the Australian Special Interest Group for Computer Human Interaction on - OzCHI '15. pp. 397–406. http://dx.doi.org/10.1145/2838739.2838764.

Paasivaara, M., Durasiewicz, S., Lassenius, C., 2009. Using scrum in distributed agile development: a multiple case study. In: Proceedings - 2009 4th IEEE International Conference on Global Software Engineering, ICGSE 2009. http://dx.doi.org/10.1109/ICGSE.2009.27.

Paasivaara, M., Lassenius, C., 2011. Scaling scrum in a large distributed project. In: 2011 International Symposium on Empirical Software Engineering and Measurement. http://dx.doi.org/10.1109/ESEM.2011.49.

Paasivaara, M., Lassenius, C., Heikkilä, V.T., 2012. Inter-team coordination in large-scale globally distributed scrum. In: Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM '12. p. 235. http://dx.doi.org/10.1145/2372251.2372294.

Padmini, K.V.J., Dilum Bandara, H.M.N., Perera, I., 2015. Use of software metrics in agile software development process. In: MERCon 2015 - Moratuwa Engineering Research Conference. http://dx.doi.org/10.1109/MERCon.2015.7112365.

Palomino, M., Dávila, A., Melendez, K., Pessoa, M., 2017. Agile practices adoption in CMMI organizations: A systematic literature review. Adv. Intell. Syst. Comput. 537, 57–67. http://dx.doi.org/10.1007/978-3-319-48523-2_6.

Paré, G., Trudel, M.C., Jaana, M., Kitsiou, S., 2015. Synthesizing information systems knowledge: A typology of literature reviews. Inf. Manage. 52 (2), 183–199. http://dx.doi.org/10.1016/j.im.2014.08.008.

Pastrana, M., Ordóñez, H., Ordóñez, A., Merchan, L., 2017. Requirements elicitation based on inception deck and business processes models in scrum. Commun. Comput. Inf. Sci. 735, 327–339. http://dx.doi.org/10.1007/978-3-319-66562-7_24.

Peixoto, C.S.A., 2009. Human–computer interface expert system for agile methods. In: Proceedings of the International Conference on Information Technology Interfaces, ITI. http://dx.doi.org/10.1109/ITI.2009.5196100.

Rahman, S.S.M.M, Mollah, S.A., Anirban, S., Rahman, M.H., Rahman, M., Hassan, M.M., Sharif, M.H., 2018. OSCRUM: A modified scrum for open source software development. Int. J. Simul.: Syst. Sci. Technol. 19 (3), 20–21. http://dx.doi.org/10.5013/IJSSST.a.19.03.20.

Ramesh, B., Cao, L., Kim, J., Mohan, K., James, T.L., 2017. Conflicts and complements between eastern cultures and agile methods: An empirical investigation. Eur. J. Inf. Syst. http://dx.doi.org/10.1057/s41303-016-0023-0.

Roth, A.P., 2019. Meaning and Method in the Social Sciences: A Case for Methodological Pluralism. Cornell University Press.

Rubart, J., Freykamp, F., 2009. Supporting daily scrum meetings with change structure. In: Proceedings of the 20th ACM Conference on Hypertext and Hypermedia. http://dx.doi.org/10.1145/1557914.1557927.

Santos, N., Fernandes, J.M., Carvalho, M.S., Silva, P.V., Fernandes, F.A., Rebelo, M.P., Barbosa, D., Maia, P., Couto, M., Machado, R.J., 2016a. Using scrum together with UML models: A collaborative university-industry R & D software project. In: International Conference on Computational Science and Its Applications. pp. 480–495. http://dx.doi.org/10.1007/978-3-319-42089-9_34.

Santos, N., Fernandes, J.M., Sameiro Carvalho, M., Silva, P.V., Fernandes, F.A., Rebelo, M.P., Barbosa, D., Maia, P., Couto, M., Machado, R.J., 2016b. Using scrum together with UML models: A collaborative university-industry R & D software project. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer, pp. 480–495. http://dx.doi.org/10.1007/978-3-319-42089-9_34.

Santos, A.R., Sales, A., Fernandes, P., Nichols, M., 2015. Combining challenge-based learning and scrum framework for mobile application development. In: Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education - ITiCSE '15. http://dx.doi.org/10.1145/2729094.2742602.

Schar, B., Jungling, S., Thonssen, B., 2016. Towards an agile requirements engineering process combining HERMES 5 and SCRUM. In: Proceedings - 2015 3rd International Conference on Enterprise Systems, ES 2015. http://dx.doi.org/10.1109/ES.2015.17.

Schlauderer, S., Overhage, S., 2015. Widely used but also highly valued ? Acceptance factors and their perceptions in water-scrum-fall projects. Icis 1–19.

Schwaber, K., 1995. SCRUM development process. In: Business Object Design and Implementation. pp. 117–134. http://dx.doi.org/10.1007/978-1-4471-0947-1_11.
Schwaber, K., 2007. The Enterprise and Scrum. Microsoft Press.
Schwaber, K., Sutherland, J., 2020. The 2020 Scrum Guide. https://www.scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf#zoom=100.
Schwarzer, J., Draheim, S., von Luck, K., Wang, Q., Casaseca, P., Grecos, C., 2016. Ambient surfaces: Interactive displays in the informative workspace of co-located scrum teams. In: Proceedings of the 9th Nordic Conference on Human-Computer Interaction - NordiCHI '16. http://dx.doi.org/10.1016/S0140-6736(15)00947-2.
Scotland, K., Boutin, A., 2008. Integrating scrum with the process framework at Yahoo! Europe. In: Proceedings - Agile 2008 Conference. http://dx.doi.org/10.1109/Agile.2008.22.
Sedeño, J., Schön, E.-M., Torrecilla-Salinas, C., Thomaschewski, J., Escalona, M.J., Mejías, M., 2017. Modelling agile requirements using context-based Persona Stories. In: Proceedings of the 13th International Conference on Web Information Systems and Technologies. pp. 196–203. http://dx.doi.org/10.5220/0006220301960203.
Sharma, R., Wherry, B., 2009. Software development for disney animated feature film production. In: Proceedings - 2009 Agile Conference, AGILE 2009. pp. 410–415. http://dx.doi.org/10.1109/AGILE.2009.60.
Sikamani, T.K., Raj Dharmapal, S., 2016. Using key six sigma and lean metrics on agile scrum methodology for performance improvement. Int. J. Appl. Eng. Res. ISSN 11 (6), 973–4562.
Silva, R.R., Kimura, F.Y., Bernardino, J., De Castro Lima, J., 2017. Custom process to small business. In: Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE. p. 622. http://dx.doi.org/10.18293/SEKE2017-31.
Singh, M., 2008. U-SCRUM: An agile methodology for promoting usability. In: Proceedings - Agile 2008 Conference. pp. 555–560. http://dx.doi.org/10.1109/Agile.2008.33.
Spitzer, R.L., Cohen, J., Fleiss, J.L., Endicott, J., 1967. Quantification of Agreement in Psychiatric Diagnosis: A new approach. Arch. Gen. Psychiatry 17 (1), 83–87.
Stålhane, T., Myklebust, T., Hanssen, G., 2012. The application of safe scrum to IEC 61508 certifiable software. In: 11th International Probabilistic Safety Assessment and Management Conference and the Annual European Safety and Reliability Conference 2012, PSAM11 ESREL 2012. http://dx.doi.org/10.1109/ICSE.2013.6606635.
Stray, V.G., Lindsjorn, Y., Sjoberg, D.I.K., 2013. Obstacles to efficient daily meetings in agile development projects: A case study. In: International Symposium on Empirical Software Engineering and Measurement. http://dx.doi.org/10.1109/ESEM.2013.30.
Streule, T., Miserini, N., Bartlomé, O., Klippel, M., de Soto, B.G., 2016. Implementation of scrum in the construction industry. Procedia Eng. 164, 269–276. http://dx.doi.org/10.1016/j.proeng.2016.11.619.
Sungkur, R.K., Ramasawmy, M., 2014. Knowledge4Scrum, a novel knowledge management tool for agile distributed teams. VINE 44 (3), 394–419. http://dx.doi.org/10.1108/VINE-12-2013-0068.
Sutherland, J., 2005. Future of scrum: Parallel pipelining of sprints in complex projects. In: Proceedings - AGILE Confernce 2005, 2005. pp. 90–99. http://dx.doi.org/10.1109/ADC.2005.28.
Sutherland, J., Jakobsen, C.R., Johnson, K., 2007. Scrum and CMMI level 5: The magic potion for code warriors. In: Proceedings - AGILE 2007. pp. 272–277. http://dx.doi.org/10.1109/AGILE.2007.52.
Suwanya, S., Kurutach, W., 2009. Applying agility framework in small and medium enterprises. In: Communications in Computer and Information Science, 59 CCIS. pp. 102–110. http://dx.doi.org/10.1007/978-3-642-10619-4_13.
Szoke, A., 2010. Optimized feature distribution in distributed agile environments. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), http://dx.doi.org/10.1007/978-3-642-13792-1_7.
Tanveer, M., 2016. Agile for large scale projects - A hybrid approach. In: 2015 National Software Engineering Conference, NSEC 2015. http://dx.doi.org/10.1109/NSEC.2015.7396338.
Tuomikoski, J., Tervonen, I., 2009. Absorbing software testing into the scrum method. In: Lecture Notes in Business Information Processing, pp. 199–215. http://dx.doi.org/10.1007/978-3-642-02152-7_16.
Turk, D., France, R., Rumpe, B., 2005. Assumptions underlying agile software-development processes. J. Database Manage. 57, 52–65. http://dx.doi.org/10.4018/jdm.2005100104.
Vallon, R., da Silva Estácio, B.J., Prikladnicki, R., Grechenig, T., 2018. Systematic literature review on agile practices in global software development. Inf. Softw. Technol. 96, 161–180. http://dx.doi.org/10.1016/j.infsof.2017.12.004.
VersionOne, 2018. 12th Annual State of Agile Report. https://explore.versionone.com/state-of-agile.
VersionOne, 2020. 14th Annual State of Agile Report.
Vlietland, J., Van Solingen, R., Van Vliet, H., 2016. Aligning codependent scrum teams to enable fast business value delivery: A governance framework and set of intervention actions. J. Syst. Softw. 113, 418–429. http://dx.doi.org/10.1016/j.jss.2015.11.010.
Vlietland, J., Van Vliet, H., 2015. Towards a governance framework for chains of scrum teams. Inf. Softw. Technol. 57, 52–65. http://dx.doi.org/10.1016/j.infsof.2014.08.008.
Wagner, S., 2014. Scrum for cyber–physical systems: a process proposal. In: Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering - RCoSE 2014. pp. 51–56. http://dx.doi.org/10.1145/2593812.2593819.
Wang, Y., Bogicevic, I., Wagner, S., 2017. A study of safety documentation in a scrum development process. In: Proceedings of the XP2017 Scientific Workshops. pp. 1–5. http://dx.doi.org/10.1145/3120459.3120482.
Wang, X., Conboy, K., Cawley, O., 2012. Leagile software development: An experience report analysis of the application of lean approaches in agile software development. J. Syst. Softw. 85 (6), 1287–1299. http://dx.doi.org/10.1016/j.jss.2012.01.061.
Wang, Y., Wagner, S., 2016. Toward integrating a system theoretic safety analysis in an agile development process. In: CEUR Workshop Proceedings.
Wautelet, Y., Heng, S., Hintea, D., Kolp, M., Poelmans, S., 2016. Bridging user story sets with the use case model. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), http://dx.doi.org/10.1007/978-3-319-47717-6_11.
Webster, J., Watson, R.T., 2002. Analyzing the past to prepare for the future : Writing a literature review analyzing the past to prepare for the future : Writing A. MIS Q. 26 (2), 13–23, 10.1.1.104.6570.
Wieland, T., 2019. REAT: A Regional economic Analysis Toolbox for R. REGION 6 (3), R1–R57. http://dx.doi.org/10.18335/region.v6i3.267.

Michal Hron is a Ph.D. candidate at the Department of Management at Aarhus BSS in Denmark, where he is affiliated with the Information Systems Research Group. His dissertation focuses on the organizational dynamics of digital innovation and transformation. He holds a MSc. in Business Intelligence from Aarhus BSS.

Nikolaus Obwegeser is a Professor at the Bern University of Applied Sciences and Director of the Institute for Information Systems and Digital Transformation. He is specialized in the areas of digital business transformation, innovation, and information systems. His recent works are published in various academic and practitioner outlets, including MIT Sloan Management Review, Journal of Product Innovation Management, and Technovation. Nikolaus received a Ph.D. degree from the Vienna University of Economics and Business (Austria) and previously held positions at EBS University (Germany), Aarhus University (Denmark), and IMD Business School (Switzerland).