Antecedents and Consequences of Time Pressure in Scrum Projects: Insights from a Qualitative Study

Stefan Linßen
University of Cologne
stefan_linssen@web.de

Dirk Basten
University of Cologne
basten@wiso-uni-koeln.de

Janek Richter
University of Cologne
janek.richter@wiso.uni-koeln.de

Abstract

Time pressure represents one of the most critical factors associated with software development project failure. However, its role in previous research is limited to traditional development approaches and is also inconclusive, suggesting both negative and positive effects, while others argue for dependence on the level of time pressure. In the context of agile software development projects, particularly Scrum, we seek to analyze the role of time pressure in software development projects and aim to develop a better understanding of its antecedents and consequences. We apply an exploratory, interview-based research approach grounded in data and complemented by thematic analysis. Our findings support a differentiated picture of time pressure, revealing that not all antecedents equally lead to negative consequences and that effects are dependent not only on intensity but also on temporality of time pressure. Understanding antecedents and consequences serves organizations to advantageously manage time pressure.

1. Introduction

One of the most critical factors that can lead to failure of software development projects is time pressure [1], which is commonly referred to as "scarcity of time available to complete a task, or set of tasks, relative to the demands of the task(s) at hand" [2, p. 1315]. However, previous research is inconclusive concerning the role of time pressure because studies reveal both positive and negative impacts on software development project outcomes [2, 3]. Additionally, other researchers suggest that the type of effect depends on the level of time pressure. While previous research has a prevailing focus on traditional development approaches [2, 4, 5, 6, 7] and analyzes time pressure's effect on selected project outcomes such as effectiveness [5] and efficiency [6], there is evidence that time pressure is a constraint to agile software development projects [1]. We aim to develop a better understanding of time pressure's antecedents and consequences in this context. We thus pose the following research question: *What are the antecedents and consequences of time pressure in agile software development projects?*

Due to a lack of previous insights, we use a qualitative research approach that is grounded in data. By means of semi-structured interviews with professionals, we shed light on the phenomenon of time pressure in agile software development. With this study, we contribute to the literature stream on time pressure in software development projects by providing in-depth insights about the antecedents and consequences of time pressure and by focusing on the specific context of agile management of software development projects. Furthermore, we contribute to a deeper understanding of agile software development projects by revealing that non-adherence to rules causes time pressure in diverse ways.

This paper is structured as follows. In Section 2, we describe the underpinnings of agile software development – Scrum in particular. We also reflect the previous research on time pressure in software development projects. In Section 3, we explain our research approach, that is, the collection of data through semi-structured interviews and data analysis by means of thematic analysis. We use Section 4 to present our results. We then discuss contributions, limitations, and implications of our work in Section 5. Our study ends with a short conclusion in Section 6.

2. Time Pressure in Agile Software Development Projects

2.1. Scrum as Agile Management Approach

Software development is the process of creating software that realizes and fulfills customer expectations [8]. In general, agile software

URI: http://hdl.handle.net/10125/50497
ISBN: 978-0-9981331-1-9

development can be described as the combination of creative teamwork and intense focus on effectiveness and maneuverability [9]. Agile development is "both a philosophy and an umbrella term for a collection of methods or approaches that share certain common characteristics" [10, p. 9] rather than a single method. Two of the most popular approaches are Extreme Programming (XP) and Scrum [11].

XP is an agile approach; it focuses on a continuous development process that converges customer requirements step by step. There is no masterplan before starting the development. Developers can respond to changing customer requirements even late in the project lifecycle. XP delivers software that customers need [12]. Scrum is an iterative and incremental project management framework to address complex problems and to optimize the value of the product to be delivered [13]. Transparency, inspection, and adaption are the three pillars of this framework. Transparency means that the process is visible for all project stakeholders. Frequent inspections are important to determine problems in early project stages. If problems are detected, the adaption of scope and process is needed to avoid further problems. While "XP and Scrum may each be incomplete in their coverage of the overall development process, they are very complementary in that XP provides good support for the more technical and coding aspects of development while Scrum provides a very good framework for project planning and tracking" [11, p. 210].

It can be presumed that issues related to time pressure are primarily managed in planning stages in a project management context. Given our interest in time pressure in agile software development projects and Scrum's nature as a *management* framework, we focus on Scrum rather than XP, which can be considered a *development* method.

Scrum consists of roles, events, artifacts, and rules. The roles in a Scrum Team are Product Owner, Scrum Master, and the Development Team. The Product Owner is responsible for managing the requirements included in the Product Backlog. The Product Owner is the contact person for the client who determines the content of the Product Backlog. This role also includes explaining the Product Backlog items and goals of the project to the Development Team. The Scrum Master, amongst others, has to ensure that Scrum is understood and enacted. For instance, the Scrum Master coaches the Development Team to self-organize and to create high-value products. The Scrum Master also collaborates with other Scrum Masters in the organization to improve the Scrum process. The Development Team is responsible for implementing and developing the product during Sprints. It is self-organized and manages the work on its own.

Scrum events are time-boxed events that help to inspect and adapt. The Sprint, which includes all Scrum events, lasts no longer than one month.

Scrum artifacts include the Product Backlog and the Sprint Backlog. The sum of Backlog Items turned into a potentially shippable product during one Sprint is an increment. The Product Backlog is a list of requirements, functions, and enhancements of the product. This list is derived by the Product Owner who explains the items in the Product Backlog to the Development Team. The Sprint Backlog includes the Spring Goal and the items of the Product Backlog that are selected for the specific Sprint.

2.2. Time Pressure in Software Projects

Managers of software development projects have to consider the three indices of time, budget, and quality, which are known as the 'iron triangle' [14] and 'triple constraints' [15]. These indices depend on each other (e.g., if the budget is reduced, this reduction has a direct influence on the quality of the software or the time). Related effort estimates are typically unrealistically low [16], and overruns of schedule and budget are common phenomena in software projects [17, 18, 19], which lead to time pressure.

Several studies have addressed the role of time pressure in software projects [e.g., 2, 4, 6, 20]. However, respective findings are inconclusive, which can be attributed to the lack of a generally accepted definition of the concept [21]. Studies find both a negative and a positive impact of time pressure on work performance [2]. Since software developers under time pressure simply work faster rather than better [22], time pressure can lead to software quality reductions [6]. Contrarily, quality of decisions has been found to increase when time-dependent incentives are given [23]. Additionally, research proposes this relation to depend on the level of time pressure [2, 3]. While moderate levels of time pressure are likely to be beneficial, high levels of time pressure are counterproductive. In general, many aspects of time pressure's impact on software projects remain unexplained [24], and the relationship between time pressure and software projects is more complex than it might seem at first glance [5].

The research concerning agile software development is scarce. With one exception, previous research has focused on traditional development approaches. The study by Malgonde et al. [1] is the only one that concentrates on agile software development. However, the authors analyze projects using XP as an agile development method and provide

preliminary results only. Considering the increasing importance of Scrum as an agile project management framework for software projects and given the limited insights from previous research, time has come to analyze time pressure in such projects.

3. Research Approach

3.1. Data Collection

For our exploratory research, we rely on semi-structured, qualitative interviews. Such interviews are among the most important data collection approaches and used in qualitative research of all kinds [25]. Interviews in this study have been conducted with employees of an internal IT service provider from Germany. This service provider primarily works for a company in the retail sector. The main tasks of the IT service provider are planning, developing, configuring, and operating efficient systems and applications. The service provider employs a workforce of about 1,000 professionals. As shown in Figure 1, the interview process is organized by four parts: (1) introduction to interview, (2) question about personal background, (3) evaluation of the role of time pressure in agile software development projects, and (4) conclusion of interview and next steps.

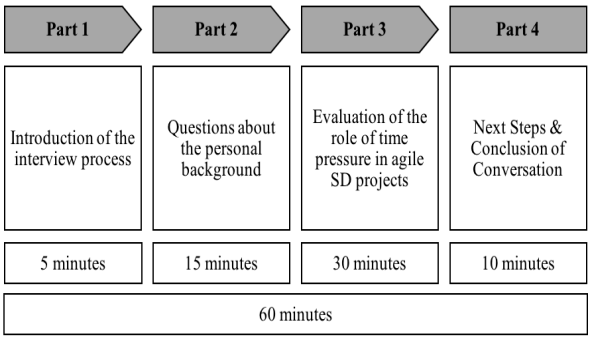

Figure 1: Overview of the Interview Process

With the consent of the participants, we audio-recorded and transcribed the interviews. During the transcription process, we anonymized the interviews to ensure that no link between the interviewee and transcript exists. The transcription of the nine interviews resulted in more than 70 pages of transcribed data.

Two different large-scale and long-term IT projects (code-named Alpha and Beta) form the foundation for our interviews with members of Scrum Teams. Both projects have adopted Scrum with minor adaptions, which we address in Section 4. Table 1 provides an overview of both projects, while Table 2 shows an overview of all interview participants.

Table 1: Project Characteristics

Characteristics	Alpha	Beta
# of employees	11	6
# External members	5	2
Project duration	3 years	1.5 years
Project budget	1,000,000 EUR	693,000 EUR
Client	external	internal

Project Alpha has the goal to support the online business of the partner company. This project introduces an e-commerce solution which connects the customer with the partner company. Project Alpha has contextual factors; for example, the management had announced a deadline before the project started. The next factor is the completely agile working client. This factor is an advantage because the client understands the working process of the IT service provider. The last contextual factor is that the project development depends on external data. If there is a problem with the delivery of external data, the development of the e-commerce solution will be delayed. In total, project Alpha has been considered successful by the interviewees.

The second project is called Beta, and it is a continuous running software project. It is a live system and needs to be improved continuously. The fact that this system needs to be available for the end customer during business hours makes it business-critical. The system always needs updates, seasonal supplements, and support. It is necessary that the live system has no downtime or only limited downtime while these tasks are realized. Several contextual factors influence project Beta. For example, the seasonal product developments have deadlines. A deadline may be a product supplement that has to be developed by Christmas. If there is a delay and the product is finished after Christmas, this part of the project has failed. Project Beta has several external dependencies. Graphics or external data are only two factors that have to be delivered by external suppliers. The last contextual factor is the cold and frozen zone. The IT service provider has a cold and frozen zone, which means deployments can just go live with a special permit by the top management. The cold and frozen zone are a security measure to ensure a working system without problems during critical business phases, for example, Christmas or Easter. In total, this project is also described as successful by the interviewees.

The interviewees have been chosen for their role in the project. The study tries to cover every aspect of the project from different points of view, so every Scrum role has to be part of the interview. In total, five

Product Owners, one Scrum Master, and three members of the Development Team have been interviewed. The interviewees all have different backgrounds and are both internal and external to the company.

Table 2: Overview of Interviewee Demographics

Pseudonym	Project	Role	Working Experience (years)	# Scrum Projects
Andrew	Alpha	PO	7	2
Charlie		PO	>20	2
David		PO	1,5	1
Harry		Dev	5	3
Edward	Alpha & Beta	SM	10	2
Benjamin	Beta	PO	1.5	1
George		PO	15	1
Frederick		Dev	8.5	1
Isaac		Dev	11	1

SM – Scrum Master
PO – Product Owner
Dev – Member of the Development Team

3.2. Data Analysis

We analyzed and coded the data following Flick [26] and used the software tool MAXQDA. In our coding approach, we relied on thematic analysis, which is a method to identify, analyze and report patterns within data [27]. According to Rubin and Rubin [28, p. 226], thematic coding is used to "discover themes and concepts embedded throughout your interviews". The themes depend on the answers given by the interview participants. "A theme captures something important about the data in relation to the research question, and represents some level of patterned response or meaning within the data set." [27, p. 82]. The key of a theme is not necessarily dependent on quantifiable measures but rather on whether it captures something important in relation to the overall research question. Themes of the thematic analysis can be identified in a theoretical or inductive way. An inductive approach means the identified themes are strongly linked to the data themselves; for example, the themes have a relation to the specific questions that have been asked. Braun and Clarke [27] provide a step-by-step guide through the six phases of analysis, which is described below:

1. Familiarizing yourself with your data
2. Generating initial codes
3. Searching for themes
4. Reviewing themes
5. Defining and naming themes
6. Producing the report

Step one is familiarizing oneself with the data, for which transcribing interviews represents an excellent way. Additional reading and re-reading with notes on initial ideas is helpful. Step two is generating initial codes from the data. These codes identify a feature of the data that appears interesting to the analyst. To search for themes in step three, all initial codings from the previous step are sorted and collated into themes. Codings are analyzed by considering whether they may be combined to form an overarching theme. This step ends with a collection of candidate themes, sub-themes and all extracts of data that have been coded. While reviewing the candidate themes in step four, it will become evident that some candidates do not represent themes or that some candidates need to be broken down into separate themes. This step involves a review at the level of coded data, first, within a theme and, second, of a theme in relation to the entire data set. Step five is defining and naming themes. In this step, it is important to consider how each individual theme fits into the broader overall story. It is necessary to consider the themes separately and each theme in relation to the others. Step six is the production of the report consisting of final analysis and write-up. The report provides a concise, coherent, logical, non-repetitive and interesting account of the story told by data.

4. Results

In this section, we describe the ten antecedents and five consequences of time pressure in Scrum-managed software development projects that emerged from our interviews (see Figure 2).

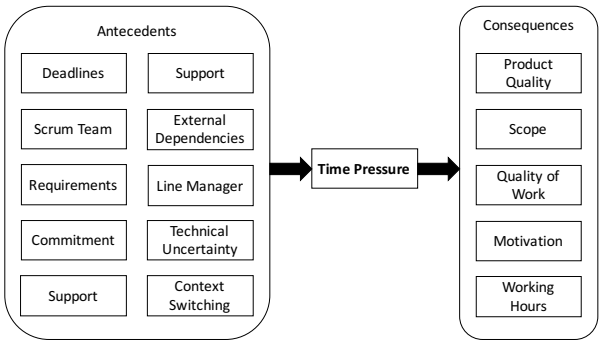

Figure 2: Antecedents and Consequences of Time Pressure

4.1. Antecedents of Time Pressure

4.1.1. Deadlines. Deadlines are an antecedent of time pressure mentioned by five interviewees. Time pressure occurs if company management external to a

project determines a date to deliver the product before the project has actually started. The Scrum Team is simply instructed to deliver the product until the deadline (Andrew). Such an external deadline contradicts Scrum. If a project is set up with an agile approach, it is accepted that the effort is not predictable for all tasks required to develop the product. The agile project begins with a rough idea of the product.

Other reasons for deadlines include advertising tactics or campaigns (George). This includes fixed dates to deliver the product. For example, the company management decides to increase promotion before the World Cup, Easter, or Christmas. Having in mind an idea for an application to increase promotion, management discusses the idea with the Scrum Team. The Scrum Team can improve the idea or can tell the client which part is not possible to realize. However, if the Scrum Team commits to this project, they automatically experience time pressure because they are required to deliver the product before the deadline.

4.1.2. Scrum Team. The Scrum Team is also an antecedent for time pressure. In our interview, four reasons emerged: Onboarding process, team size, temporary reduction, and commitment.

The onboarding process of a new team member is one reason for this antecedent of time pressure (Andrew, Charlie, David). As described by Frederick, a team member who is new in a project will not be staffed with 100% of a workforce: "No team member who has just started one week ago will be included with 100% workforce. At the beginning, the performance of the new team member is 0% and for everybody else in the Scrum Team performance will be reduced, so they have more time to introduce the new team member". The introduction of a new team member is a time-consuming factor. While higher productivity might be expected due to increased team size, the effort of the onboarding process is often neglected. The Development Team has a reduced velocity because of the new team member and is not as productive as before. Therefore, they will be under time pressure to meet the client's expectations.

Another reason for the Scrum Team being an antecedent of time pressure is the size of the Development Team (Frederick). If the client company requires to increase the number of developers, problems are likely to occur concerning the synchronization of all members of the Development Team (e.g., during Sprint Planning). The larger the size of the Development Team, the more time is required to organize Sprints.

Time pressure also arises in case of illness or temporary reductions of the Development Team (Harry, George). Due to such incidents, the rest of the Development Team needs to handle tasks that have been assigned to missing team members. The Development Team in its entirety commits to the Sprint Goal and the items selected from the Product Backlog for the Sprint. While planned holidays can be anticipated in Sprint Planning, illness or other reasons for temporary team reductions cannot be considered in project planning.

The commitment of the Development Team to items from the Product Backlog is another reason for the Scrum Team being an antecedent for time pressure in agile software development projects (Isaac). Within a Sprint or close to its end, the Development Team may have to work overtime to deliver a potentially shippable product; otherwise, the Sprint will not be finished completely (Frederick).

4.1.3. Requirements. Requirements concerning a product are elicited from the client. The client can change the requirements after a Sprint, and this change can lead to time pressure. A reason for changing scope is a requirement that arises during a Sprint. "For example, the top management requires reports of a campaign" (Benjamin). Consequently, more requirements than planned exist in this Sprint, which can lead to a new prioritization of the tasks; some tasks will thus not be realized in this Sprint (Benjamin). For example, campaigns are a marketing product, which generates end customer desires for product specifications. The client recognizes after a Sprint presentation which additional requirement is missing for a perfect product. Because of the growing number of requirements, the scope, which has been agreed on in the beginning of the project, cannot be realized completely (George). "In doubt, we have to prioritize the tasks again and this automatically leads to time pressure" (Benjamin).

4.1.4. Commitment. The Development Team commits to the tasks selected for a Sprint. The Development Team determines the number of tasks for the Sprints on their own. The velocity of previous Sprints can be helpful to improve planning. If the Development Team has committed to a certain number of tasks, they promise to finish these tasks within the Sprint (Isaac). "Time pressure occurs during a Sprint or at the end of a Sprint, if we realize that certain functionalities are missing" (Frederick). These missing functionalities can be necessary for acceptance of the Product Owner. The Development Team can either indicate that this story is not possible to finish in this Sprint or that it must dedicate more time than what is typically allotted to this task. "That way you put yourself under pressure" (Frederick). The worst experience that is mentioned by Frederick is a situation where the Development Team

has already been in the Sprint Review while still working on one task to satisfy the client. In general, all interviewed developers mentioned commitment as an antecedent of time pressure, especially at the end of a Sprint (Isaac).

4.1.5. Support. Extra support effort during a Sprint is another antecedent of time pressure. The planning phase of the Sprint considers a basic support effort. If an extra support request occurs during a Sprint, the overall object of the Sprint is endangered because there is less time for the tasks. For instance, "Bug-Tickets are the trigger for time pressure" (Harry). These tickets involve bugs in the software that has been developed in the Sprint before. The system has been tested and afterward implemented. Sometimes, problems occur due to the new system, and these bugs are documented and routed in a ticket system to the development team (Harry). "The problem is the ambivalence" (Isaac). The support effort is not assessable beforehand. For example, if the last Sprint had no support effort, development performance would be high because the Development Team could focus completely on the Sprint tasks. Therefore, the velocity of the Sprint increases, and the new Sprint is planned with a higher velocity. The next Sprint has substantially more support requests than the previous Sprint. Consequently, the Development Team works on more tasks because they have planned this Sprint with a higher velocity, and they work on more support requests (Isaac). This leads to time pressure in the Sprint for the developers. The support effort cannot be estimated properly. That is why there is always a factor of uncertainty in the support planning.

4.1.6. Contextual Conditions. Contextual conditions are factors that exist and influence the project and may cause time pressure. These factors are inevitable. The Scrum Team has to deal with them and consider these conditions in the Sprint Planning. An example of contextual conditions in the case company is the cold- and frozen-zone, that is, a phase during critical retail business phases such as Christmas or Easter (Benjamin). During the cold zone, changes to a running system are only allowed with special permission by top management. During the frozen zone, no changes are allowed. This security measure aims to ensure a working system for two months and has a big impact on projects. If the project has to be ready for the Easter event, the Scrum Team has to consider that February and March underlie special contextual conditions for the development process. Accordingly, the goal is to deliver a working product until the cold zone starts. This product may not be the perfect product, but it is working. It is a so-called "Minimal Valuable Product" (MVP) (George). "These contextual conditions determined by the company put you under time pressure" (Benjamin).

4.1.7. External Dependencies. External dependencies cannot be influenced by members of the Scrum Team, for example, dependence on other systems or an external delivery. Unexpected events concerning these dependencies can result in time pressure for the Scrum Team. As a good illustration of this antecedent of time pressure, Benjamin describes a case in which the Development Team relied on an external delivery to successfully complete a two-week Sprint. In the reported case, the external supplier had a delay of one week, so the Scrum Team was at the end of the Sprint but had to implement this delivery to successfully complete the Sprint. The supplier's delay led to time pressure. The Product Owner accepted the reason of external delay for not completing the Sprint. However, the Development Team had committed itself to completing the Sprint, which motivated them to finish the task in a timely manner.

4.1.8. Line Manager. Line Manager is the role that involves different political interests outside the Scrum Team. At the company in our study, the line manager is the team's supervisor, while the client is the one who pays. Problems arise for the Scrum Team if interests of the line manager conflict with those of the client. The team has to decide which opinion to follow (Edward). The line manager pursues targets that may be related to budget, staffing, or support for agile development. These targets do not necessarily match with the scope of the Scrum Team; rather, they have a negative impact on the scope. Edward presumes that lack of understanding of Scrum is not the reason for this behavior but the lack of trust of line managers in agile approaches is the problem. Since agile projects are not under the control of line managers, they are eager to determine milestones in order to control the Scrum Team (Edward). Furthermore, such behavior of line managers contradicts agile development because a project manager role does not exist (Edward). "Different motivation for this behavior exists, unfortunately, line manager is not only factual oriented" (Edward).

4.1.9. Technical Uncertainty. Technical developments cannot be completely estimated. Technical uncertainties represent the use of emerging technologies in the project, the downtime of a running system, or the use of a new software. For example, the running system produces a system failure. "This is a source of time pressure because you have to react quickly to find a solution" (Frederick). If there is a

system failure with a running system, it is the overall objective to have the shortest possible downtime.

Another technical uncertainty is the use of emerging technologies or new services (Harry). An exemplary requirement could be the development of an animation for a printing process. First, the development team estimates no problems and schedules this task with one workforce for a half day because the developers estimate that this task will be realized with few well-known functions. Afterward, the Development Team realizes that this is the wrong way, and they have to implement it in a technically different way. This wrong estimation resulting from technical uncertainties leads to time pressure (Harry).

4.1.10. Context Switching. Context switching is an antecedent for time pressure because a number of changes between topics cannot be estimated. If someone interrupts a developer during his work because of another problem, this context switching will steal a lot of time since he has to become acquainted with the previous task again (Frederick). Context switching can be the onboarding process of a new team member (Frederick). In the beginning, a new team member has many questions that he will ask colleagues. Every time he asks a question, the developer, who will answer, has to interrupt his work and loses his focus on the task. "If a Bug-Ticket arises which is more important than the story, time pressure occurs for me due to context switching" (Harry). Bug-Tickets that have a higher prioritization than the Scrum story are an antecedent for context switching, and that is why time pressure exists. Context switching can also involve wasting time in meetings that divagate from their main intention. "I am doing nothing useful here" (Isaac). As a developer, it is your aim to spend your time programming, not wasting time in meetings, which divagate from their main intention. For a developer, this wasted time results in time pressure (Isaac).

4.2. Consequences of Time Pressure

4.2.1. Product Quality. Time pressure can lead to a reduction in testing, lack of technology updates, and missing functionalities, in other words, lower product quality. "Sometimes, it is true that time pressure leads to a lower quality, but this is not planned, but rather a side effect" (Benjamin). Under time pressure, the Development Team needs to work faster; that is why they do not test the software in detail. Reduction of the story is done consciously, but a reduction in product quality due to time pressure is unconscious (Benjamin, Isaac, Harry, Frederick). If the Development Team finishes coding of the software on the last day of a Sprint, there is no time for testing in detail (Benjamin). Thus, the testing is restricted to minimal testing, and the product quality suffers automatically. "You could definitely have a better product with more time" (David).

Product quality suffers due to time pressure because the development team cannot do every task in less time. For example, Charlie mentioned that the surface and backend technology have not been improved due to time pressure. These tasks were planned for the Sprint, but the story has been reduced due to time pressure. Consequently, the software is built with old technology, which means product quality is not as good as planned.

Another consequence is abidance to processes. If the planned architecture is not possible to realize in the Sprint, the Development Team develops "quick wins" (Benjamin). "Quick win sounds bad, but means not to build the perfect product, but rather a workaround" (Benjamin). These workarounds are interim solutions for the product as the Development Team tries to improve the product in the next Sprint. However, the list of tasks for the next Sprint keeps getting longer as a result (Benjamin). For this reason, product quality suffers due to time pressure because the developer team tries to deliver a product with a workaround.

4.2.2. Scope. The change in scope is a consequence of time pressure because the Scrum Team tries to deliver a product in time by reducing the scope. "In doubt, we reduce the tasks with regard to content" (Benjamin). If the Scrum Team is under pressure, they decide to develop a quick and simple solution instead of performing at a higher level of complexity. The most important tasks of the scope are done first, and the rest is shifted (Benjamin). Benjamin points out that quality does not suffer from a reduction in scope. A reduction in scope is a common way to address time pressure in a client's organization (Benjamin, George). The resulting product is called MVP. The client communicates to the Scrum Team what represents the minimum characteristics of the product (Benjamin). The reduction in scope has to be agreed upon with the client. The Product Owner informs the client that the resource capacity is insufficient to deliver the product in time. One way to deliver a product at the end of the Sprint is to deliver an MVP. The other way is to prioritize the tasks in agreement with the client. If it is a non-critical request, the developers continue their story tasks, but if it is a critical request, the client and the Product Owner has to decide which task to focus on (George). "Changing the scope is a weighing up with the client" (George).

4.2.3. Quality of Work. Working under time pressure can lead to oversights in working quality. This consequence is mentioned by product owners only. "The quality of my overall work has suffered" (Andrew). This means the Product Owner does not have the time to be as deeply involved in the topic as would normally be the case. Additionally, due to time pressure, the product owner is not able to work on every single service request (Andrew). The quality of work suffers because of time pressure, but it has a positive side effect – if a Product Owner is under time pressure, she/he prioritizes tasks, so time pressure leads to a review of the working process (Andrew). Under time pressure, the working process of a Product Owner is changed to satisfy the desire of the client (Benjamin). Benjamin mentioned, for example, that the definition of 'ready' or the definition of 'done' includes task documentation. Sometimes, this task is skipped to meet the deadline. "Continuous time pressure influences the project negatively, but temporary time pressure can be positive, if the product owner focuses on the most important tasks" (George).

4.2.4. Motivation. The work motivation can suffer if there is time pressure in the project. The interview participants have different opinions about the consequences for motivation, but all statements are from developers.

The first statement is "Under time pressure I am more motivated" (Harry). This developer states that there is no disadvantage of time pressure, and he is more motivated to work on a task than without time pressure.

The second statement confirms the first, but with a small restriction. Frederick mentions that motivation does not suffer as long as the time pressure is not permanent.

"At the beginning of a Sprint, a lack of motivation exists instead of the end of the Sprint" (Isaac). Overall, the last statement confirms that motivation does not suffer, although again with a restriction. Isaac mentions that in the beginning of a Sprint, a lack of motivation exists, but at the end of the Sprint, there is greater motivation to reach the deadline.

4.2.5. Working Hours. A consequence of time pressure can be more time spent finishing the work. For example, Isaac mentioned that under time pressure, developers try to finish the task even if they must work more (Edward). This is only possible if they extend their working hours and spend more time in the office. They start earlier in the morning and stay later to finish the product before the deadline.

5. Discussion

In our research, we identified ten antecedents and five consequences of time pressure in agile software development projects. In the following, we discuss theoretical contributions, limitations, and practical implications of our work.

5.1. Theoretical Contributions

The findings of the current study indicate that even in agile software development projects, time pressure occurs. Only one study has focused on time pressure in agile software development [1] by investigating time pressure in extreme programming. The findings presented here are based on Scrum projects and show time pressure's antecedents and consequences.

With our results, we contribute to the extant literature by providing a more differentiated answer to the question whether time pressure leads to positive or negative effects. Not all antecedents of time pressure seem to lead to the same consequences. For example, commitment – as one of the Scrum values – of the Development Team to specific goals that lead in combination with other factors to time pressure increased motivation of team members to achieve their goals. While this generally leads to negative consequences such as worsening of product quality, work quality, or work-life balance (due to increased working hours), there are also positive effects in the short run. Short-term positive effects include a higher motivation within the Scrum Team and focus (also a Scrum value) on important tasks. Thus, our results provide evidence that the effects of time pressure not only depend on intensity but also on temporality of time pressures (i.e., how long does the team endure phases of time pressure). In the long-term perspective, neglecting the Scrum values might intensify the negative consequences of time pressure. While our findings provide limited insights in this regard only, we consider a deeper look at the role of Scrum values as a fruitful avenue for further research.

Several of the antecedents of time pressure in the analyzed projects occurred because the company and the Scrum teams did not strictly follow the Scrum Guide [13]. For instance, time pressure that resulted from external decisions to change the size of the Development Team, introduce new requirements during a Sprint, or determine deadlines in advance contrast the roles, events, artifacts, and rules defined in the Scrum Guide. While previous research has shown the benefits of customizing agile methods [11, 29], we contribute to a better understanding of customizing agile methods by revealing that – while customizing methods might be a necessity to fit prevailing

organizational structures – the changes applied to the frameworks can also lead to negative effects (i.e., time pressure and related consequence in our context).

5.2. Limitations

We see two major limitations of our study. First, the number of interviews (9) is rather small and does not cover all perspectives (e.g., we did not interview every member of the Development Team at Alpha and Beta). Consequently, our findings are exploratory in nature, and generalization for the role of time pressure in agile software development projects is limited. Increasing the sample size could advance and confirm patterns observed. Suggested by our observations, a beneficial follow-up might be to analyze adherence to Scrum values as a moderator of effects of time pressure (see also Section 5.1).

Second, all interviews have been conducted with employees of an internal IT service provider from Germany. This service provider works primarily for a company in the retail sector. Each organization and branch has its own characteristics and processes. Employees often align themselves with their employer. Further research could concentrate on a higher diversity of practice partners (e.g., different branches and providers with internal as well as external clients) and in a more global context [30].

5.3. Practical Implications

Understanding specific antecedents can help organizations to mitigate time pressure in Scrum projects. This knowledge poses a necessary condition for changing and managing consequences that arise. Our results provide evidence that time pressure leads to positive effects on the performance of projects under certain conditions.

Regarding Scrum projects, the antecedent of time pressure that results from non-adherence to the Scrum guide emphasizes the importance of keeping high levels of autonomy and self-organization in Scrum Teams. In grasping for external control over Scrum Teams, external management, such as the line managers at the company in our study, need to be aware of the consequences. As clearly expressed in the Scrum guide, “although implementing only parts of Scrum is possible, the result is not Scrum. Scrum exists only in its entirety and functions well as a container for other techniques, methodologies, and practices” [13, p. 16].

Since our results reveal that some findings predominantly result from the view of either members of the Development Team (e.g., commitment) or Product Owners (e.g., quality of work), lack of mutual understanding might be an issue in Scrum Teams. Considering the interviewees’ limited experience with Scrum Teams (see Table 2), we thus emphasize the importance of mutual understanding in Scrum Teams and internalizing the Scrum values.

6. Conclusion

Based on interviews with members of two Scrum Teams, our study provides evidence for a differentiated picture of time pressure, revealing that not all antecedents equally lead to negative consequences and that effects are dependent not only on intensity but on temporality of time pressure as well. We contribute to a deeper understanding of Scrum-managed software development projects by revealing that non-adherence to Scrum rules can cause time pressure in diverse ways. With our exploratory study, we make a first step toward paving the way for research that generalizes such effects as well as formally conceptualize the role of time pressure and its contingencies in agile software development projects.

7. References

[1] O. Malgonde, R. Collins, and A. Hevner, "Applying Emergent Outcome Controls to Mitigate Time Pressure in Agile Software Development," in *Proceedings of Americas Conference on Information Systems*, Savannah: Association for Information Systems, 2014, pp. 1-7.

[2] L. Maruping, V. Venkatesh, S. Thatcher, and P. Patel, "Folding under Pressure or Rising to the Occasion? Perceived Time Pressure and the Moderating Role of Team Temporal Leadership," *Academy of Management Journal,* vol. 58, no. 5, pp. 1313-1333, 2015.

[3] N. Nan and D. E. Harter, "Impact of Budget and Schedule Pressure on Software Development Cycle Time and Effort," *IEEE Transactions on Software Engineering,* vol. 35, no. 5, pp. 624–637, 2009.

[4] R. D. Austin, "The Effects of Time Pressure on Quality in Software Development: An Agency Model," *Information Systems Research,* vol. 12, no. 2, pp. 195–207, 2001.

[5] M. V. Mäntylä and J. Itkonen, "More Testers – The Effect of Crowd Size and Time Restriction in Software Testing," *Information and Software Technology,* vol. 55, no. 6, pp. 986–1003, 2013.

[6] M. V. Mäntylä, K. Petersen, T. O. Lehtinen, and C. Lassenius, "Time Pressure: A Controlled Experiment of Test Case Development and Requirements Review," in *Proceedings of the 36th International Conference on Software Engineering*, L. Briand and A. van der Hoek, Eds., New York: ACM, 2014, pp. 83–94.

[7] H. Shah, M. J. Harrold, and S. Sinha, "Global Software Testing under Deadline Pressure: Vendor-side Experiences," *Information and Software Technology,* vol. 56, no. 1, pp. 6–19, 2014.

[8] P. Bourque and R. E. Fairley, *Swebok. Guide to the Software Engineering Body of Knowledge*, Version 3.0 ed. Los Alamitos: IEEE Computer Society, 2014.

[9] J. Highsmith and A. Cockburn, "Agile Software Development: The Business of Innovation," *Computer,* vol. 34, no. 9, pp. 120-127, 2001.

[10] M. S. Palmquist, M. A. Lapham, S. Miller, T. Chick, and I. Ozkaya, *Parallel Worlds: Agile and Waterfall Differences and Similarities*. Technical Report: Carnegie Mellon University, 2013.

[11] B. Fitzgerald, G. Hartnett, and K. Conboy, "Customising Agile Methods to Software Practices at Intel Shannon," *European Journal of Information Systems,* vol. 15, no. 2, pp. 200–213, 2006.

[12] K. Beck, *Extreme Programming Explained: Embrace Change*. Reading: Addison-Wesley, 2000.

[13] K. Schwaber and J. Sutherland. (2016). *The Scrum Guide. The Definitive Guide to Scrum: The Rules of the Game*. Available at: http://www.scrumguides.org/index.html

[14] R. Atkinson, "Project Management: Cost, Time and Quality, Two Best Guesses and a Phenomenon, Its Time to Accept other Success Criteria," *International Journal of Project Management,* vol. 17, no. 6, pp. 337–342, 1999.

[15] J. K. Pinto, "The Elements of Project Success," in *Field Guide to Project Management*, D. I. Cleland, Ed., Hoboken: Wiley, 2004, pp. 14–27.

[16] D. Basten and A. Sunyaev, "A Systematic Mapping of Factors Affecting Accuracy of Software Development Effort Estimation," *Communications of the Association for Information Systems,* vol. 34, no. Article 4, pp. 51–86, 2014.

[17] S. Grimstad, M. Jørgensen, and K. Moløkken-Østvold, "Software Effort Estimation Terminology: The Tower of Babel," *Information and Software Technology,* vol. 48, no. 4, pp. 302–310, 2006.

[18] L. A. Kappelman, R. McKeeman, and L. Zhang, "Early Warning Signs of IT Project Failure: The Dominant Dozen," *Information Systems Management,* vol. 23, no. 4, pp. 31–36, 2006.

[19] T. Mukhopadhyay, S. S. Vicinanza, and M. J. Prietula, "Examining the Feasibility of a Case-based Reasoning Model for Software Effort Estimation," *MIS Quarterly,* vol. 16, no. 2, pp. 155–171, 1992.

[20] R. D. Banker, S. M. Datar, and C. F. Kemerer, "Factors Affecting Software Maintenance Productivity: an Exploratory Study," in *Proceedings of the International Conference on Information Systems*, Pittsburgh: Association for Information Systems, 1987, pp. 160-175.

[21] M. I. Hwang, "Decision Making under Time Pressure: A Model for Information Systems Research," *Information & Management,* vol. 27, no. 4, pp. 197-203, 1994.

[22] T. DeMarco, *Controlling Software Projects. Management, Measurement & Estimation*. New York: Prentice Hall, 1982.

[23] M. G. Kocher and M. Sutter, "Time is Money - Time Pressure, Incentives, and the Quality of Decision-making," *Journal of Economic Behavior & Organization,* vol. 61, no. 3, pp. 375–392, 2006.

[24] K. Siau, Y. Long, and M. Ling, "Toward a Unified Model of Information Systems Development Success," *Journal of Database Management,* vol. 21, no. 1, pp. 80–101, 2010.

[25] M. D. Myers and M. Newman, "The Qualitative Interview in IS Research: Examining the Craft," *Information Organization,* vol. 17, no. 1, pp. 2–26, 2007.

[26] U. Flick, *An Introduction to Qualitative Research*, 4 ed. Los Angeles: Sage, 2009.

[27] V. Braun and V. Clarke, "Using Thematic Analysis in Psychology," *Qualitative Research in Psychology,* vol. 3, no. 2, pp. 77-101, 2006.

[28] H. J. Rubin and I. Rubin, *Qualitative Interviewing. The Art of Hearing Data*, 2nd ed ed. Thousand Oaks: SAGE Publications, 2005.

[29] D. Pauly, B. Michalik, and D. Basten, "Do Daily Scrums Have to Take Place Each Day? A Case Study of Customized Scrum Principles at an E-Commerce Company," in *Proceedings of the Hawaii International Conference on System Sciences*, Kauai: IEEE Computer Society, 2015, pp. 5074-5083.

[30] T. Dreesen, R. Linden, C. Meures, N. Schmidt, and C. Rosenkranz, "Beyond the Border: A Comparative Literature Review on Communication Practices for Agile Global Outsourced Software Development Projects," in *Proceedings of the Hawaii International Conference on System Sciences*, Koloa: IEEE Computer Society, 2016, pp. 4932-4941.