Journal of Engineering, Project, and Production Management
2021, 11(3), 255-262

Agile Practices and Impacts on Project Success

Rebecca Sandstø[1] and Cornelia Reme-Ness[2]
[1]Researcher, Department of Mechanical and Industrial Engineering, Norwegian University of Science and Technology, S. P. Andersens vei 5, 7491 Trondheim, Norway. E-mail: r.sandstoe@gmail.com (corresponding author).
[2]Researcher, Department of Mechanical and Industrial Engineering, Norwegian University of Science and Technology, S. P. Andersens vei 5, 7491 Trondheim, Norway. E-mail: corneliaremeness@gmail.com

Integration of Project and Production Management
Received February 19, 2021; revised April 20, 2021; accepted April 24, 2021
Available online May 4, 2021

Abstract: The creation of Agile frameworks signified the development of practices specifically aimed at delivering projects - in an unpredictable world - on time, within budget and to the required quality. The purpose of this study is to present the potential effects of common Agile practices on conditions for project success. Two literature reviews are conducted. The first literature review identifies commonly reported Agile practices, while the second literature review focuses on these Agile practices' reported effects. The two literature reviews identify twelve commonly reported Agile practices and their reported effects on six conditions for project success. Some Agile practices are shown to be more common than others. An abundance of identified terms used for Agile practices complicates the review of the existing empirical studies and the establishment of a common research agenda. Furthermore, the research shows that most of the examined Agile practices have both positive and negative effects on the conditions for project success. Only a few of the commonly reported Agile practices are shown to have a solely positive effect. The study identifies variations in the amount of available research on the effects of the various Agile practices. For research, this study identifies Agile practices and effects that need further attention. For practice, this study shows that Agile practices should be implemented with the desired effects, and the organizational and contextual conditions, in mind.

Keywords: Agile methodology, Agile practices, project success.

DOI 10.2478/jeppm-2021-0024

1. Introduction

Many software projects have struggled to achieve project success, and software projects of all kinds worldwide have failed (Charette, 2005). Software projects have struggled to deliver the right quality within schedule and under budget (Jørgensen and Moløkken-Østvold, 2006; Keil et al., 2000; Nasir and Sahibuddin, 2011). The struggle to achieve success in software projects led to the creation of the Agile frameworks. The Manifesto for Agile Software Development (hereby called "Agile Manifesto") emerged as a reaction to the Waterfall framework to create a new basis for frameworks that would better cope with the increasing uncertainties (Dybå and Dingsøyr, 2008). The Agile Manifesto (Beck et al., 2001) established four values (hereby called "Agile values") and twelve principles (hereby called "Agile principles"). Agile frameworks are frameworks that align with and are based on the Agile Manifesto (Beck et al., 2001), and they were supposed to provide solutions through their lightweight processes (Martin, 2012, p. 8).

It has been established that Agile frameworks have a positive impact on project success (Papatheocharous and Andreou, 2014; Serrador and Pinto, 2015). However, it is common for projects to implement a set of individual Agile practices instead of a complete Agile framework with all its Agile practices (Sidky et al., 2007; Cao and Ramesh, 2008). There is an abundance of Agile practices utilized in projects all around the world (VersionOne, 2020), affecting the project in their own unique ways (Fitzgerald et al., 2006). It is therefore important to understand how each Agile practice can impact conditions for project success. Several studies have addressed the positive and negative effects on, e.g., communication and motivation - caused by the various Agile practices (Pikkarainen et al., 2008; McHugh et al., 2011). However, the studies tend to focus either on a limited number of Agile practices or on a limited number of conditions for project success. To the best of the authors' knowledge, there is no existing framework for the expected impact from each Agile practice on a variety of conditions for project success. This study will contribute to the body of knowledge by integrating literature on the reported effects of common Agile practices and consequently establishing a framework for the Agile practitioners who wish to implement a selected set of Agile practices.

Two research questions, RQ1 and RQ2, respectively, will be answered through two separate literature reviews:

What Agile practices are commonly reported?

What are the reported effects of commonly reported Agile practices?

2. Method

2.1. Design of Literature Reviews

To answer the research questions, there were conducted narrative literature reviews based on the description by Green et al. (2006) and Ferrari (2015).

Since the publication of the Agile Manifesto (Beck et al., 2001) in 2001, Agile practices have abundantly been described in research (Dingsøyr et al., 2012). Since most of scientific research relevant for Agile frameworks occurred after the articulation of the Agile Manifesto (Beck et al., 2001), as mentioned by Dingsøyr et al. (2012), all literature from before 2001 was discarded. To find the most relevant literature, a combination of keyword searches and reference snowballing was used. The snowballing was conducted as described by Sayers (2008) and using the Google Scholar search engine. All searches were conducted in English.

The nature of the two investigated research questions was slightly different. For RQ1, the task was to identify the most relevant and acknowledged Agile practices. For RQ2, there was a need for a set of sources specifically investigating reported effects. Consequently, the two research questions demanded different literature searches.

2.2. RQ1 Literature Search

One search criterion utilized for the RQ1 literature search concerned the number of citations. To ensure identification of the most common Agile practices, mainstream references were deemed the most relevant. In this context, "mainstream" refers to references that present theories regarded as conventional and widely accepted. Mainstream references will often have many citations, as they complement the findings of many other references. Thus, to increase the probability of the Agile practices being widely reported, publications with a high number of citations were preferred. Both peer-reviewed articles and textbooks by credited authors were included in the final list of references.

The Norwegian University of Science and Technology (NTNU) Library (Oria) was the chosen database for the keyword search. The keyword search was conducted with various search phrases, where the most relevant proved to be the following:

• Agile AND (practice* OR method* OR technique*)

• "Agile software development" AND (practice* OR method* OR technique*)

The choice of relevant literature was based on its relevance to RQ1 (evaluated through title and later full text), and the search criteria explained above. The keyword search resulted in adding more recent references with an investigatory perspective on Agile practices. The RQ1 literature search was considered finished when adding references no longer resulted in entirely new Agile practices, but rather in confirming already identified ones.

2.3. RQ2 Literature Search

The objective of the literature review for RQ2 was to identify reported effects of the commonly reported Agile practices. The literature search was performed as a narrative literature search, but with elements from the systematic review methodology - e.g., the systematic exclusion of sources. To evaluate the relevance and credibility of the sources for the literature review, the following search criteria were used:

1. Literature topic: Empirical studies, relevant for RQ2
2. Literature type: Peer-reviewed articles
3. Publishing year: 2001 - 2020
4. Citations: Minimum ten citations

The literature search was conducted according to the search criteria defined above, and RQ2 and its scope. Web of Science was the chosen database for the search phrases, as it encompasses journals of high impact factor (NTNU, 2020). The initial search was conducted using the following phrase:

• Agile AND (practice* OR method* OR technique*) AND (effect* OR impact*) AND empirical*

Additionally, specific searches targeting all the Agile practices under investigation were conducted. To further narrow down the search hits, articles not related to the Agile Manifesto (Beck et al., 2001) and articles with less than ten citations were eliminated. The title and abstract of the remaining articles were screened for relevance to RQ2, before snowballing was performed using Google Scholar.

3. Findings and Discussion

3.1. Common Agile Practices

The commonly reported Agile practices resulting from the literature review will be written in "italic" – e.g., *sprints* – and each will be provided a unique ID – e.g., AP1. This is to highlight the commonly reported Agile practices in the text and to enable their traceability. Furthermore, it serves to distinguish between the commonly reported Agile practices and the terms for Agile practices as used by the references when answering RQ2 in Sub-Section 3.2.

The list of commonly reported Agile practices was created in four steps as follows:

1. Identifying Agile practices
2. Clustering similar Agile practices
3. Filtering out rarely mentioned Agile practices
4. Sorting Agile practices by categories

To make the initial list, the relevant references were reviewed, and all reported Agile practices were noted. Many of the Agile practices in the initial list were similar. In some cases, the Agile practices were completely identical but listed under different names - e.g., "stand-up meetings" and "daily progress tracking meetings". To simplify the list - and thus simplify future investigation and discussion - similar Agile practices were clustered together. The clustered Agile practices were given a common name. Widely accepted and intuitively understandable names were chosen as the common names. Furthermore, the clustered Agile practices mentioned by less than five references were excluded to ensure that only commonly reported Agile practices were included. Finally, the Agile practices were sorted by category and listed with their common names. The final list of twelve commonly reported Agile practices could be seen in Table 1. The unused names for the Agile practices were moved to a separate column, along with the names appearing while answering RQ2. Out

of the identified twelve Agile practices, the most commonly reported Agile practices are *sprints*, *continuous code integration*, *incremental releases*, and *value-prioritized requirements*.

Table 1. Commonly reported Agile practices

Agile Practice	Additional Names and Similar Agile Practices
AP1: Co-located team	Open office landscape/space, co-located office space, open working environment, face-to-face conversations at the workspace, open workspaces
AP2: Customer involvement	Customer team member, client driven iterations, end-user involvement, on-site customer, customer collaboration
AP3: Self-organized team	Team autonomy
AP4: Continuous progress visualization	Tracking iteration progress, continuous visualization, task board, storyboard, status board, informative workspace
AP5: Incremental releases	Frequent releases, review of deliveries, early releases, agile feedback, sprint review, demonstration, customer feedback, iteration reviews, collective / review meetings
AP6: Minimal documentation	Agile documentation
AP7: Retrospectives	Retrospective meetings, iteration retrospective, reflection workshop, post-iteration workshops, sprint retrospective, collective retrospective meetings
AP8: Value-prioritized requirements	Planning game, simplicity in design, specification of requirements, prioritization of requirements, user stories, client-driven iterations, iteration backlog, product backlog, sprint backlog, metaphor, agile specification, gathering and clarifying requirements
AP9: Continuous code integration	Frequent integration, continuous builds, automated builds, continuous integration
AP10: Pair programming	No additional names or similar Agile practices identified.
AP11: Sprints	Short iterative cycles, iteration planning (meetings), constant planning, adaptive planning, agile sprints
AP12: Stand-up meetings	Daily progress tracking, daily (team) meetings, daily stand-up (meeting), scrum meeting

3.2. Effects of Agile Practices

3.2.1. Communication

The findings by Pikkarainen et al. (2008) might suggest that the internal communication – i.e., the communication within the team – is positively impacted by *stand-up meetings* (AP12), *co-located team* (AP1), *continuous progress visualization* (AP4), *sprints* (AP11), *retrospectives* (AP7), *pair programming* (AP10), and *continuous code integration* (AP9). This positive effect from the *co-located team* on internal communication was also supported by Mishra et al. (2012), who claimed that an open working environment was important to facilitate internal communication.

In contrast, *co-located team*, *sprints*, and *pair programming* could also have a negative effect on the internal communication, along with *value-prioritized requirements* (AP8). Pikkarainen et al. (2008) observed that the open office space could make it more difficult to focus. They also claimed that the product backlog and *sprints* were negatively associated with internal communication by decreasing the project's long-term visibility. This might indicate that *value-prioritized requirements* and *sprints* have a negative effect on internal communication. Furthermore, *pair programming* could result in challenges and frustration among team members when utilized daily (Pikkarainen et al., 2008).

Sprints can also be understood to have a positive effect on the external communication – i.e., the communication between the team and the various stakeholders – along with *incremental releases* and *value-prioritized requirements*. Pikkarainen et al. (2008) claimed that iteration planning improved the external communication by enabling an enhanced short-term focus, suggesting a positive effect from *sprints* on external communication. They further argued that iteration reviews, product backlog and sprint backlogs established new communication methods between the team and the stakeholders, thus ensuring that the product would be in accordance with the requirements. This might indicate that *incremental releases* and *value-prioritized requirements* also can improve the external communication. The findings by Hummel et al. (2015) might indicate a support for the positive impact from *incremental releases* on the external communication, as *incremental releases* may be understood to enable frequent direct communication between the team members and the customer.

Co-located team, *stand-up meetings*, *sprints*, *pair programming*, *incremental releases*, and *retrospectives* could have a positive effect on direct communication – defined as "synchronous" and "spoken" communication – according to Hummel et al. (2015). They argued that a co-located office space, daily *stand-up meetings*, iteration planning meetings, *pair programming*, sprint review and sprint retrospective meetings were the most effective Agile practices for direct communication.

Face-to-face was perceived as the most efficient communication method (Hummel et al., 2015). Another suggested positive effect from *co-located team* was its ability to reduce external disturbances as there was a physical separation from other projects. *Stand-up meetings* were found to enable extensive direct communication –

both during and after the meetings – and *retrospectives* to facilitate the improvement of the direct communication (Hummel et al., 2015).

Retrospectives, *value-prioritized requirements*, and *continuous progress visualization* could be important enablers for indirect communication – defined as "asynchronous" and "written" communication – according to Hummel et al. (2015) and Mishra et al. (2012). Although not as evident, indirect communication could also be important for project success (Hummel et al., 2015; Pikkarainen et al., 2008). Hummel et al. (2015) noted product backlog and agreements from *retrospectives* as valuable documents. As documentation is encompassed by the definition of indirect communication, *value-prioritized requirements* and *retrospectives* could be understood to have a positive effect on indirect communication. Mishra et al. (2012) claimed that status boards could facilitate non-verbal communication – also considered a form of indirect communication – indicating a positive effect from *continuous progress visualization*.

3.2.2. Motivation

Motivation is connected to autonomy (Pelletier et al., 2001), and a *self-organized team* (AP3) leads to autonomous teams. Thus, it is natural to believe that teams implementing *self-organized teams* will experience increased motivation. However, the literature is divided on this matter, and it is difficult to make a direct connection. Pelletier et al. (2001) argued that self-determined motivation was affected positively when the social context encouraged individual autonomy. Furthermore, Tripp et al. (2016) found that Agile practices impacted perceived job autonomy, and thereby motivation. On the other hand, Moe et al. (2010) noted that group-level autonomy could conflict with individual autonomy. They argued that self-organizing groups could inflict stricter control on the individual members than team leaders in traditional frameworks, e.g., the Waterfall framework. Langfred (2007) supported this claim by arguing that self-organized teams could tend to reduce individual autonomy – especially when responding to conflicts. Therefore, it cannot be affirmed if implementing *self-organized team* provides more motivated individuals or if the self-organized group ends up decreasing individual autonomy and thereby motivation.

The findings of McHugh et al. (2011) indicate that implementing *sprints* (AP11), *stand-up meetings* (AP12), and *retrospectives* (AP7) could contribute to stronger motivation. They reported that the increased motivation was caused by increasing motivation-contributors such as a sense of belonging and employee participation. McHugh et al. (2011) also reported negative effects on motivation from implementing *sprints*, *stand-up meetings*, and *retrospectives*. They reported that the three Agile practices caused de-motivation due to increased meeting length and frequency, and challenges related to implementing the Agile practices on complex tasks. Additionally, McHugh et al. (2011) found that *stand-up meetings* and *retrospectives* also contributed to decreased motivation by increasing stress.

3.2.3. Transparency

The findings of McHugh et al. (2012) can indicate how *sprints* (AP11), *stand-up meetings* (AP12) and *retrospectives* (AP7) affect transparency in Agile teams. McHugh et al. (2012) found that *sprints* improved visibility of requirements and time estimates. *Stand-up meetings* led to transparency of the daily progression in the project. *Retrospectives* provided transparency of the progression in the project, leading to faster clarifications concerning delays and immediate improvements in the work processes that caused the delays. Additionally, developers reported that the process of openly reporting time estimates raised the level of mutual responsibility the team members felt towards each other in reaching the time and quality goals.

3.2.4. Trust

Sprints (AP11), *stand-up meetings* (AP12) and *retrospectives* (AP7) may increase trust in Agile teams (McHugh et al., 2012). McHugh et al. (2012) argued that these three Agile practices increased trust indirectly through, e.g., communication and knowledge-sharing. Their findings are somewhat contrary to the ones reported by Piccoli and Ives (2003), who claimed behavioral control mechanisms (activities such as explicit work assignments and specifications of rules and procedures, found in *sprints* and *stand-up meetings*) led to a decrease in trust.

McHugh et al. (2012) also found that the Agile practices sometimes affected trust in negative ways. Specifically, individuals would feel a higher team pressure if they did not deliver on time or with the pre-agreed quality. The open sharing of time estimates – created to increase trust – was observed to possibly have the opposite effect (McHugh et al., 2012). *Sprints*, *stand-up meetings*, and *retrospectives* led to quicker identification of tasks that had not been delivered on time, and the trust consequently decreased. The challenges related to implementing behavioral control, found in *sprints* and *stand-up meetings*, were supported by Piccoli and Ives (2003).

3.2.5. Knowledge-sharing

Co-located team (AP1), *continuous progress visualization* (AP4), *stand-up meetings* (AP12), *sprints* (AP11), *incremental releases* (AP5), *retrospectives* (AP7), and *customer involvement* (AP2) could cause increased knowledge-sharing. Santos et al. (2015) claimed that insufficient knowledge-sharing could cause rework and that sufficient knowledge-sharing could result in time savings by enabling reuse and improvements of previous work.

The study by Santos et al. (2015) might suggest that *co-located team* increases knowledge-sharing through identifying team members' knowledge, enabling feedback and visibility, and improving the work environment. Open workspaces could enable the creation of social areas at work – such as areas for playing games and consumption – that facilitated informal conversations, as observed by Santos et al. (2015). These informal conversations could enable knowledge-sharing because people with useful knowledge of the discussed topics were able to enter conversations and share what they knew (Santos et al., 2015). The findings by Santos et al. (2015) on the *co-located team* might also suggest an indirect connection between face-to-face communication and knowledge-sharing through problem-solving. This was supported by Pikkarainen et al. (2008) and Hummel et al. (2015), who argued that the co-located and open office space accelerated problem-solving.

The findings by Santos et al. (2015) might indicate that *continuous progress visualization* increases knowledge-sharing. The informative workspaces made progress and

status of the various projects visible to the team members, thus directly enabling feedback and visibility.

Furthermore, the study by Santos et al. (2015) might suggest that *stand-up meetings*, *sprints*, *incremental releases*, and *retrospectives* increase knowledge-sharing through enabling feedback and visibility, and through improving the work environment. Again, their findings imply an indirect connection between these Agile practices and knowledge-sharing through problem-solving, which Pikkarainen et al. (2008) supported for *stand-up meetings*. Parts of the findings by Santos et al. (2015) were supported by McHugh et al. (2012), who reported that *sprints*, *stand-up meetings*, and *retrospectives* increased knowledge-sharing.

Dingsøyr and Lassenius (2016) and Karlstrom and Runeson (2005) found that *customer involvement* increased the feedback rate from the customer and gave the supplier a better understanding of the customer value. Their findings were indirectly backed by Hoda et al. (2011), who found that insufficient *customer involvement* caused problems in prioritizing requirements. These findings might indicate that *customer involvement* increases knowledge-sharing between customer and supplier.

3.2.6. Agility

Agility can be defined as the team's ability to respond to the customer's changing requirements in an extensive and efficient manner (Lee and Xia, 2010; Recker et al., 2017). Response extensiveness and response efficiency are two dimensions of software project agility. Response extensiveness is related to the amount, magnitude, and variety of the response, while response efficiency is related to the required resources to respond (Lee and Xia, 2010).

Pair programming (AP10) could cause increased response extensiveness, according to Recker et al. (2017). They reasoned that Agile development practices – such as *pair programming* – would improve the project outcome. On the other hand, *self-organized team* (AP3) and *stand-up meetings* (AP12) could cause a decreased response extensiveness, according to Lee and Xia (2010) and Recker et al. (2017) respectively. Lee and Xia (2010) reported a negative effect from team autonomy on response extensiveness, suggesting a negative effect from *self-organized team* (AP3). They discovered that self-organized teams at times intentionally decreased their response extensiveness to increase the probability of meeting the project objectives.

Self-organized team could cause an increased response efficiency, according to Lee and Xia (2010), who reported a positive effect of team autonomy on response efficiency. The increased response efficiency could come from the empowered decision-making of self-organized teams (Lee and Xia, 2010). *Stand-up meetings* and *pair programming* have no indicated significant effect on response efficiency, as Recker et al. (2017) reported that neither of the two Agile practices was significantly related to response efficiency. However, Recker et al. (2017) argued that the absence of customers at the *stand-up meetings* could increase the response efficiency, as less time would be needed to discuss the requirements at the meetings.

3.3. Key Trends in Findings

The findings are summarized in Table 2, where "+", "-" and "N" indicate positive, negative, and no effect, respectively. Each symbol signifies one reported effect. Cells with conflicting symbols indicate that varying effects have been reported and that the Agile practice might need more attention in both research and practice.

Table 2. Effects of Agile practices on conditions for project success

	Agility		Communication				Motivation	Transparency	Trust	Knowledge-Sharing
	Extensiveness	Efficiency	Internal	External	Direct	Indirect				
AP1: Co-located team			-/++		+					+++
AP2: Customer involvement										+++
AP3: Self-organized team	-	+					-/+			
AP4: Continuous progress visualization			+			+				+
AP5: Incremental releases				+	+					+
AP6: Minimal documentation										
AP7: Retrospectives			+		+	+	-/+	+	+	+
AP8: Value-prioritized requirements			-	+		+				
AP9: Continuous code integration			+							
AP10: Pair programming	+	N	-/+		+					
AP11: Sprints			-/+	+	+		-/+	+	-/+	+
AP12: Stand-up meetings	--	N	+		+		-/+	+	-/+	++

There are noteworthy variations in the number of references that report each Agile practice. Even considering the qualitative nature of this study, this evidence suggests that the most common Agile practices are: *sprints*, *continuous code integration*, *incremental releases*, and *value-prioritized requirements*.

About half of the findings marked in Table 2 indicate an exclusively positive effect of an Agile practice on a project success condition – e.g., the effect of *retrospectives* on indirect communication. Some effects are agreed upon by several references – e.g., Lee and Xia (2010) and Recker et al. (2017) agreeing on *stand-up meetings* affecting response extensiveness (agility) negatively.

A few of the effects are exclusively negative – e.g., the effect from *self-organized team* on response extensiveness (agility). Only two of the project success conditions are only positively affected; transparency and knowledge-sharing. Furthermore, communication is shown to experience numerous positive effects of various Agile practices. External communication, along with direct and indirect communication, are only positively affected. No project success condition is affected only negatively. Most of the project success conditions are affected in both positive and negative ways.

Almost half of the Agile practices have various – i.e., conflicting – reported effects on the same project success condition, as either "+/-", "N/+", or "N/-/+". An example is the conflicting effects of *retrospectives* on motivation. The Agile practice with the highest number of conflicting effects is *sprints*. These findings might indicate that practitioners should consider the advantages and disadvantages of the Agile practices carefully before implementing them. The Agile frameworks and their practices are not perfect, and most of the Agile practices will imply both positive and negative consequences.

Table 2 also reveals that most of the project success conditions experience different effects from the different Agile practices – e.g., the conflicting effects from *retrospectives* and *value-prioritized requirements* on internal communication. Furthermore, for each project success condition, the number of reported effects and degree of positive effects vary.

Table 2 shows noteworthy variations in the number of reported effects for each Agile practice. E.g., no references matching the search criteria were found to report effects on *minimal documentation*. In contrast, *sprints* have been found to affect five project success conditions. This variation might indicate which of the Agile practices have widespread effects and which have a more limited influence on project success. The variation in the number of reported effects for each Agile practice could also potentially be explained by varying amounts of existing literature and deficiencies in the RQ2 literature search.

Out of the total 120 cells in Table 2, 77 cells are empty. This could indicate that numerous effects caused by Agile practices are yet to be investigated.

Although no relevant sources were found investigating the effect of *minimal documentation* directly, there exist reasons to believe that *minimal documentation* has a positive impact on project success conditions (Arisholm et al., 2006; Lethbridge et al., 2003; Garousi et al., 2015).

4. Conclusion

The first research question was defined as follows: *What Agile practices are commonly reported?*

The study resulted in the identification of twelve commonly reported Agile practices. These resulted in being: *co-located team*, *customer involvement*, *self-organized team*, *continuous progress visualization*, *incremental releases*, *minimal documentation*, *retrospectives, value-prioritized requirements*, *continuous code integration*, *pair programming*, *sprints*, and *stand-up meetings*. Although this study was qualitative in nature, evidence suggests that *sprints*, *continuous code integration*, *incremental releases*, and *value-prioritized requirements* constitute the most common Agile practices.

Furthermore, the findings suggest that establishing a generalized set of Agile practices is difficult since many authors use different terms which ultimately mean the same thing. An example is *value-prioritized requirements*, which has been implemented in various ways through, e.g., backlog, simplicity in design, user stories and metaphors. A backlog is not the same as a metaphor, but they both serve the same goal of delivering the most valuable features to the customer. Nevertheless, the effects of a backlog can potentially be greater than that of a metaphor. Thus, grouping these Agile practices together – even though they serve the same purpose – can give misleading impressions of their impacts.

The second research question was defined as follows: *What are the reported effects of commonly reported Agile practices?*

It has been observed that the literature provides some evidence regarding the effects from the commonly reported Agile practices. Some of the most visible effects of Agile practices concern improvement of communication. The Agile practices create forums for interaction, such as *stand-up meetings* and *retrospectives*, which have a substantial impact on communication. This is as well evident for knowledge-sharing and transparency, which are impacted exclusively positively. The increased communication and knowledge-sharing are in line with the vision of the Agile Manifesto's focus on interaction and responsiveness. Other impacts are vaguer – especially the high rate of conflicting reported effects is striking. Nearly half of the reported effects are conflicting. Organizational or contextual conditions could be parts of the cause for these conflicting reported effects. At the same time, Agile practices – such as *stand-up meetings* – have been shown to have both positive and negative impacts. This may indicate that the implementation of Agile practices should reflect upon the project context and presume both positive and negative impacts.

This study has contributed to the discussion of Agile practices by creating an overview of their effects. All commonly reported Agile practices identified have shown to have at least some positive effect. However, some Agile practices have been identified to have more positive effects than others and to affect a wider range of project success conditions – e.g., *sprints* and *retrospectives*. It has also been shown that some Agile practices might need more attention before one can conclude if implementing them is worthwhile – e.g., *self-organized team*.

For practice, this study demonstrates that there is an abundance of Agile practices affecting project success. Most of the reviewed Agile practices are found to have a

range of effects on project success conditions, both positive and negative. Based on the findings in this study, the decision on what Agile practices to implement should be tailored for each project's requirements. Every Agile practice should be carefully assessed to ensure that the effects of the chosen Agile practice correspond to the desired and necessary effects in each project. Practitioners are encouraged to be aware of the reported effects of each implemented Agile practice.

Furthermore, this study demonstrates that an Agile practice can have various effects on project success conditions. E.g., *pair programming* has a reported positive or negative effect on internal communication. This implies that the effects of an Agile practice on project success conditions are sensitive to various contextual elements, such as team size, culture and project complexity. An implementation of an Agile practice solely based on its reported effects is therefore inadvisable. Practitioners are encouraged to be aware of the fact that project circumstances, context, implementation and the combination of Agile practices influence the project success conditions.

5. Future Research and Limitations

For future research, this study demonstrates that there is an abundance of terms for Agile practices. Many Agile practices overlap and reflect nuances of the same Agile practice – e.g., *co-located team*, "open office landscape," and "co-located office space." Other Agile practices are ultimately the same but presented with slightly different names – e.g., *self-organized team* and "team autonomy." This makes it difficult to review existing empirical studies and to establish a common research agenda for the effects of the various Agile practices. It is therefore recommended to use the Agile practices' names cautiously. While empirical studies on Agile practices' effects are limited, one should not fall for the temptation of clustering "too different" Agile practices together and generalize their effect.

Furthermore, this review demonstrates that the effects of some Agile practices are more examined than others. The effects of, e.g., *stand-up meetings* are addressed by many empirical studies, whereas there is a lack of empirical studies on the effects of, e.g., *minimal documentation*. The abundance of empty cells in Table 2 might indicate that the vast majority of Agile practices' effects are not yet investigated. It is recommended to do more research on the effects of Agile practices in order to unveil whether they bring benefit or harm to Agile projects.

Agile practices are implemented differently, with differing duration, under varying circumstances, and in various combinations. These variables have not been taken into account and are probable to have influenced the reported effects addressed in this study. Furthermore, some Agile practices have been studied more than others. This makes the reported effects of some Agile practices (e.g., *stand-up meetings*) to be more certain than others (e.g., *minimal documentation*).

This study is also limited by the deficiency in the data collection and data analysis methods used in the creation of this study. Firstly, only a limited number of sources have been reviewed. As time was limited, the study does not address all reported effects from the commonly reported Agile practices. Therefore, this study should not be used to compare or prioritize the commonly reported Agile practices, but rather to give an indication. Secondly, there are shortcomings in the processing of Agile practices to create the list of commonly reported Agile practices to answer RQ2. Separating Agile practices and Agile principles can be complicated and almost completely dependent on the authors´ interpretation. The clustering of Agile practices might have resulted in the list of commonly reported Agile practices spanning too wide and causing inconsistencies in the reported effects – e.g., *sprints* encompassing "short iterative cycles" and "iteration planning". The clustering process could also have been biased by the authors' perception of the Agile practices' purpose, as well as the authors' knowledge of the Agile practices and their ability to cluster names and practices correctly.

Acknowledgments

The authors express their gratitude to Assoc. Prof. Bassam Hussein at the Norwegian University of Science and Technology for the support during the research. The authors are also grateful for the support from Torbjørn Kågen at Metier OEC.

References

Arisholm, E., Briand, L. C., Hove, S. E., and Labiche, Y. (2006). The impact of UML documentation on software maintenance: an experimental evaluation. *IEEE Transactions on Software Engineering*, 32, 365-81.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., and Jeffries. R. (2001). Manifesto for agile software development. Retrieved from https://agilemanifesto.org/ on November 22, 2020.

Cao, L. and Ramesh, B. (2008). Agile requirements engineering practices: An empirical study. *IEEE Software*, 25, 60-67.

Charette, R. N. (2005). Why software fails [software failure]. *IEEE Spectrum*, 42, 42-49.

Langfred, C. W. (2007). The Downside of Self-Management: A Longitudinal Study of the Effects of Conflict on Trust, Autonomy, and Task Interdependence in Self-Managing Teams. *Academy of Management Journal*, 50, 885-900.

Dingsøyr, T. and Lassenius, C. (2016). Emerging themes in agile software development: Introduction to the special section on continuous value delivery. *Information and software technology*, 77, 56-60.

Dingsøyr, T., Nerur, S., Balijepally, V. G., and Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. *The Journal of systems and software*, 85, 1213-21.

Dybå, T. and Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and software technology*, 50, 833-59.

Ferrari, R. (2015). Writing narrative style literature reviews. *Medical writing (Leeds)*, 24, 230-35.

Fitzgerald, B., Hartnett, G., and Conboy, K. (2006). Customising agile methods to software practices at Intel Shannon. *European Journal of Information Systems*, 15, 200-13.

Garousi, G., Garousi-Yusifoğlu, V., Ruhe, G., Zhi, J., Moussavi, M., and Smith, B. (2015). Usage and usefulness of technical software documentation: An industrial case study. *Information and software technology*, 57, 664-82.

Green, B. N., Johnson, C. D., and Adams, A. (2006). Writing narrative literature reviews for peer-reviewed journals: secrets of the trade. *Journal of chiropractic medicine*, 5, 101-17.

Hoda, R., Noble, J., and Marshall, S. (2011). The impact of inadequate customer collaboration on self-organizing Agile teams. *Information and software technology*, 53, 521-34.

Hummel, M., Rosenkranz, C., and Holten, R. (2015). The role of social agile practices for direct and indirect communication in information systems development teams. *Communications of the Association for Information Systems*, 36, 15.

Jørgensen, M. And Moløkken-Østvold, K. (2006). How large are software cost overruns? A review of the 1994 CHAOS report. *Information and software technology*, 48, 297-301.

Karlstrom, D. and Runeson, P. (2005). Combining agile methods with stage-gate project management. *IEEE Software*, 22, 43-49.

Keil, M., Mann, J., and Rai, A. (2000). Why software projects escalate: An empirical analysis and test of four theoretical models. *MIS quarterly*, 631-64.

Lee, G., Xia, W. (2010). Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility. *MIS quarterly*, 34, 87-114.

Lethbridge, T. C., Singer, J., and Forward, A. (2003). How software engineers use documentation: the state of the practice. *IEEE Software*, 20, 35-39.

Martin, R. C. (2012). *Agile Software Development: Principles, Patterns, and Practices*. Pearson Education, Inc.

McHugh, O., Conboy, K., and Lang, M. (2012). Agile Practices: The Impact on Trust in Software Project Teams. *IEEE software*, 29, 71-76.

McHugh, O., Conboy, K., and Lang, M. (2011). Using agile practices to influence motivation within it project teams.

Mishra, D., Mishra, A., and Ostrovska, S. (2012). Impact of physical ambiance on communication, collaboration and coordination in agile software development: An empirical evaluation. *Information and software technology*, 54, 1067-78.

Moe, N. B., Dingsøyr, T., and Dybå, T. (2010). A teamwork model for understanding an agile team: A case study of a Scrum project. *Information and software technology*, 52, 480-91.

Nasir, M. H. N. and Sahibuddin, S. (2011). Critical success factors for software projects: A comparative study. *Scientific research and essays*, 6, 2174-86.

NTNU. (2020). Finding Sources. Retrieved from https://innsida.ntnu.no/wiki/-/wiki/English/finding+sources on December 7, 2020.

Papatheocharous, E. and Andreou, A. S. (2014). Empirical evidence and state of practice of software agile teams. *Journal of software: evolution and process*, 26, 855-66.

Pelletier, L. G., Fortier, M. S., Vallerand, R. J., and Briere, N. M. (2001). Associations among perceived autonomy support, forms of self-regulation, and persistence: A prospective study. *Motivation and Emotion*, 25, 279-306.

Piccoli, G. and Ives, B. (2003). Trust and the Unintended Effects of Behavior Control in Virtual Teams. *MIS Quarterly*, 27, 365-95.

Pikkarainen, M., Haikara, J., Salo, O., Abrahamsson, P., and Still, J. (2008). The impact of agile practices on communication in software development. *Empirical Software Engineering*, 13, 303-37.

Recker, J., Holten, R., Hummel, M., and Rosenkranz, C. (2017). How agile practices impact customer responsiveness and development success: a field study. *Project Management Journal*, 48, 99-121.

Santos, V., Goldman, A., and De Souza, C. R. B. (2015). Fostering effective inter-team knowledge sharing in agile software development. *Empirical Software Engineering*, 20, 1006-51.

Sayers, A. (2008). Tips and tricks in performing a systematic review chapter 4. *Br J Gen Pract*, 58, 136-36.

Serrador, P. and Pinto, J. K. (2015). Does Agile work? - A quantitative analysis of agile project success. *International Journal of Project Management*, 33, 1040-51.

Sidky, A., Arthur, J., and Bohner, S. (2007). A disciplined approach to adopting agile practices: the agile adoption framework. *Innovations in Systems and Software Engineering*, 3, 203-16.

Tripp, J. F., Riemenschneider, C., and Thatcher, J. B. (2016). Job Satisfaction in Agile Development Teams: Agile Development as Work Redesign. *Journal of the Association for Information Systems*, 17, 267-307.

VersionOne. (2020). 14th Annual State of Agile Report. Retrieved from https://stateofagile.com/#ufh-i-615706098-14th-annual-state-of-agile-report/7027494 on November 28, 2020.

Rebecca Sandstø has experience as Project Manager for the team *Revolve NTNU*, competing in the world's biggest engineering student competition, *Formula Student*. In this position, she gained hands-on experience with tasks such as following up on technical progression, controlling finances, increasing motivation, and leader development. She has experience in the commercial car business as well, where she has developed price forecasting models. Furthermore, she has studied how strategic alliances affect individual motivation in projects in the oil and gas industry. Her research interests are within Agile practices in projects.

Cornelia Reme-Ness has experience as Project Manager for *Revolve NTNU*, developing one electric and one autonomous race car from scratch to compete in *Formula Student*. The responsibilities comprised of project planning and execution, project performance, organizational development, team building and motivation, financial management, HR and HSE, marketing and sponsor relations, and development of mechanical, electrical and software systems. Furthermore, she has experience as a Project Planner for *Aker Solutions* in the oil and gas industry and as Project Controller for *Multiconsult* in both the construction industry and oil and gas industry. Her research interests are within Agile practices in projects.