View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by ScholarSpace at University of Hawai'i at Manoa

Proceedings of the 53rd Hawaii International Conference on System Sciences | 2020

"We Use Scrum, but …": Agile Modifications and Project Success

Salma Hassani-Alaoui
HEC Montréal
salma.hassani-alaoui@hec.ca

Ann-Frances Cameron
HEC Montréal
ann-frances.cameron@hec.ca

Tanya Giannelia
HEC Montréal
tanya.giannelia@hec.ca

Abstract

While the Agile-Scrum (scrum) framework has specific guidelines, these guidelines are often adapted by practitioners. This research aims to understand how scrum changes in practice and how these changes impact various aspects of project success. Through interviews with representatives from 11 organizations who use scrum for software development, we found variability in the application of the guidelines, namely, that only a small number of guidelines are systematically followed, and that some guidelines are rarely followed consistently. Examining these method deviations and mapping them to specific dimensions of project success, four patterns emerged. Further, we uncovered practices that are often followed but were not part of the original Scrum guidelines, including how organizations scale scrum projects. These insights into how scrum is used in practice can help industry professionals determine how to best adapt scrum. They also serve as a promising agenda for research on the application of the scrum framework in industry.

1. Introduction

Agile project management was developed in the late 1990s, and popularized with the Agile Manifesto in 2001 [1]. It is described as a way to manage IT projects that increases the likelihood of project success. Agile is a values-based approach to project management that promotes collaboration, transparency, continuous improvement, and adaptation to the client's needs. An agile approach to project management is considered more flexible than traditional project management and therefore claims to better adapt and respond to the client's needs, particularly when these needs are emergent [2]. Because agile is a set of values rather than a detailed approach, several methods have been developed to provide more concrete guidelines to practitioners on how to adopt the values and principles of the agile approach. Scrum is one of the most widely used and well-known agile methods. Since its development in the 1990s, scrum has grown in popularity and as of 2018, was used in 72% of organizations using agile [3].

Scrum has been codified by a series of guidelines found in the Scrum Guide [4]. This guide states that while scrum is easy to learn, it is difficult to master. In this vein, previous work has examined how scrum application in industry differs from these guidelines [5]. To extend this work, this study seeks to examine how specific modifications made to the scrum guidelines impact project success. To answer this question, we interviewed participants in 11 different organizations that use scrum to find out how they apply or do not apply the guidelines in practice and how these modifications impact the success of their projects. We found that while a few guidelines are often followed, others are systematically not followed, and additional practices have emerged in response to organizational and industry needs. Paradoxically, some of the ways the scrum framework is adapted replicate some of the very elements that it claims to change. We conclude with future research directions in the field of how scrum is applied and adapted in practice, and recommendations for practitioners who use the scrum framework.

2. Literature review

2.1 Project success

Although project success is an important theme in project management literature, there is no widely agreed upon definition of success. Traditional success metrics are the "iron triangle" criteria of time, budget and quality [6]. These are often referred to as the foundation of project management [7] and are generally used to evaluate project managers themselves [8]. However, these metrics have been criticized for several reasons. First, failures relative to time and budget are often attributable to poor estimation and not poor project management [9]. Second, the "iron triangle" only represents the success of the management of the project, and does not necessarily reflect the success of the project itself [6, 10, 11].

While there is general consensus of time and cost as being important to project success, the third point,

URI: https://hdl.handle.net/10125/64507
978-0-9981331-3-3
(CC BY-NC-ND 4.0)

quality, has often been modified alternately to "scope" or "requirements," or even include scope as a fourth point [12]. Further success metrics relevant to software development include stakeholder success [13, 14], including satisfaction of the customer [11], senior management/sponsor [10], team [15]; meeting organizational goals [15]; and strategic success/value [10, 11].

2.2 The agile approach

Traditionally, software was developed using sequential methods with phases such as "requirements definition," "solution development," "testing" and "deployment" [2], commonly referred to as a "waterfall" approach to software development. As early as 1970 this approach was criticized, calling for a more iterative approach to software development [16]. While it took several decades to emerge, the agile approach was born partly in response to this criticism.

The Agile Manifesto [1], published in 2001 by a group of experienced software practitioners, brought increasing popularity and interest for agile methodologies and drove unprecedented changes to the field of software development [17, 18]. This approach was developed to address many of the weaknesses of traditional, cascade planning methods [19, 20]. The Agile Manifesto is based on four core values and 12 principles, rather than specific guidelines, which emphasize short development cycles, customer involvement and programmer empowerment [1, 17]. Consequently, Agile has been described as a philosophy rather than a methodology [21, 22].

Agile approaches were purported to make better, faster and cheaper software [18] that responded to the dynamic environments in which IT projects were being developed. They addressed the emerging challenges of the software development industry, characterized by high uncertainty, short development cycles, and the absence of a physical deliverable [23]. Agile approaches have delivered on this promise: Standish group results on project success indicate that across all project sizes, those that use an agile approach are more likely to succeed [24].

2.3 Agile Scrum

Several methods and frameworks have been developed to provide structure to the agile approach of software development, the two most common of which are scrum and extreme programming. Developed in early 1990s, Scrum is described as a lightweight software development method (in contrast to heavyweight traditional methods) [4, 25].

The scrum framework is based on empiricism [4]. According to this philosophy, knowledge can only come from primary, sensory experience [26]. Empiricism in scrum relies on three pillars: transparency, inspection, and adaptation. Transparency requires every aspect of the project to be visible to anyone considered responsible for the outcome. Inspection means that every sprint review is an opportunity to inspect and correct software. The adaptation pillar supports the claim that the process or the material being processed may be changed to adjusted in case of deviations from the acceptable limits. Therefore, Scrum advocates for continuous frequent inspection and adaptation through a disciplined management process [25].

While the pillars of empiricism and the values of scrum are stated in the scrum guide, the guide focuses primarily on describing the operations of scrum [4]. The scrum team consists of the product owner (PO), the scrum master (SM) and the developers. The scrum events, sprint planning, the daily scrum, the sprint review (of the product), and the sprint retrospective (of the process) are used to promote inspection and adaptation. The scrum artifacts are the product backlog (overall product requirements), the sprint backlog (requirements for each sprint) and the increment (the specific deliverable for each sprint).

2.4 Modifications to Scrum

The Scrum guide warns that its artifacts, roles, events and rules are immutable, stating that when changes are made or partial implementation of scrum is conducted, the result is not actually scrum, because "Scrum exists only in its entirety" [4]. However, it is commonly understood in industry that the official practices are not consistently followed [5]. Often referred to as "ScrumBut," practitioners state that they want to respect the guidelines of scrum, *but* for certain reasons they do not feel able [5].

A small stream of research has examined modifications that are commonly made to scrum [5, 27]. Some research has concluded that practitioners will alter scrum to optimize it [27, 28]; to respond to requirements [29], or to adapt to distributed teams or larger projects [29]. Agarwal presented a modified scrum methodology adapted to ongoing software delivery [30]. Sometimes adaptations have been attributed to legacies of previous, non-agile processes [28]. One of the most important, yet most challenging, scrum guidelines to respect is autonomous, self-organizing teams [18]. Stettina and Heijstek [31] studied five dimensions of team dynamics within agile scrum and found that team autonomy was consistently the least respected dimension. Literature has also begun to address adaptations to scrum for large [23] and very large [32] projects. Hobbs & Petit [23]

found that the need for the product owner to balance availability, knowledge of business needs and decision-making authority was particularly challenging, because on large projects these three aspects were often contradictory.

While research has begun to look at adaptations to scrum in general, most prior work focuses on sharing codified transformations of scrum that have been empirically validated to have a positive impact on project success. However, as suggested by Eloranta and colleagues [5] who identified 14 anti-patterns, or deviations from scrum that could potentially be harmful, some modifications to the framework can negatively impact success. The focus of the present paper is understanding how modifications made to the scrum framework either positively or negatively impact the success of the project, whether from a project management or customer satisfaction perspective.

3. Research Method

3.1 Research design

To understand the nuances of modifications made to the Agile-scrum framework in industry and their impact on project success, a qualitative case-study approach was adopted. This method is particularly suited to our research question, as it aims to seek understanding and interpret meaning [33]. Snowball sampling was used to identify representatives from 11 different companies that have been using scrum for software development on at least one project. The unit of analysis was the project itself. Table 1 summarizes the companies in the study. The diversity in the sample allowed us to capture and describe patterns that span wide variability in the cases [34].

Semi-structured interviews [35] were conducted with individuals who were or had recently been involved with a software development project that used scrum. The interview guide was developed using the main principles of scrum. Respondents provided information on the project, the use of scrum, how scrum was adapted, and how adaptations affected project success. To identify potential deviations across the entire scrum framework, 44 specific guidelines were derived from the scrum guide. The list of guidelines was provided to respondents, who were asked to what degree each guideline was respected for the project under study (1 = never; 5 = always). To help understand how and why each guideline was modified or not, participants were asked to elaborate on each of their answers. Interviews concluded with questions about how the modifications impacted project success and team satisfaction with the process. Interviews were conducted in person or over Skype. The interviews lasted 68 minutes on average.

Table 1: Description of cases

Project	Multi teams	Company Size	Company industry	Role of respondent
MEDIC	Y	Med.	Medical Device	Developer
FIN	N	Large	Financial services	Head of proj. mgmt. office (PMO)
SPORT	Y	Large	Sport clothing and equipment	Agile coach
REGIS	Y	Med.	Online registration services	VP
STREAM	Y	Med.	Online streaming services	Product Owner
ERP1	N	Med.	ERP and soft. development	Arch. and Dev.
INTERNET	Y	Large	Internet services	Developer
AUTH	Y	Med.	Security & authentication systems	Scrum master
ERP2	N	Med.	ERP	Developer
GAME	Y	Large	Video Games	Developer
FOOD	N	Large	Food Retail	PM

3.2 Data analysis

Interviews were transcribed (341 pages) and analyzed using NVivo software. Initially, a deductive coding approach [36] was taken, using codes that were based on the findings of the literature review. This step was followed by magnitude coding [36] and the development of matrices to determine similarities and differences between cases. Finally, an inductive, data-driven coding process was used to identify new themes mentioned by respondents. These themes are discussed below.

In addition to analyzing the qualitative interview data, the responses to the 44 guidelines were examined and compared between cases. Five guidelines emerged as being consistently applied and six as having a particularly low rate of adherence. The guidelines with the highest (5) and lowest (6) rates of application are discussed below. Finally, the seven guidelines that had the most varied rate of application were mapped to the two main success metrics, namely project management success and client success to identify patterns. The results are presented below.

4. Results

4.1 Consistently followed scrum practices

Of the five scrum practices that are consistently followed, three of them are related to the mechanics of scrum, and only two are related to the values scrum recommends. These two guidelines described below encourage transparency in the development process, as per the recommendations in the scrum guide [4]. First, 10/11 respondents reported that the team is open about their work and the challenges they are facing however several reported that the teams required training and coaching to achieve this. The PO from STREAM noted: "*It was a work in progress, in the sense that when the product owners arrived Agile was clearly not well understood, the teams were not valued, because it hadn't been put in place 100% as it should have been. So it was really complicated, just to get them on board, because you know, two weeks after they [the product owners, agile coaches and scrum masters] arrived they really made the process more visible."* Similarly, the agile coach from SPORT mentioned: "*Even though they had two years of experience, it was not very well understood. Why? Because they had not been helped out, they had not received training and they had acquired bad habits that were not linked to bad intentions, but that were linked to a misunderstanding of what working in scrum means."*

Second, 10/11 respondents reported conducting sprint reviews. However, only 3/11 fully respected this guideline, and ensured that external stakeholders were present at reviews. This had an impact on the ability to adapt to client requests. The scrum master at AUTH – which does not have the client at their reviews – remarked: *"But the reality is, often the client will look [at the final product] and say 'that's not exactly what I wanted, I'd rather it like this' […] there are lots of advantages of showing the client the product with the developers present."* The three remaining guidelines were related to the mechanics of scrum and are summarized in Table 2 below.

The result that some practices are consistently followed while others are not is not surprising. Even when traditional IT project management methodologies are implemented in one single firm, some of the practices are applied consistently while others are not [37].

.

Table 2: Practices consistently followed

Practice	Quote
(A) One product owner who is a person not a committee The product owner is a role and not a person, and the responsibility can be transferred to another employee, or is added to other responsibilities, and in some cases, is shared (senior-junior relationship).	**Dev. at MEDIC**: "That being said, because of the size of the project, we had proxy product owners [POs] because we could not always speak with the PO because the project was so big. Each team had their proxy PO and if we had any questions about the product, we would see the proxy PO and he would ensure synchronization with the PO."
(B) Scrum master in charge of process and enforcement This was also a role and not an exclusive responsibility taken on by different people in different projects.	**VP at REGIS**: "So in one team a developer plays the role of scrum master, and another team it's the product owner and another team it's me, so it's kind of varies but the role is still here. The project has one scrum master, so it's a role."
(C) Back to back sprints of 4 weeks or less All companies mentioned sprints lasting max four weeks.	**PO at STREAM**: "Yes we had three-week sprints. I have done two weeks, it was a lot of ritual, really. Sometimes you don't have the choice, if your project is only two months long for example. I've even seen one-week sprints. That was too intense. I personally like 3 weeks."

4.2 Rarely followed scrum practices

In addition to the practices that are largely followed, we identified six practices that show low rates of adherence (summarized in Table 3 below). Three practices (G, H, I) are related directly to the mechanics of how scrum is organized. However, three of these practices (D, E, F), are relevant for the core agile.

principle of self-organizing teams, and two (F, I) are key for scrum's transparency [4]. The low rate of adherence to practices related to the autonomy of the team is consistent with Stettina and Heijstek's [31] findings.

4.3 Industry-driven practices

In addition to the scrum guidelines, our respondents described three practices that are not specifically mentioned in the Scrum framework. Two of these practices, grooming and scaling, are specific to Agile projects in large organizations. They have been described in formal large-scale scrum methodologies [38, 39], but in our results, these modifications seem driven by ad-hoc reactions and not by strategy or plan.

The third industry-driven practice, continuous scrum with no set deadline, was specific to the eight companies

Table 3: Practices with a low rate of adherence

Practice	Quotes
(D) Protecting the team from distractions during the Sprint In 8/11 projects, the development team members had other responsibilities outside of the sprint. Many respondents cited these distractions as one of the main sources of delay in delivery.	**Dev. at INTERNET: "**We're always 'disrupted' by something. I think it's both [project and non-project-related disruptions]. I think that in general, the more experience you have, the more people ask things of you. I think that ideally you shouldn't be 100% booked. You should be at 60%, and keep 40% for other things, to help other teams, or other unexpected things."
(E) The development team decides how many items from the product Backlog to include in a Sprint In 7/11 projects, other people were involved in deciding the number of items.	**SM at AUTH:** "It's me and the devs. It's not just the devs, but sooner or later yes, I hope that the team will be autonomous and be able to do it automatically, and be responsible for all of the estimations, yes." **Dev. at MEDIC:** "That was hard. It was hard because the Product Owner always wants to push more, and it's always "let's go guys, we are at 110% capacity but we can do it."
(F) Keeping daily scrums internal to the development team 6/11 teams did not hold daily scrums. 4/11 teams had external participants in the daily scrums.	**VP at REGIS:** "So sometimes developers will stand up, they will do their daily scrum, they will talk about blockers, plans for the next 24 hours but someone from support will be there and [interviewee frowns]." **PMO at FIN:** "No daily scrums…Yeah, there's no way they'll join daily."
(G) Using the backlog as the only source of requirements 5/11 teams did not respect this guideline.	**SM at AUTH**: "as a small start-up, we are also client-facing and need to integrate client requests as well" **PM at FOOD**: "We also, and this is what I don't like about [FOOD], have to have detailed requirement analysis documents, and these are the main source of truth, not the product backlog"
(H) The product owner is the only person responsible for managing the Product Backlog 5/11 teams did not respect this guideline. Reasons included: lack of experience of PO (SPORT), shared with business analyst (FOOD), PMO instead (FIN), responsibility of "closer," a new role (GAME), comments instead of backlog (ERP1).	**PMO at FIN:** "[The product owner] doesn't do it. He has an opinion but is not responsible for managing it. I am the only one who does it. Because people are busy, this is the real world. It's not a book that you're writing, like a book 'Oh you're responsible to do that'." **Dev. at GAME**: "No, I think it's not just him, I think other people around him can modify things [in the backlog]."
(I) Monitoring and sharing progress Only 6/11 teams regularly monitored and shared progress: 3 using a burndown chart, 3 using other means (e.g. JIRA, Gantt chart). Others did not monitor progress at all.	**Dev. from ERP2:** "We talked about it, haha. I don't know, to be honest. I'm not entirely sure if there's a graphical way of tracing the progress. I just know we have a backlog and then we define what should be done in every Sprint, but I'm not sure who's keeping track of the bigger image, you know, of who is looking at how our things are done. I don't know, I am not aware."

in our sample where the product at the heart of the scrum project was part of the company's core business (AUTH, ERP1, ERP2, FIN, GAME, INTERNET, REGIS, STREAM).

4.3.1 Scaled scrum teams

While the Scrum guide describes Agile-scrum practices within a single team, our research uncovered a different reality. To respect the optimal team size while working on a larger project needing more than 9 developers, 7/11 respondents reported some degree of scaling of scrum teams. For two (SPORT, REGIST), the scaling corresponded to the Nexus structure of nested scrum teams as proposed by Schwaber [39]. Our respondents expressed that maintaining the spirit of Scrum while scaled to a larger project required coordination on the part of all team members.

However, not all teams scaled according to industry recommendations. For MEDIC and INTERNET, scrum teams were simply split when the team became too large, creating multiple teams working in parallel on the same project. A developer from MEDIC expressed that this split limited visibility and created silos: *"the challenge is to not always work in a silo. That's what is hardest. It's hard for the work, but also for the quality of the product. Everyone has their strengths and weaknesses, if you can't see what the others are doing nobody improves. You struggle with things that others could solve easily, and you don't learn from what the others do."* At STREAM, AUTH and GAME, teams were formed based on function, with each team responsible for a specific function of the product (e.g. UX, Back-end), and each functional team conducting its own, independent sprints. Appropriately scaling scrum is a challenge evoked by Hobbs & Petit [23] who indicate that research on this issue is "ongoing but is still incomplete" (p. 16).

4.3.2 Grooming and triage

Formally, the Scrum guide prescribes three main events: Sprint Planning, Sprint Review and Sprint Retrospective. Refinement, while only briefly mentioned in the scrum guide, is described by industry as a critical phase in scaled scrum. According to the LeSS framework, it serves to determine the global sprint backlog and assign tasks [38]. The Refinement phase is the initial step in the Nexus Process Flow [39] and serves to decompose the product backlog and remove or minimize dependencies between teams.

Five respondents (STREAM, AUTH, SPORT, INTER, and MEDIC) described a regular refinement meeting, often referred to as "grooming." However, not all organizations implemented refinement in the same way. For example, STREAM, MEDIC and SPORT used this phase as recommended, to break down the complexity of requirements to come to facilitate future planning meetings. The product owner at STREAM considers this meeting the most important of all scrum events. The agile coach from SPORT explained that before his arrival, grooming, estimation and planning were all part of the same meeting, which resulted in long planning meetings that were not very effective. He described why he implemented a separate grooming phase: *"it allows the team to work the needs in advance and reduce the set-up time. [Now] the planning ceremony is much shorter and the team has a better understanding of the needs."*

At AUTH, however, the opposite happened. Estimation, which is usually reserved for the sprint planning event, has been included in the grooming phase, with the goal of making dependencies more transparent. For INTERNET, a "triage" meeting occurred bi-weekly that served to discuss items that are not progressing and understand why they have stalled so they can be moved forward. This process did not address the complexity and dependencies of the stories in the backlog, however.

Grooming, as described in the Nexus guide, is the responsibility of scrum team members [39]. However, at MEDIC, grooming was sometimes done by middle management rather than with the development team. One of the consequences of not involving the development team in grooming is that it limits the team's understanding of the big picture. Team involvement – which reflects the agile recommendation of self-organizing teams [31] and the scrum pillar of transparency [4] – was particularly important for large projects with bigger challenges and greater complexity.

4.3.3 Continuous scrum in software companies

All software-based companies in our sample noted that the project was ongoing, with no specified scope, budget or deadline, similar to the case study described by Agarwal [30]. These companies still reported having time-boxed sprints and their related events. However, because of the lack of formal deadlines, tracking progress – regardless if using a burndown chart or other tool – seemed to become irrelevant. For some, the term "project management" did not apply to their organization's business model. The VP at REGIS shared: *"A project, by definition, has a start date, end date, budget, resources, objective, and a plan. Agility doesn't have a start date, doesn't have allocated resources, and doesn't have an end date, and doesn't have any of this."* The lack of milestones and a clear end-date reportedly created tensions with clients who

were not familiar with the process, whether they were external end-users (REGIS) or internal customers such as the sales department (INTERNET, STREAM). The VP at REGIS shared: "*No! Most customers are not satisfied with the pace at which we are going, and I think it's fair to say that this stems from changing their mind from a project or a more traditional way of doing things towards agility cannot be done overnight without any coaching or extreme curiosity. And a cheer club owner doesn't have the resources, the coach or the interest to go as far as to really understanding agility. So most of them are not happy with this.*"

4.4 Definition of success

Agile is supposed to increase customer satisfaction. For software development companies, success was primarily defined as client satisfaction with the product. Our respondents reported that customers were satisfied with the product; however, in the case of REGIS, this was despite dissatisfaction with the process, particularly the pace of development. The VP noted: *"So they love our product […] because it does everything for them, and nothing else does it, but they don't like the pace at which we're going."* Software development companies, for whom the product was their core business, did not consider budget or schedule as measures of project level success because these were measured at the enterprise level. These companies instead focused on the quality of the product being developed. This tendency is somewhat paradoxical, in that agile software development is thought to improve control over time and budget [18], not to make them irrelevant.

For organizations working on more defined projects, the dimensions of on-time, on-budget and high-quality were used to measure success. Budget and time were closely related. However, the different dimensions of success carried different weights and interpretations for each organization. Completing the project on time emerged as the most important dimension of success across these projects. The three projects that were considered unsuccessful overall by our respondents (ERP1, GAME, and SPORT) did not meet their target schedule. Quality was defined in different ways. AUTH and SPORT identified bug-free code as a marker of quality. For ERP2 quality was defined as a global characteristic of the project. For REGIS, quality meant conforming to the Definition of Done.

4.5 Links to success

One of the goals of this research was to determine how deviations from the scrum guidelines impact project success. To this end, the seven guidelines that were identified as having medium rates of adherence were mapped to respondents' evaluation of project management success and customer satisfaction (the two metrics exhibiting the greatest variability) to identify any emerging patterns. These guidelines are indicated in Appendix 1. From this analysis we identified three specific recommended guidelines that appear the most closely related to project management success, and one that seemed associated with customer satisfaction. Not respecting these specific guidelines seemed to impact these elements of success. The guidelines are described below.

First, our analysis suggests there may be a positive relationship between the development team's control over how many items from the backlog assigned to a Sprint and project quality. More specifically, it suggests that meeting the target quality of the project may be affected by the team's power to independently plan each sprint. The Scrum guide recommends that development teams autonomously decide the work estimates for the sprint. However, sometimes authoritative Product Owners/Project Managers/Scrum Masters, and hierarchical work practices, interfere with this autonomy. This lack of control can result in reduced commitment from the team and inaccurate estimates which may translate into poor performance and a defective schedule [5]. Four of our respondents reported that the development team did not have complete control over the backlog. For three of these projects, the respondents also mentioned issues relating to quality. The fourth respondent, a developer at MEDIC, described the impact that a lack of control could have on the quality of the project. He touched on the tension between his development team and his Product Owner over the number of elements to include in a Sprint. He explained that the Product Owner represents the customer/client side in the project and pushes for features to be developed faster. While this could motivate the team to be more effective and productive, it can also contribute to technical debt and employee stress, which in turns impacts the quality of the increment. The developer explained that while the pressure from the Product Owner helped them finish the essential parts that made up an increment, a lot of work surrounding that increment such as specifications adjustment and testing was not done appropriately resulting in a build-up of technical debt. Similarly, Codabux and Williams suggested there were negative impacts on project outcomes when technical debt was improperly managed [40].

Second, a positive relationship between the Product Owner's sole responsibility for the Product Backlog seemed to relate to project management success. While the Scrum Guide allows for input from the entire Scrum team about the Product Backlog's items, it does

emphasize the responsibility of the product owner for the Product Backlog [4]. The interviews revealed that the Product Owners on only six projects took on full responsibility for the product backlog. Projects where the product owner was the only one responsible for the product backlog usually succeeded in meeting their scope, budget and schedule targets. When product owners were not solely responsible, the project often did not meet its targets. For example, the Agile coach from SPORT noted that because the Product Owner was a novice, other team members helped out. This impacted the organization of the backlog and ultimately the work on the project: *"There was a bit of everything. People who verbally said "hey, we can do this, we can do that" which is logical, and can even be a good thing…but the problem is that it was not organized, which meant that in the end we had a backlog that was all over the place, because the product owner did not do his job of filtering, homogenizing everything."* This disorganization caused delays, over-spending and quality issues with the project: *"The team removed things [items from the backlog] because they lacked maturity, because they lacked knowledge, and by removing these things, it prevented the team from […] targeting quality, it prevented the team from working at a sustainable pace, it prevented lots of things."* As the Agile coach explained, he was hired to correct these issues and ensure success of future projects. When the product owner cannot properly organize and plan the backlog, the team may not know which items the client has prioritized. This can create complications in late stages of the project, which may explain the pattern observed [5].

Third, we noted a positive relationship between conducting retrospective meetings with the objective of inspection and improvement, and both project management and customer success. The goal of sprint retrospectives, unlike sprint reviews, is to improve how the team functions [4]. Only 6 of the 11 respondents confirmed having a sprint retrospective meeting that serves to inspect the team and addresses work process improvement. The product owner at STREAM explained that *"the key to the retro[spective] is really that [problems] are solved, because otherwise, there's no point in having a retro[spective]."* Two projects (FOOD and SPORT) did hold retrospective meetings, but these were treated as an opportunity to air grievances and were not focused on improving work processes. ERP1 did not conduct retrospectives because they were seen as a waste of time by senior management. Skipping retrospectives or holding ones that are not centered on improvement may result in higher levels of frustration within the team. Moreover, it may also hinder the team's ability to communicate, reflect, and progress in their work. This can lead to team stagnation and lost efficiency and productivity, which may explain the pattern identified [5]. This is demonstrated in the reflection of the Agile coach from SPORT: *"To me, what explained why the team didn't move forward is because they didn't inspect what it was doing. They didn't reflect on what they did, and they didn't improve […..] Communication, reflection and improvement were not done. They just delivered and it wasn't even going well […] and it's the retrospectives that let you work on those three [things]."* Sprint retrospectives provide an opportunity for organizational learning, defined as "an aggregation of local action and reflection cycles" [41] (p. 129). Research has found that when these cycles break down in production teams, the product could suffer from "inadequate quality and cost improvement, potentially harming customer satisfaction (p. 143)." Thus, explaining the possible link between project retrospectives and project success.

5. Discussion

Consistent with [5], none of the organizations in our sample follow all the guidelines of Scrum all the time. Three of the practices that are not consistently followed are related to team organization (letting the development team decide how many items from the product Backlog to include in a Sprint; Protecting the team from distractions during the Sprint; Keeping Daily scrums internal to the development team). This result is consistent with [18] who stated that maintaining autonomous, self-organizing teams is the most challenging aspect of scrum to achieve, and with [31] who noted that team autonomy was the least respected aspect of scrum. The one team-related practice that was consistently followed was being open about their work and problems, however, as was noted above in Section 4.5, sometimes this was interpreted as an invitation to air grievances, rather than as an opportunity to improve.

Agile scrum is built on empiricism, a theory which emphasizes the importance of experience, and of making decisions based on what is known [26]. Scrum stands on three pillars that reflect empiricism: transparency, inspection and adaptation. From our research, however, not all modifications made to the scrum process reflect these pillars. First, ad-hoc scaling of scrum teams, or creating teams based on product function, can limit the visibility of certain aspects of the project to members of the development teams, going against the pillar of transparency of the entire process for any who are responsible for the outcome, including the development team members. Second, not involving external stakeholders in reviews limits the possibility for inspection and subsequent adaptation of the product, and as such contradicts these pillars. Third,

retrospectives focused on process improvement serve to inspect and adapt the team's work; by not focusing retrospectives on improvement, these pillars are again contradicted and can impact project outcomes.

Consistent with previous research [23], our results demonstrated that scrum is being scaled in response to industry needs. However, rather than strategically implemented, in our sample this appears ad-hoc and unplanned, sometimes hindering visibility and collaboration between teams working on the same project, contradicting the pillars and the values of Scrum.

This research unveiled that it takes more than simply adopting the prescribed scrum guidelines to benefit from the added value for which the scrum framework was initially developed. Ensuring that a project has all the roles, practices, and artifacts of a scrum framework does not guarantee optimal adoption or the benefits it claims to bring. Rather, it is knowledge and understanding of the objectives and principles behind framework components that contribute to achieving agility in software development. For example, the inconsistent interpretation and application of grooming, a relatively new addition to the scrum framework, provides further evidence for the claim that scrum, while easy to learn, is difficult to master. Teams that frequently respected the pillars and values of scrum, even when modifications were made, appeared to achieve better outcomes. The results of this research therefore invite practitioners to reflect on how to best educate their teams on what the roles, practices, and artifacts mean and how they contribute to having the most advantageous development environment.

6. Conclusion

While this study presents important findings, it does have some limitations. First, only one person was interviewed per organization, and the role of the person interviewed was not the same for each organization. It is possible that responses were biased by incomplete knowledge of the project and organizational practices, or by their perspective based on their role within the project. Despite this, as scrum favors transparent communication and close collaboration, we feel that respondents, regardless of their role, were able to comment satisfactorily on the project. A second limitation is the sample size. However, the goal of the qualitative approach used was to uncover practices and identify potential patterns, not demonstrate statistical significance and we feel our sample was large and diverse enough to identify emerging patterns. We plan to conduct additional interviews to collect additional data in the future. A third potential limitation is that the conclusions are solely based on interview responses. Further work on this project will include re-engaging with the participants, and collecting artifacts and documentation to further support the findings.

Despite the limitations, this research has important implications both for future research and for practice. For research, this project uncovered some important insights into how agile-scrum practices diverge from formal guidelines. While it has been long understood that Scrum is often adapted by industry, this project provides a list of specific areas where adherence to the guidelines is not uniform, providing a framework for future research in this area. Second, the understanding and implication of industry-driven practices such as scaling scrum and formalizing grooming appears inconsistent. Further research could measure the impact of a strategic vs. ad-hoc use of these practices.

Of interest to practitioners, we have identified three practices that, when applied, seem to be related to project success. This would suggest that these are some of the most important practices in scrum to which to adhere. Second, while the scrum framework has been adapted to large and very large projects, many of the organizations we interviewed have not adopted a formal scaled scrum approach, but instead have used an ad-hoc approach to expanding scrum. One of the dangers of this ad-hoc approach is that it can result in creating silos and limiting visibility between sub-teams, potentially impacting project success. Practitioners should take note that if scrum is easy to learn but difficult to master, scaled scrum is even more difficult to master, and conducting it improperly could have consequences for project success.

7. References

1. Beck, K., et al., *Manifesto for agile software development.* 2001.
2. Leau, Y.B., et al. *Software development life cycle AGILE vs traditional approaches*. in *International Conference on Information and Network Technology*. 2012.
3. VersionOne. *13th Annual State of Agile Survey*. 2018; Available from: https://www.stateofagile.com/#ufh-i-521251909-13th-annual-state-of-agile-report/473508.
4. Schwaber, K. and J. Sutherland, *The scrum guide-the definitive guide to scrum: The rules of the game*. 2017.
5. Eloranta, V.-P., K. Koskimies, and T. Mikkonen, *Exploring ScrumBut—An empirical study of Scrum anti-patterns.* Information and Software Technology, 2016. **74**: p. 194-203.
6. Ika, L.A., *Project Success as a Topic in Project Management Journals.* Project Management Journal, 2009. **40**(4): p. 6-19.
7. Atkinson, R., *Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria.* International journal of project management, 1999. **17**(6): p. 337-342.

8. Wateridge, J., *How can IS/IT projects be measured for success?* International journal of project management, 1998. **16**(1): p. 59-63.
9. Nelson, R.R. and M.G. Morris, *IT Project Estimation: Contemporary Practices and Management Guidelines.* MIS Quarterly Executive, 2014. **13**(1).
10. Jugdev, K. and R. Müller, *A retrospective look at our evolving understanding of project success.* Project management journal, 2005. **36**(4): p. 19-31.
11. Shenhar, A.J., et al., *Project success: a multidimensional strategic concept.* Long range planning, 2001. **34**(6): p. 699-725.
12. Pollack, J., J. Helm, and D. Adler, *What is the Iron Triangle, and how has it changed?* International Journal of Managing Projects in Business, 2018. **11**(2): p. 527-547.
13. Davis, K., *Different stakeholder groups and their perceptions of project success.* International journal of project management, 2014. **32**(2): p. 189-201.
14. Davis, K., *Reconciling the Views of Project Success: A Multiple Stakeholder Model.* Project Management Journal, 2018. **49**(5): p. 38-47.
15. Belassi, W. and O.I. Tukel, *A new framework for determining critical success/failure factors in projects.* International journal of project management, 1996. **14**(3): p. 141-151.
16. Royce, W.W. *Managing the development of large software systems: concepts and techniques*. in *Proceedings of the 9th international conference on Software Engineering*. 1987. IEEE Computer Society Press.
17. Vial, G. and S. Rivard, *Understanding Agility in ISD Projects*, in *ICIS*. 2015: Fort Worth, Texas.
18. Dybå, T. and T. Dingsøyr, *Empirical studies of agile software development: A systematic review.* Information and software technology, 2008. **50**(9-10): p. 833-859.
19. Serrador, P. and J.K. Pinto, *Does Agile work?—A quantitative analysis of agile project success.* International Journal of Project Management, 2015. **33**(5): p. 1040-1051.
20. Thummadi, B.V., et al. *Enacted Software Development Routines Based on Waterfall and Agile Software Methods: Socio-Technical Event Sequence Study*. 2011. Berlin, Heidelberg: Springer Berlin Heidelberg.
21. Palmquist, M.S., et al., *Parallel worlds: Agile and waterfall differences and similarities*. 2013, CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.
22. Lapham, M.A., et al., *Considerations for using agile in DoD acquisition*. 2010, CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.
23. Hobbs, B. and Y. Petit, *Agile methods on large projects in large organizations.* Project Management Journal, 2017. **48**(3): p. 3-19.
24. Hastie, S. and S. Wojewoda, *Standish group 2015 chaos report-q&a with Jennifer Lynch.* Retrieved, 2015. **1**(15): p. 2016.
25. Ashraf, S. and S. Aftab, *Pragmatic Evaluation of IScrum & Scrum.* International Journal of Modern Education and Computer Science, 2018. **10**(1): p. 24.
26. Curd, M. and S. Psillos, *The Routledge Companion to Philosophy of Science*. 2010, London: Routledge.
27. Ashraf, S. and S. Aftab, *Latest Transformations in Scrum: A State of the Art Review.* International Journal of Modern Education and Computer Science, 2017. **9**(7): p. 12.
28. Diebold, P. and M. Dahlem. *Agile practices in practice: a mapping study*. in *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*. 2014. ACM.
29. Hron, M. and N. Obwegeser. *Scrum in practice: an overview of Scrum adaptations*. in *Proceedings of the 51st Hawaii International Conference on System Sciences*. 2018.
30. Agarwal, P. *Continuous SCRUM: agile management of SAAS products*. in *Proceedings of the 4th India Software Engineering Conference*. 2011. ACM.
31. Stettina, C.J. and W. Heijstek. *Five agile factors: Helping self-management to self-reflect*. in *European Conference on Software Process Improvement*. 2011. Springer.
32. Dingsøyr, T., N.B. Moe, and E.A. Seim, *Coordinating Knowledge Work in Multiteam Programs: Findings From a Large-Scale Agile Development Program.* Project Management Journal, 2018. **49**(6): p. 64-77.
33. Silverman, D., *Interpreting qualitative data*. 2015: Sage.
34. Robinson, O.C., *Sampling in interview-based qualitative research: A theoretical and practical guide.* Qualitative research in psychology, 2014. **11**(1): p. 25-41.
35. Miles, M.B., A.M. Huberman, and J. Saldana, *Qualitative data analysis*. 2013: Sage.
36. Saldaña, J., *The coding manual for qualitative researchers*. 2015: Sage.
37. Terlizzi, M.A., F. de Souza Meirelles, and H.R.O.C. de Moraes, *Barriers to the use of an IT Project Management Methodology in a large financial institution.* International Journal of Project Management, 2016. **34**(3): p. 467-479.
38. *Why LeSS?* 2018; Available from: https://less.works/less/framework/why-less.html.
39. Schwaber, K. *The Nexus (TM) Guide*. 2018.
40. Codabux, Z. and B. Williams. *Managing technical debt: An industrial case study*. in *Proceedings of the 4th International Workshop on Managing Technical Debt*. 2013. IEEE Press.
41. Edmondson, A.C., *The local and variegated nature of learning in organizations: A group-level perspective.* Organization Science, 2002. **13**(2): p. 128-146.

Appendix

Appendix 1 and 2 are available here:
http://chairemulticommunication.hec.ca/publications/