SCHOOL OF
ECONOMICS AND
MANAGEMENT

Challenges in Self-organizing within Scrum Teams

Theory Evaluation and Recommendations Based on Scrum Practitioners

By
Ahmad Elhemaily and Boman Lyngfelt
June 2019

Master's Programme in Management

Supervisor: Ola Mattisson
Examiner: Bo Göransson

Abstract

Being self-organizing is a core element of successful Scrum teams. Such teams require a high level of autonomy to perform, yet they operate within the context of an organization and its rules. Scrum teams should decide their work process, what to work on and how to deliver value for customers. The existing literature on Scrum teams investigates the challenges Scrum teams face in their daily life or the challenges teams face when transforming their work process from traditional product development (such as waterfall) to agile product development (such as Scrum). However, little is known about the challenges and impediments that directly relate to and threaten the self-organizing nature of Scrum teams. Through a mixed method approach inspired by grounded theory elements, this study compares the challenges acknowledged in the literature with the practical experience of Scrum practitioners. The study presents expert knowledge from 13 interviewed Scrum masters in various European countries as well as 65 questionnaire respondents (Scrum practitioners) from all around the world. The study then presents recommendations based on these practitioners' expert views. The recommendations are aimed towards future Scrum teams to help them mitigate and possibly prevent challenges that could possibly threaten their efforts towards being self-organizing.

Acknowledgments

We would like to express our gratitude to the participants of our study, without whom this would not have been possible. We would also like to thank our Thesis Supervisor, Ola Mattisson, as well as the peer-review group from the M.Sc. in Management programme for valuable input and support.

Special thanks go to Fredrik Wendt, Agile Coach at ProAgile AB, for taking the time to enhance our insights into the Scrum framework and providing in-depth experiences that highlight the challenges we investigated. Special thanks also go to Benjamin Lyngfelt for constant support and encouragement throughout our writing process.

1. Introduction

In traditionally-managed projects, team management is said to be focused on a command and control style where meticulous planning aims to maintain a routine and minimize change (Nerur, Mahapatra & Mangalaraj 2005, p. 75). Scholars such as Carroll (1999) Roper and Phillips (2007), and Hamel (2017) have highlighted the cumbersome nature of the traditional styles of product development and noted a shift to self-organizing teams. A typical example of such traditional product development models is waterfall. It is a "linear sequential life cycle model" where "each stage must be completed before next one can start. At the end of each stage the project is reviewed to ensure compliance with requirements" (Stoica, Mircea and Ghilic-Micu 2013, p.66). Among the main disadvantages of such product development models is the lack of iteration and feedback. Such methods are prone to suffer from "problems that occur if there is no iteration and feedback among phases, beginning with the need to define the system requirements specifications accurately. Unless there is iteration and feedback, there may be no way to improve initial imperfections in one of the later phases" (Cusumano & Smith 1995, p.4) Agile emerged as a flexible alternative placing critical importance on the interaction with markets and customers. Within agile, Scrum highlights self-organization as one of the core elements needed for the success of an agile team. That is why this paper focuses primarily on self-organizing teams operating within the Scrum framework.

A definition that encompasses all self-organizing teams in a fair and meaningful way is an almost impossible task to produce. A general description, however, is provided by Yeatts and Hyten (1998, p. 816) who depict such teams as teams that are "responsible for managing and performing technical tasks that result in a product or service being delivered to an internal or external customer." A central core of such teams seems to be a level of autonomy and self-direction that traditionally managed teams do not enjoy. This empowerment of team autonomy led to the birth of agile methodologies that enable teams to streamline their competencies, manage complex products and respond to changing customer needs promptly. This shift in power and boost in team autonomy, unfortunately, led to an association of agile project management methodologies with anarchy where "everybody does what he or she wants to" (Rigby, Sutherland & Takeuchi 2016, p. 3). To fully reap the benefits of their agility, Scrum teams need to be able to choose how to work but also what they can or cannot work on. This may lead to friction with other parts of the organization where the mindset of managers could still be aligned with a traditional non-agile mindset. This paper investigates these limiting impediments and factors that hinder the process of self-organization in Scrum.

It could be common for an agile team to be self-managed. However, what makes them self-organizing is that on top of their autonomy in choosing work processes, Scrum teams define their own purpose, and the tasks and tools needed to achieve it. Holmes

(2011, p. 69-70) defines self-managed teams as teams collaborating to fulfill a goal that has been set by others outside the team. Self-organized teams, on the other hand, protect themselves from outside influence and try to directly affect what tasks can or cannot be asked of them. This idea further complicates the perception that autonomy and self-organization are synonymous with anarchy. From this, many such as Stray, Moe and Hoda (2018) have noted the difficulty and challenges for teams to achieve high levels of autonomy as needed in a self-organized team.

A lot of research has been done to bring to light the main challenges Scrum teams face in their general development. However, a gap exists in the literature when it comes to extensive research highlighting the severity and frequency of challenges that directly relate to self-organizing within the Scrum framework. Murugesan (2016) describes this gap emphasizing the lack of extensive research and our lack of understanding of the daily impediments self-organizing teams face when working in an agile work environment. She refers to this gap saying: "While existing literature has explored some challenges in self-organizing agile teams, little is known about how the high involvement of self-organizing agile teams overcome the challenges in day-to-day activities." (Murugesan 2016, p. 1). This gap in the research is also acknowledged by Hoda, Marshall, and Noble (2011). They refer to this in the following statement:

> *Self-organizing teams have been identified as one of the critical success factors of Agile projects* […] *However, while self-organization is a vital characteristic of Agile teams, there has been limited research on the subject and almost none across multiple projects, organizations, and cultures.*

Furthermore, in the literature for agile product management, there is a need for a compilation of expert views and practical solutions that tackle the challenges pertaining to self-organization. Therefore, one of the goals of the study is to assess the challenges that the current literature highlights. An investigation is then conducted to understand the impact of these challenges on Scrum teams' ability to self-organize. A theoretical gap is then filled using data from Scrum Masters and other practitioners in order to generalize those expert opinions and strategies to aid future Scrum teams in self-organizing.

1.1. Study Background and Motivation

Agile product management has established itself as a discipline rather than a passing trend or fad. Scholars such as Melnik and Maurer (2003) and Gregory, Barroca, Taylor, Salah, and Sharp (2015) comment on this as they agree that agile as a whole is "here to stay" when it comes to product development. A lot of literature highlights the gains and strengths of self-managed teams and especially in fast-paced environments such as agile software development. This creates the need for a thorough exploration of the limiting

factors and challenges that might prevent Scrum teams from capitalizing on the team empowerment and autonomy that Scrum promotes.

According to Tata and Prasad (2004), a team that is self-organized in a way that allows a high level of autonomy can affect decisions throughout the organization. It is a team that enjoys a unique and independent work process in a way that forces the top management to respect and work with it, instead of imposing commands and tasks on it. This creates a paradox for the top management whereby it is their purpose to raise predictability and ensure order, while at the same time respecting the autonomy of the Scrum team. The Scrum team makes decisions to protect its self-organizing nature, which is something that can undermine an organization's set plans and may force it to adapt. This is one of the reasons why a study in the challenges of self-organizing in Scrum teams is of value for future organizations using the framework.

Agile practitioners and experts understand agile teams can appease "tightly wound project management authorities" by conforming with organizational rigid plans but this "won't deliver business value in volatile, high-speed environments" (Highsmith 2002, p. 18). This is because a self-organizing team may lose its essence and autonomy when directed and controlled by the top management. However, in order to maintain order, the goals of the Scrum team must align with the organization as a whole to ensure coherence and capitalize on the power of self-organizing. This problematic nature of self-organization in a highly volatile agile work environment leads to important research questions that this study aims to answer (which will be discussed in 1.3).

This study positions itself within the discipline of agile product management. Since Scrum is the agile framework in question, a lot of focus is placed on the management of software development because, historically, this is the area where Scrum has been used the most. Agile methodologies and mindsets have come to leave a great impact on the area of product management which is why this area is of interest for research. In evidence of this, a report of ForbesInsights sponsored by the Project Management Institute surveyed 506 Senior executives revealing that "a staggering 92 percent of executives believe that organizational agility, or the ability to rapidly respond to market conditions and external factors, is critical to business success" and that agile frameworks help with that (ForbesInsights 2017, p. 2).

This means that Scrum, as a part of agile, is a framework that is assimilated more and more in the work of companies working within the discipline of product management. The solutions to the challenges this study provides could be extended to fit other agile frameworks and management styles such as Kanban, Extreme Programming, etc. However, the primary focus is on agile product management within Scrum work environments.

1.2. Purpose

This study aims to provide generalizable solutions to the challenges Scrum teams might face in their efforts to be self-organizing. It compares the challenges discussed in previous research with the experiences of current Scrum practitioners. This is to investigate how those challenges manifest themselves in practice and as a result come up with remedies for them. Thus, another research objective is an improved understanding of both the frequency and severity of those challenges in reality.

1.3. Research Questions

Q1: How much do the challenges to self-organizing acknowledged in the literature reflect and align with the practical experiences of Scrum practitioners?
Q2: How frequent are these challenges in reality and how severe is their impact on a Scrum team's effort to self-organize?
Q3: What are some possibly generalizable solutions and recommendations that Scrum practitioners offer with respect to each of these challenges?

2. Scrum Theoretical Background

The following section offers a theoretical overview of the Scrum framework. This is to familiarize the reader with its components and work process, as well as to clarify some of the terminology used throughout the paper. First, an explanation of the history of the framework's development is provided.

2.1. Scrum History and Evolution

Jeff Sutherland, one of the founders of Scrum, came up with the term "Scrum" inspired by a famous 1986 article in the discipline of product development (Bergman 2011, p. 7). The article, entitled "The New New Product Development Game," discusses a rugby mentality to teaming as it depicts a shift in Japanese and American companies to "a holistic method—as in rugby, the ball gets passed within the team as it moves as a unit up the field." (Takeuchi & Nonaka 1986, p. 137). The agile and self-organizing nature of Scrum teams is a product of a number of Scrum rules, values, roles and ceremonies that a team should fully grasp and embody.

Scrum is focused on creating an iterative mindset where change is embraced in an agile fashion to respond to changing customer demands. It is noteworthy that Scrum itself has been constantly changing and evolving over the last decades. In its early introduction by Schwaber, Scrum is described as a "development process" and "a management, enhancement and maintenance methodology" (Schwaber 1997, p. 1-3). Currently, Scrum is being widely described as being a framework rather than a development process or methodology. Schwaber (2004, p. xvii) describes Scrum as "a framework and set of practices" that enable its practitioners to raise transparency in projects and iterate their work process to match customer demand. The definition has since then been further refined in the Scrum guide to be a "framework that has been used to manage work on complex products since the early 1990s. Scrum is not a process, technique, or definitive method. Rather, it is a framework within which you can employ various processes and techniques" (Schwaber & Sutherland 2017, p. 3).

2.2. Scrum Overview

Scrum consists of roles, events, artifacts, and rules that have a unique purpose within the framework and are necessary success factors (Schwaber & Sutherland 2017, p. 3). A Scrum team should be a team of 3-9 employees. On a Scrum team, there are three roles: a product owner, a Scrum master, and developers. They work in sprints of no longer than one month towards a pre-defined sprint goal to deliver a functional and releasable end-product called an increment. The Scrum roles are detailed below.

Scrum Roles:

1) Product Owner:

The product owner's (PO) responsibility is maximizing the value of their team's activities and ensuring stakeholder and customer feedback is being received and understood. A PO's main task is managing the product backlog (PB) which is an ordered list of tasks that must be carried out in the course of developing a product, sometimes stretching over many years (Pries & Quigley 2010, p. 22). PB items are sometimes referred to in terms of user stories. A user story is described as "functionality that will be valuable to either a user or purchaser of a system or software" which includes "a written description of the story used for planning and as a reminder" (Cohn 2004, p. 4).

A PB is never complete and exists as long as the product being developed exists and is on the market (Schwaber 2010, p. 4). The PO's role is to make sure the tasks at the top of the PB are the ones with the highest value for business and highest clarity for the team. As put by McKenna (2016, p. 28-29), "the PO lives in the backlog" and "should be constantly prioritizing the Backlog" to respond to the rapid change in business value due to market trends or different customer requirements.

2) Development Team:

The development team consists of the employees responsible for turning the product backlog items into a releasable increment. Cole and Scotcher (2016, p. 94) call development teams the "project engine house." This is because they are cross-functional which means they have all the competencies required to fulfill PB items "without needing micro-management" or external help (Cole & Scotcher 2016, p. 94). A development team is a single unit with no sub-teams and no specialized job titles other than the role of 'developer' since accountability belongs to the whole team (Schwaber & Sutherland 2017, p. 7). Finally, the development team has ownership of the second Scrum artifact, the sprint backlog. The sprint backlog is a list of PB items the development team has agreed to fulfill during the sprint and "how the team plans to design, build, integrate, and test" them (Rubin 2013, p. 18). This high level of autonomy and accountability within the development team reflects the self-organizing nature of Scrum but can create challenges that this study aims to explore.

3) Scrum Master:

Since the Scrum Master is the role most concerned with overcoming challenges and helping others with them, this study places considerable focus on it. The Scrum Master is described in the Scrum Guide as a "servant-leader for the Scrum team" who is "responsible for promoting and supporting Scrum" (Schwaber & Sutherland 2017, p. 7). Among a Scrum Master's central tasks is removing impediments and obstacles hindering the progress and high-performance of the development team (Schwaber & Sutherland

2017, p. 8). The Scrum Master can, thus, be seen as a protector of the self-organization of the Scrum team. This is because the role entails facilitating the interactions of outsiders with the Scrum team. That could be done by coaching various parties on Scrum values and practices to ensure the team's self-organizing autonomy is fully respected.

Events:

The Scrum framework depicts five events one must understand in order to grasp how Scrum teams self-organize. These events offer chances to investigate a team's process and improve it in what is called *inspect* and *adapt* opportunities. *Inspect* and *adapt* opportunities are there for the team "to see and learn what is going on and then evolve based on feedback, in repeating cycles" (Deemer, Benefield, Larman & Vodde 2010, p.14). The five Scrum events and inspect and adapt opportunities are detailed below:

1) **The Sprint:** This is the period of work the Scrum team agrees to work in order to produce one increment. It should not exceed the period of one month to preserve the agile nature of Scrum and reduce complexity. It can be seen as the "container for all the other Scrum events" or "mini-projects" within the lifecycle of a product's development (Cole & Scotcher 2016, p. 96).
2) **Sprint Planning:** This is a meeting at the beginning of every sprint where the entire Scrum team craft a sprint goal describing what functionality can be produced in the sprint. The second output of this meeting is a sprint backlog detailing what PB items can be done in the sprint. As a product of this, the development team should be able to describe to the PO how it will attempt to work as "a self-organizing team to accomplish the Sprint Goal and create the anticipated Increment" (Schwaber & Sutherland 2017, p. 11).
3) **Daily Scrum:** This is a daily 15-minute meeting in which the development team details how it will work throughout the day. It is also an opportunity to discuss improvements to its work process (Schwaber & Sutherland 2017, p. 12).
4) **Sprint Review:** This meeting happens at the end of the sprint and is the opportunity to review the produced increment. The PO should ensure the presence of stakeholders at this meeting. The output is an adjusted PB based on what has been accomplished and based on stakeholders' feedback (Schwaber & Sutherland 2017, p. 13).
5) **Sprint Retrospective:** This meeting signifies a self-inspection landmark for the Scrum team after the end of a sprint. The output is a plan of process and possibly relationship improvements that should be enacted in coming sprints.

Rules and Values:

A final element of the Scrum framework that enables self-organization is the team's alignment with the five Scrum values. This is a decisive factor setting self-organizing team apart from traditional command and control teams (Stellman & Greene 2014, p. 104). These values according to the Scrum guide (Schwaber & Sutherland 2017) are:

1) Courage: Among other traits, this is signified with the ability to accept and initiate change.

2) Commitment: Allocating all efforts to boosting the Scrum teams performance and ensuring the delivery of an increment.

3) Respect: Among other traits, respecting the commitment and capability of fellow team members.

4) Focus: Securing the sprint goal and steering away from disruptions and outside forces

5) Openness: Maintaining a high level of transparency that enables self-organization and a smooth flow of information in the organization.

3. Literature Review

This section reviews the challenges and impediments that Scrum teams face in self-organizing according to existing studies touching on the topic. Presented below are the collected findings showing the current perception of Scrum team challenges in self-organizing. The challenges concern matters regarding organizational structure, interpersonal issues, and difficulties in adapting to context.

3.1. Challenges Found in Literature

Below are the 15 challenges outlined and explained from previous research pertaining to self-organizing within the Scrum framework. The challenges are presented one at a time.

1. Support from the Organization

The Scrum Master is seen as a protector of the development team enabling collaboration with top management and ensuring that "those outside the Scrum Team understand which of their interactions with the Scrum Team are helpful and which aren't" (Schwaber & Sutherland 2017, p. 4). A challenge can occur when the Scrum master often is working against rather than with the top management. One of the Scrum values is courage. And thus, a good Scrum master has "the courage to protect and guide the Team" and is able to stand up to "stakeholders at the right time" (Jeldi & Chavali 2013, p. 2). This might cause friction between the Scrum team and the organization which could sometimes lead to the top management attempting to assert control. This can manifest itself in the challenge of a lack of sponsorship of projects and a general lack of support from the organization for the Scrum team. This can also mean that the top management does not support the Scrum team when the team suggests tasks or projects it wants to take on. The management may instead want to dictate how the work is to be done or what is to be achieved in the sprints. This could undermine the Scrum team's autonomy and hinder its endeavors towards self-organization.

Another alternative explanation of this challenge can be that "one of the primary reasons why the senior management were hesitant" about projects "was that they were focused on the final goal and delivery of the product and did not want to risk it in the slightest way" (Hoda & Murugesan 2016, p. 250). Either way, if a Scrum team feels that it is not backed by the organization and can choose its processes, tools or methods, team autonomy could suffer and further friction between the team and the organization might manifest itself.

2. Lack of Trust

When working in an agile complex environment with so many variables affecting process stability, a high level of trust is needed. Scrum values such as respect, openness, and commitment dictate this. If a team lacks mutual trust between members, the development team may find difficulties in committing to the backlog and in meeting sprint goals (Moe & Dingsøyr 2008, p. 15). Consequently, the success of an agile team may depend on a level of trust enabling them to be agile in the first place (Nerur, Mahapatra & Mangalaraj 2005, p. 76).

A low level of trust between team members could threaten commitment, an integral value of the Scrum framework. On the other hand, a low level of trust between the team and external managers may cause managers to demand more reports and documentation to regain control, something that might threaten and undermine the Scrum team's autonomy (Stray, Moe & Hoda 2018, p. 3). This is a challenge that usually takes time to tackle and it is naturally difficult for Scrum masters to identify the root source of the problem to remove the impediment.

Furthermore, decision-making in agile is supposed to be based on its lightweight framework and responsive team members. In a sense, "decision making in this environment is more difficult compared to the traditional approach where the project manager is responsible for most decisions" (Nerur, Mahapatra & Mangalaraj 2005, p. 76). A lack of trust between members could further complicate this task and threaten self-organization even more.

3. Task Dependency

Meaningful estimates of the effort and time of tasks (discussed later in 10) in order to deliver a releasable increment at the end of a sprint can be challenging just considering the difficulty of each task. When also considering which tasks are dependent on other tasks to first be completed, the planning becomes even more complex. A self-organizing team, as flexible and adaptable as it may be, could still encounter challenges in "task dependency that exists between the tasks within a sprint or across sprints" (Hoda & Murugesan 2016, p. 252). Some user stories will build upon others, for instance, testing cannot be done until enough code has been produced and so on. A self-organizing team left alone, should have the tools to plan accordingly. However, the team often operates in the context of its organization and in relation to clients, something which can result in task dependency with entities outside the team.

> *If the team needs to reach an agreement or synchronize deliverables with too many experts, managers, stakeholders, and other teams, their authority to make decisions regarding the development process, technology, architecture, and product is reduced. For*

> *example, the software architecture may limit team autonomy if the architecture results in many technical dependencies between teams, which requires a constant need for alignment and coordination.*
>
> (Stray, Moe & Hoda 2018, p. 3)

This may not only affect the team's ability to efficiently plan and execute a sprint, therefore not being as autonomous and self-organizing as they should be, but it might lead to a more severe "consequence of task dependency" which is the "cancellation of sprints" (Hoda & Murugesan 2016, p. 252). Waiting for another team or part of the organization, outside resources or clients, might lead to the planned sprint becoming obsolete altogether since planned actions cannot be executed. The Scrum guide highlights the negative effects of these cancellations as they "consume resources since everyone regroups in another Sprint Planning to start another Sprint. Sprint cancellations are often traumatic to the Scrum Team" (Schwaber & Sutherland 2017, p. 10)

4. Negative Perception of Scrum Events

One of the criticisms leveled against the Scrum framework is that the ceremonies and Scrum events take developers away from doing their job, which can lead to a drop in efficiency (Scrum Alliance, p. 3). Agile frameworks, in general, are a reaction to a lack of responsiveness in traditional product management. This led to a shift to agile frameworks in a way to inspire "more lightweight and faster development processes" in complex industries (Holmström, Fitzgerald, Ågerfalk & Conchúir 2006, p. 8). This means that a rigid mindset that sees Scrum events as a waste of time may signify a regression to traditional non-agile project management styles. A depiction of this challenge is presented by Akif and Majeed in their 2012 study of 20 agile teams where they say:

> *Due to increase in communication, some team members feel disturbance in their work. They are not able to concentrate properly. Effective communication also adds frequent meetings. Some team members think attending all meetings are unnecessary for them, hence they do not show their interests in those meetings which are not directly related to them.*
>
> (Akif & Majeed 2012, p. 3)

5. Geographic Issues

Being geographically dispersed may threaten self-organizing teams due to more problems with communication and a lower level of transparency in the team. A Scrum Alliance report claims that although "high quality communication systems do help to some extent, it is not same as being face to face at the same location." (Scrum Alliance, p. 2). This

creates extra work for the Scrum master whose job is to raise transparency in organizations and ensure constant communication.

Some of the specific problems due to geographic dispersion may be time difference, less frequent communication and the risk of uninformed team members in discussions (Murugesan 2016, p. 2). This can be a difficult impediment for Scrum masters to mitigate due to the fact that it deals with circumstances that they rarely are able to influence. Sometimes being geographically dispersed is a necessity rather than a choice for teams. This might lead to the introduction of software or tools that the Scrum master might have otherwise not needed to use.

6. Multiple Projects

In a study of 49 Scrum practitioners across 7 countries, it was revealed that 70 % of them "had been working on multiple projects simultaneously for the last 3 months" (Jia, Larusdottir & Cajander 2012, p. 335). This may become especially problematic for a framework that attempts to eliminate waste and delays by retaining high levels of focus within the development team. The problem can also lead to a drop in efficiency because "when a person needs to concentrate on two different deadlines, switching from one task to another would take a lot of time" (Murugesan 2016, p. 2).

This is a challenge that may reflect a problem in self-organization since it is affected by how much the Scrum team is able to choose which tasks or projects it can fulfill. As established previously, a truly self-organizing team chooses what it can work on successfully based on its cross-functionality and the competencies in the team. Focus and agility could drop if developers are constantly cycling between projects and possibly answering to different acceptance criteria given by multiple stakeholders.

7. Personality Clashes

Working in a Scrum environment requires a high level of collaboration and putting aside differences in order to achieve self-organization. In Scrum, diversity and differences in expertise are promoted and even celebrated. Scrum teams should be "cross-functional" and this means that the team, by definition, "includes all the expertise necessary to deliver the potentially shippable product each Sprint" (Deemer, Benefield, Larman & Vodde 2010, p. 6).

This is positive and should ensure a high level of competences in the team but high diversity sometimes breeds conflict due to different personalities on the team. Scholars such as Nerur, Mahapatra, and Mangalaraj (2005) point to a difference in mindset and personality that is especially relevant for an agile software development environment. They assert that "for programmers accustomed to solitary activities or working with relatively homogeneous groups of analysts and designers, the ideas of

shared learning, reflection workshops, pair-programming, and collaborative decision making may be overwhelming" (Nerur, Mahapatra & Mangalaraj 2005, p. 76). This is further supported in a study by (Cho 2008) where personality differences manifest themselves in how developers view what a positive work environment is. He asserts that some of the interviewed developers found open-space working environments as a boost to creativity or collaboration, while others saw it as a mere distraction (Cho 2008, p. 194). These differences and the diversity in the team could lead to personality clashes which the Scrum master will have to handle in order to ensure productivity and a positive work environment.

Furthermore, the Scrum Alliance report warns against previous "old prejudices and notions" that new workers bring to teams opening the team for a bigger risk of personality clashes and conflict (Scrum Alliance, p. 2).

8. Division of Tasks

One challenge that can come from being self-organized, and therefore also threaten a team in their efforts to be self-organizing, is the division of appropriate tasks among team members. Cross-functionality and differences in experience can lead to decisions that threaten the agile nature of the team, simply due to the fact that team members are different and therefore perceive tasks differently. Self-organizing within the team can lead to "underqualified people who have no idea of the task [assigning] it to themselves" while "overqualified people self-assign the tasks" (Murugesan 2016, p. 2) which can waste both time and resources.

Maybe it is a natural conclusion to delegate a task to the most knowledgeable, who then will have less of a challenge in executing the task, but that may not necessarily be the best use of resources available. Research shows that "team members picked up the task based on their individual expertise or specializations" which can be a threat to the team's cross-functionality (Hoda & Murugesan 2016, p. 252). Even with the "help of the scrum master or the project manager [...] the task assignment was generally driven by team members' previous experience" (Hoda & Murugesan 2016, p. 252). This suggests that a self-organizing team, by organizing out of preference and experience, actually might harm its own capability to self-organize, since choices then become more and more pre-determined and not necessarily in the team's best interest.

9. Drop in Quality and Technical Debt

The Scrum guide specifies that in the process of self-organizing Scrum teams "no changes are made that would endanger the Sprint Goal" and "quality goals do not decrease" (Schwaber & Sutherland 2017, p. 9). This is sometimes a tough formula to satisfy, especially if the team is organizing itself and not being told what to do. Akif and

Majid (2012, p. 2) depict the challenges of trying to achieve this when they say that "due to agility of work in Scrum, teams have an obligation to present something in a short time regardless of their scope in Sprint Planning. Because of this sometimes teams ignore quality of software."

This is a problem known to agile practitioners as technical debt which depicts the "trade-offs made during software development to ensure speedy releases" (Codabux & Williams 2013, p. 8). In these cases, developers could "sometimes accept compromises in a system in one dimension (e.g., modularity) to meet an urgent demand in some other dimension (e.g., a deadline), and that such compromises incur a "debt" (Brown et al 2010, p. 1). This debt means that the development team has to go back to redo tasks or improve the quality of the work done. Too much technical debt may affect the overall velocity, which is one measure of a Scrum team's effectiveness. Velocity is a "measurement of "amount of work done", where each item is weighted in terms of its initial estimate" (Kniberg 2015, p.20). Simply put, it describes how many backlog items can be 'done' by the development team each sprint. Velocity could be compromised if the team cannot self-organize to meet goals in a way that does not compromise code or product quality.

10. Lack of Valuable Estimations

Being self-organized, not only while working but also in the planning of work to be done, is among the core features of Scrum teams. This is in contrast with traditional management situations where it is customary for a designated decision maker, typically a manager, to make effort estimations and delegate tasks (Brooks 1995). "Agile methodologies rely on speculation, or planning with the understanding that everything is uncertain, to guide the rapid development of flexible and adaptive systems of high value" (Nerur, Mahapatra & Mangalaraj 2005, p. 77). It is, therefore "recommended to perform the estimation exercise with the whole team so that individual members can provide their input and perspectives into the estimations." (Hoda & Murugesan 2016, p. 251).

The team in Scrum should be cross-functional. This means that it has access to all the "competencies needed to accomplish the work without depending on others not part of the team"(Schwaber & Sutherland 2017, p. 6) Therefore, it can be positive to have the team that are going to ultimately perform the task and have the technical expertise perform the effort estimation. Some challenges might, however, arise from this. Since cross-functionality is an important part of Scrum teams in order to have multiple competencies in-house, there may also be discrepancies in how tasks are being perceived, based on experience and areas of specialization. Hoda and Murugesan (2016, p. 251) stated that "experienced professionals tended to buffer some extra time for unexpected issues and therefore inflated their estimation, the newer team members tended to underestimate the workload." One member's estimation might not be accurate for another

member performing the task, and since estimations are being made with the whole team, this might affect the effectiveness or usefulness of the estimation.

11. Lack of Individual Accountability

One of the pillars of self-organizing in agile is that the whole development team bears responsibility for its own work. As the Scrum guide puts it: “accountability belongs to the Development Team as a whole” despite any special expertise that individuals on the team may have (Schwaber & Sutherland 2017, p. 7). This means that success or failure belongs, as a whole, to the team. This is a positive set-up ensuring a sense of belonging which is in line with the Scrum values of commitment and respect.

However, this can prove to be problematic in some situations and lead to pent-up tensions in the team. The team is entirely accountable which means that personal successes or failures affect the whole team. One can, however, consider the extreme case of a certain developer single-handedly leading to the success or failure of a sprint. The accountability, success or failure will still be attributed to the team and not that person. The risk here is that “this can potentially develop into a very serious problem for those who are responsible for the success of the project’’ which can breed conflicts and be less rewarding to some individuals (Scrum Alliance, p. 3). Self-organizing could suffer if individuals in the team do not share the same ownership of the work process or the same accountability regarding its outcomes.

12. Delayed and Changing Requirements

Customers that make changes in requirements and add new ones at the last minute might be a challenge to any team, no matter what framework or structure. In the best of worlds, you could expect the client to be “Collaborative, Representative, Authorized, Committed, and Knowledgeable” (Boehm & Turner 2009, p. 44) but “it is not an easy task to find such persons, especially for complex systems” (Nerur, Mahapatra & Mangalaraj 2005, p. 76). Especially for agile teams, it is important to adapt regardless since “the tasks of eliciting, clarifying, and estimating requirements as well as dealing with any changes in requirements now falls with the team” and there is no longer “project managers or business analysts [...] to procure and manage requirements” (Hoda & Murugesan 2016, p. 249).

The iterative nature of Scrum is dependent on getting accurate feedback from customers in order to adapt and change in the next sprint. “If the requirements were delayed,” it can become “difficult for the team to estimate the user stories” which can lead to the whole process being dependent on the delayed requirements (Hoda & Murugesan 2016, p. 249).

Since the team has planned the next sprint carefully and is bound to a specified timeframe, it can become vulnerable to when a customer wants to change the requirements. “If the requirements were changed, the estimations needed to be redone” which could result in new stories being dependent, changes in priority and in the worst case that “the negative impact of unsystematic changes in requirements [...] sometimes [lead] to cancelled sprints” (Hoda & Murugesan 2016, p. 249-250).

13. Excessive Documentation Requirements

Scrum relies on constant feedback and daily discussions in order to raise transparency and promote communication. The Scrum events, especially ones where stakeholders are encouraged to participate such as the Scrum review, serve as an opportunity for the whole organization to gain an insight into the work process of the Scrum team. Despite this, some organizations find it difficult to only have such meetings and thus, they demand that their Scrum teams produce reports and documentation detailing what is being done and how. This can possibly undermine the team’s self-organizing power and might negatively affect their focus, one of the core Scrum values. So in many cases, organizations embraced Scrum in name but “still required their teams to submit relatively heavy documentation such as frequent status reports” and as a response to this “teams perceived this as a challenge to practicing agile which is meant to be a light-weight process involving minimum documentation, and wished for a change in senior management approach in this regard” (Hoda & Murugesan 2016, p. 250).

Furthermore, another problem that can undermine autonomy and self-organizing arises from this challenge. This is described by Akif and Majeed as follows:

> *Agile believes in no documentation so does Scrum; however this phenomenon is still not successful in real environment where things come in and out through email or any other source, which is quite difficult to track. From the survey, it has been found that requirements get changed through emails without relating them properly in Product Backlog which creates issues for team in traceability.*
>
> (Akif & Majeed 2012, p. 3)

This lack of traceability and making demands through channels such as emails or software might undermine self-organization. This is because it can render a great part of the product owner’s role useless. According to the Scrum guide, the product owner should be “the sole person responsible for managing the Product Backlog” (Schwaber & Sutherland 2017, p. 6). And the Product Backlog is said to be “the single source of requirements for any changes to be made to the product” (Schwaber & Sutherland 2017, p. 15). This means that any change of process should definitely go through the product owner as the top management consults with him on the best course of action. However,

when documentation and reports bypass the product owner, the self-organizing nature of the whole team can be compromised.

This challenge highlights that the top management may be stuck in a traditional product management style of work and has not fully been able to embrace agile. Traditional approaches use documentation of products to raise "traceability of design" while "agile methodologies, on the other hand, encourage lean thinking and cutting down on overhead, particularly documentation" (Nerur, Mahapatra & Mangalaraj 2005, p. 76). This means that the knowledge in traditional methods of product management remains, to a large extent, explicit and documented. On the other hand, "much of the knowledge in agile development is tacit and resides in the heads of the development team members," which makes the organization dependent on the development team, a situation that is sure to unsettle some organizations due to the shift of control and autonomy to the Scrum team (Nerur, Mahapatra & Mangalaraj 2005, p. 76). This shift of control and empowerment of autonomy are, however, prerequisites for an accountable self-organizing Scrum team. Therefore, excessive documentation can become a challenge and a threat to an agile team's efforts towards self-organization.

14. Different Perceptions of Communication Software

Software tools that facilitate communication (Slack, Trello, Jira, etc.) are sometimes met by resistance in Scrum teams due to different perceptions of how useful these tools are. On the one hand, some view such software as a facilitator of communication that raises the level of transparency in the organization. And since transparency is one of Scrum's pillars according to the Scrum guide (Schwaber & Sutherland 2017, p. 5), software or tools promoting that should be seen as an improvement to team processes. On the other hand, some might view the tools as disruptive and distracting them from doing their work.

Alternatively, the tools could be seen as inefficient since other modes of communication such as written, face-to-face, etc. work better for different people due to personality differences. Denning, for instance, warned against placing too much emphasis on software while losing track of the purpose and mindset of agile product development when he says that:

> *People are looking for and buying, processes and tools when it should be about the Agile mindset. There is a risk today there that Agile is being co-opted by natural business forces that also fuel its growth. It's not that processes and tools aren't useful. It's just that they are not as important as the Agile mindset and individual interactions.*
>
> (Denning 2016, p. 19)

15. Varying Stances on Agile Training

The literature points to a lack of consensus by organizations when it comes to how much formal agile training or certifications are valued. In some organizations, the management sees the value of being trained in agile even for developers and technical workers. In this case, they encourage training and sponsor their employees. In other cases, the management sees that it is a manager's or Scrum master's job to train the other team members and familiarize them with Scrum (Hoda & Murugesan 2016, p. 250).

In an investigation of 20 Scrum practitioners, Akif and Majeed discovered that 10 of them "lack formal training of scrum and are unaware of the scrum process. The knowledge that they have gained is either because of the other team fellows or from their scrum masters" (Akif & Majeed 2012, p. 2).

As the authors of Scrum put it, Scrum is "lightweight, simple to understand" yet "hard to master" (Schwaber & Sutherland 2017, p. 3). This can manifest itself in agile teams where they "often do not have the adequate resources and have difficulty finding a sustainable rhythm while avoiding excessive stress for the individuals. Managers can lack the training to coach for autonomy" (Stray, Moe & Hoda 2018, p. 3). Therefore, different views on training, lack of support for it and the inability to properly coach for autonomy are problems a self-organizing Scrum team might face.

3.2. Theoretical Overview and Summary

Based on the review of the literature, this study has identified 15 challenges that could threaten the self-organizing nature of Scrum teams. The challenges discussed in the literature will then later be compared to the empirical data collected from practitioners of Scrum across various organizations. Below is an overview of the challenges displayed in the literature:

Table 1

Challenges	
1. Support from the Organization	2. Lack of Trust
3. Task Dependency	4. Negative Perception of Scrum Events
5. Geographic Issues	6. Multiple Projects
7. Personality Clashes	8. Division of Tasks
9. Drop in Quality and Technical Debt	10. Lack of Valuable Estimations

11. Lack of Individual Accountability	12. Delayed and Changing Requirements
13. Excessive Documentation Requirements	14. Different Perceptions of Communication Software
15. Varying Stances on Agile Training	

4. Methodology

In order to collect appropriate data to come up with generalizable solutions to impediments for self-organizing teams, expert knowledge needed to be accessed. In this study, data is collected from expert experiences through interviews with Scrum masters, followed up by a quantitative questionnaire directed towards Scrum teams as a whole. This section explains the conscious methodological choices and decisions made, interview and questionnaire design, the selection process for interview participants, and a critical analysis of the possible impact of the chosen method upon the gathered data. We will also outline how we aim to analyze the data.

4.1. Research Approach

In order to fulfill the purpose of the thesis, a theoretical gap was identified in order to define problem areas and direct the study towards relevant data collection. This suggested that a deductive theoretical approach was to be a suitable fit for our project. This is something that is in line with research question 1 (1.3), which relates to comparing the challenges in literature with the experiences of Scrum practitioners in reality. The approach led to the identification of challenges present in existing research and allowed for a relevance-focused interview process. For the purpose of not biasing interview participants, and therefore affecting the validity of the experience-based data, questioning needed to be executed inductively. The contradictory nature of needing both approaches, while being dependant on them not affecting each other at the expense of data validity, demanded a non-biasing method and a balanced interview structure that allowed for both approaches to be present.

Using a modified version of *Grounded theory* (GT) was deemed appropriate since "Grounded Theory is being increasingly used to study the human and social aspects of Agile software development teams" (Hoda, Noble & Marshall 2011, p. 611), thus being present in related research, where "the GT researcher uncovers the main concern of the research participants and how they go about resolving it" (Hoda, Noble & Marshall 2011, p. 613). This aligns with the ambition of collecting unfiltered experience-driven data for parts of our research. Since the study needed to be focused, and thus required some elements of initial research, it was not completely aligned with "the distinguishing feature of the GT method [which] is the absence of a clear research problem or hypothesis up-front" (Hoda, Noble & Marshall 2011, p. 613). Although no clear hypotheses were defined regarding what main concerns might be present, problem areas *were* identified prior to the interview process' start. This means that, although the project started deductively and ultimately is theory-driven, elements of GT have been present throughout the process, to allow for incorporating new findings along the way.

Specifically, the intended outcome has similarities with the GT approach, since our specific line of questions asked for *experiences*, although guided by predetermined topics, and the interpretation was left to interview participants and questionnaire respondents. Collecting the accounts of Scrum masters and allowing them to offer recommendations to the challenges without biasing them too much was an important goal in answering our third research question (1.3) about recommendations.

Since the main focus of both interviews and the questionnaire is centered on the deductively predetermined topics but presented to participants with allowance for inductive elements, the method used in the paper is ultimately *Mixed method* and not GT. The approach is described in (Caracelli & Greene 1993, p. 195) as follows:
"Mixed-method designs are defined as including at least one quantitative method (designed to collect numbers) and one qualitative method (designed to collect words), where neither type of method is inherently linked to a particular inquiry paradigm or philosophy."

Utilizing the GT approach to the fullest extent would have meant adapting the questionnaire after the data found in the interview process, which was not done. This was ultimately for the purpose of keeping the study focused. Although data collection was driven by deductively determined topics, elements of inductive research were still present in both interviews and questionnaires, allowing for inspiration to still be taken from GT into the *Mixed method*. This is mainly in regards to data treatment and the possible generation of recommendations based on the data findings. This allowed us to also collect data that relates to the frequency of the challenges in reality and the severity of their impact on self-organization in Scrum teams, which is the purpose of our second research question (1.3).

4.2. Interview Design

To accommodate both inductive and deductive approaches, the interview structure was designed into two parts. After an initial introduction about the purpose of the study and the intention to use the interview participant's experience for identifying challenges and solutions to mitigate them, the first section of the interview commences with a series of semi-structured and open-ended questions (see Appendix 1). The questions aim to direct the interview participant as little as possible and the participant is encouraged to expand on his/her own answers based on the topic of challenges and solutions. No challenges from the literature are presented or shared with the participant in this section of the interview.

The second part of the interview consists of the most central topics identified as problem areas in existing research. Given that the topics are relevant, as is the intention, the expectation is that the interview participant will touch upon the topics during the first

part of the interview. The more topics covered by the interview participant themselves, the more impartial the data can be perceived to be. After checking off topics covered, the second part of the interview lists topic items, one at the time, clearly stating them as present in research. The interview participant are asked if they have experienced the issue, if they can elaborate on its impact on self-organizing and what solutions might be relevant in order to mitigate the issue.

In order to generate a quantitative overview, some shorter quantitative questions about team composition and the level of training and experience are included. The data from those questions are shown in section 5.1 The ambition is to be able to identify possible variables that can affect the data. Along with quantitative data from questionnaires, coinciding data might lead to an improved understanding of what challenges are being experienced and what solutions might be supported enough to be generalizable. All participants were recorded and gave their consent for that on record. They were informed that all data would be treated anonymously unless the opposite was clearly stated - as was the case with one interview participant who consented to being quoted by name. This refers to P9 (Agile Coach Fredrik Wendt). See Appendix 1 for the full interview design.

4.3. Questionnaire Design

To be able to get a quantitative overview of the topics but also a more extensive coverage, both in terms of location and team role, the questionnaire was designed around the 15 specific topics found in the literature research. The questionnaire presents each topic with a short description, where deemed necessary, and asks the respondent to rate the frequency and severity of each challenge with respect to self-organization, from 1-5 (respectively: *Never-All the time*, *Not at all-Significantly*). In addition to each topic, the respondent was encouraged to provide a solution in free text based on their experiences (See Appendix 2). The free text was made optional as a conscious choice to increase the chances of respondents participating from start to finish. This was done since the questionnaire, with 15 topics presented, would take well over 20 minutes to complete if all questions were mandatory. Two of the topics were adapted using the GT approach:

- *The agile training* section asked about the importance of agile training for self-organization according to the questionnaire respondent, and the perceived importance of it within the respondent's organization. This change — as opposed to asking about challenge frequency, severity, and recommendations — was made since the comparison between the individual and the organization was brought up in the interviews, thus suggesting a comparison to be more useful than measuring frequency and severity. This challenge was never brought up in the inductive part

of the interview, and considering GT influences, researching it in a different way was suitable.

- *The communication software* section asked about what tools the respondent use, and the perceived usefulness for self-organization. This was also changed like the previous section due to not much useful data being generated from the interviews. The interviews were focused on Scrum masters, which possibly affected the outcome. The questionnaire opened up the possibility of other team members participating and the possibility of different stances being stated by different roles.

The questionnaire starts out with questions about the respondent's role, years of experience, the country where they are employed, team size and whether the respondent has some sort of formal agile certification. These indicators helped analyze the data and understand the backgrounds of the respondents (see section 5.2).

The respondent was also asked to rate how closely they follow the Scrum framework and whether they have incorporated tools from other frameworks in their work process, similar to questions in the interview. Respondents who were not directly within the Scrum team were asked to indicate this. The reasoning about this will be further explained in the data selection process in the following section.

4.4. Selection Process

Interviewing Scrum masters specifically was an appropriate course of action since the role is tasked with identifying and mitigating challenges on behalf of their team. Therefore, Scrum masters should be the most familiar and acquainted with the topics covered by the interview. Random searches of websites for possible companies that might practice agile and Scrum could have proven to be too time-consuming and not feasible. Therefore, another approach was used as a complementary measure. Using job searching and connecting tools such as LinkedIn, companies looking to hire Scrum masters could easily be identified, suggesting that they practice agile and make use of Scrum masters. These searches were based on common nominators and location foremost, leading to a higher concentration of individuals in the vicinity or in other ways connected to the already existing network of the searcher. The determining factor for a potential candidate was, therefore, based on title and work experience primarily, and accessibility secondly. No conscious choices about gender, age or nationality were made. Industry plays a small factor but only due to the fact that some industries are more suited to use agile methodologies and software development frameworks, and therefore, more likely to be relevant for the study. The selection method generated 13 interview participants.

The questionnaire selection aimed for a wider scope to improve the chances of representative sample size, but also to allow for other team roles other than only the

Scrum master. All recipients were therefore asked to share the questionnaire with team members, as well as peers or other relevant contacts, regardless whether the recipient was a Scrum master or not.

The recipients of the questionnaire were reached in three ways:

- **Company selection -** To get a wide geographic scope, the LinkedIn approach (described above) was used, searching for employers using Scrum, with the search word *Scrum* sorted by specific countries. Approximately 100 companies each were contacted in Sweden, Germany and the USA, 50 in the UK and India, and 30 in Denmark amounting to 450-500 contacts initiated in total using this method.
- **Community posts -** Scrum practitioners tend to follow online communities using forums, slack groups, and other online meeting places. By posting our questionnaire and asking for participants using these channels, the technical number of reachable, possible participants, amounted to up to 10.000 or more. The expected outcome in terms of response rates was significantly lower.
- **Individual contacts -** Utilizing previous connections from the interview selection and using Web forums keeping registries of people certified within the Scrum framework provided the most direct method of contact. The individual contacts initiated using this approach amounted to 350-400.

The large total scope of initiated contacts was adjusted after the realization that the questionnaire itself was too long, which might reduce our potential return rate significantly. This was a conscious choice where quality was prioritized, and a large scope hopefully could provide some compensation. The benchmark for the lowest questionnaire sample size acceptable for data analysis was set to 30, with hopes of reaching at least double that number. That was achieved with 65 respondents answering the questionnaire.

These approaches provided less control over who participated in the questionnaire as many contacts were not directly initiated. Therefore, the specific question asking respondents to indicate whether they were *not* directly within the Scrum team was added.

4.5. Data Presentation and Analysis Method

In order to make sense of the data collected, as well as present it in a comprehensible fashion, choices about analysis execution needed to be made. These choices relate to factors such as the order of presentation, conclusive sections to provide reader overview as well as coherent and uniform ways of extracting the practical recommendations.

Ordering of the data presented on each challenge has been done based on the perceived severity of the impact of the challenges on self-organization, experienced by the participants. Severity has been determined by the total score of severity percentage

from the questionnaire data and an estimated number of 25/50/75 based on perceived severity in the interviews. The most severe challenge had a total score of 168 and the least had a score of 46.5. This is excluding the two challenges that were adapted in the questionnaire, thus yielding uncomparable percentages. Since both were perceived as less severe in the interviews as well, both are presented last.

The data presentation from each challenge has been divided into sections describing:

- **Interview Data -** This is presented as a condensed selection of the main topics covered across interviews and summaries of the most prominent viewpoints.
- **Interview Analysis -** This is a condensed, participant-led discussion regarding the challenges, showcasing the different interpretations of the challenge to provide analysis of it. In all cases but two the analysis is summed up by conclusion headlines regarding the dominant understandings of the challenge.
- **Questionnaire data and analysis -** This is a participant-led discussion based on quantitative data and qualitative responses from the questionnaire showcasing the different viewpoints about the challenge. Quantitative data is presented in a table with coloring to highlight frequency or severity from low (red) to medium (yellow) and high (green). Each table has a summarizing number called *majority share* highlighting respondent preference, color-coded to show where the challenge falls on the scale. Even the percentage and color-coding is not always a literal majority it represents the dominant trend in the data. In two instances where questionnaire data provided more extensive discussion, the summarizing conclusions for the challenge were presented in this section.
- **Recommendations -** Both qualitative and quantitative data are used in order to generate practical recommendations, inspired by the GT approach.
- **Discussion -** In Section 7.1. the findings from each challenge are discussed in relation to the literature review to showcase the alignment, or lack thereof, in regards to the first research question (1.3).

The data presentation is centered around the participant's perception and their experience of the challenges. Interview and questionnaire data analysis firstly focus on whether the challenge is experienced and to what extent. Its extent has, when applicable, been analysed in terms of frequency and severity in order to answer research question 2 (1.3). In relation to the analyses of both data sets, outcomes and reasons behind the challenge are highlighted in conclusion points, located in the section with the more extensive analysis. Lastly the analysis generates recommendations based on explicit advise found in data, either summed up from different responses or described by direct quotes. This concludes the challenge analysis section and presents an answer to research question 3 (1.3).

4.6. Method Evaluation and Challenges

This section will evaluate the method choices in the research and their possible effect on the result. Considering the amount of data generated from the *Mixed method* approach, it can be seen as a successful method of data collection. In terms of qualitative data, both the interviews and questionnaire generated far more than expected in the 13 interviews, and 40-57 free text answers per questionnaire topic. The quantity of 65 questionnaire participants exceeded the benchmark of 30 respondents as the lowest acceptable sample size, thus leading to improved data reliability. The sample size is enough to give indications on percentiles, but the main outcome is the qualitative answers, although the size of the qualitative data exceeded the expectations. Many respondents offered extensive answers and detailed recommendations to a greater extent than expected. Although offering solutions for challenges was made optional for respondents, the majority chose to participate in answering those optional questions.

The interview data show no apparent differences compared to questionnaires due to being less diverse. The interview participants were mostly Scrum masters/agile coaches operating in Europe whereas questionnaire respondents were more geographically dispersed and had different roles in the Scrum framework. An apparent difference between questionnaire respondents and interview participants was slightly lower experience levels on average for the latter party. Differences in background and experience could have affected the results causing misalignment between the questionnaire and the interviews when it comes to the severity and frequency of the challenges. This was not the case.

As intended the interviews focused on primarily Scrum masters, which might focus the data gathered around their experiences (as explained in 4.4). The questionnaire did not reach the desired level of diversity when it comes to roles, as Scrum masters again were heavily over-represented with 71 %. This percentage might be somewhat distorted by the fact that respondents had the possibility to select several roles. This was intentional, as it is not uncommon for members to have more than one role, but it affects the certainty of the data. As a result, the role differences were not taken into account in the questionnaires, but all answers were treated equally, except for specific cases in the analysis where data might be particularly affected. A possible reason for this, apart from Scrum masters making up the bulk of people contacted, could be that the Scrum master's task is to remove or prevent impediments to the team's work process. Thus, participation in a questionnaire might be seen as just the type of impediment to remove as it does not bring value to the team or the organization directly.

The selection method for questionnaires did generate more than the expected scope in terms of geographical diversity, with representatives from each continent

participating. Although the study does not focus on that, in particular, there have been no apparent differences detected in terms of the respondent's geographical location. The selection method of contacting companies was ended prematurely, not because the selected countries were deemed to be enough, but because of low return rates using that method. Switching to the other methods presented allowed for better return rates, and allowed for attempts to re-adjust the scope to include all parts of the world. The method of using Scrum forums and certified registries led to an over-representation of certified respondents (77 %), which might affect the data collected. This might mean that the average knowledge of respondents in the study might not correspond to the average knowledge of active Scrum masters in reality. This higher level of knowledge and theoretical overview that respondents possess is, however, beneficial since it adds more substance and validity to the recommendations gathered.

Although quantitative data could have benefited from even more participants, the combined quantitative data of interviews and the free text answers from questionnaires are more than sufficient in terms of reliability. It became apparent in the questionnaire data, that it was not clear at what point in time respondents were expected to refer to, which led to sometimes differing results. This could have been made clearer in the design. Some challenges were not ideal to measure by frequency, due to being spectrum-based, which also might have distorted the data.

A potential issue with interviews and questionnaires alike was the fact that questions were not mandatory. This was a conscious trade-off as a result of the questionnaire size, but even in interviews, a concise answer was not always achievable. This was due to the semi-structured approach, and even with conscious guidance by the interviewer towards more conclusive answers, the flow of conversation and time limitations affected this a few times. This was compensated by the quantity of qualitative data derived from the questionnaires and led to conclusive answers being achieved through the *Mixed Method* approach.

The challenge Division of tasks (3.1) was originally called *Proper Task Allocation* in the questionnaire which stirred some upset comments suggesting some unclarity as a result of the wording. The wording was intended to refer to allocation, both to self or amongst the group, but some respondents misunderstood it as a *top-down* allocation of tasks, which is not at all in line with self-organization, hence strong reactions. This may have affected the data.

One concern for the questionnaire, in general, was the focus on self-organization, and the risk of that focus being lost while answering questions relating to both technical and relational aspects. This happened occasionally in the interviews, but then with the interactive possibility of reminding the interview participant, it was not an issue. This was taken into account when formulating the challenge descriptions but still remained a

concern. Fortunately, respondents kept the self-organizing focus more than expected throughout the questionnaire.

Lastly, as section 6 will show, the GT-inspired aspects provided some benefits in reaching the thesis purpose of extrapolating advice from experience data to provide for the Scrum community. One challenge of the GT method that we tried to avoid by using a deductive approach, is the difficulty of separating challenges from each other. Even though this was achieved to a great extent in the literature review where 15 challenges were distinguished from each other, the lines between them were again blurred in the data, especially when it comes to causes. This provided a challenge in sorting them but also provided insights into the possibility that the different topics were not always separable. This will be discussed further in section 7.2.

5. Presentation of General Data

This section reports the general data from interviews and questionnaires. This collection of data does not tie directly into the 15 specific challenges but provides outlines and participant backgrounds that might provide additional insights relating to the challenges analyzed in Chapter 6.

5.1. General Interview Data

This section gives an overview of our interview participants. The table below depicts their distribution. Especially relevant is team size and certification since those factors can influence the ability to be self-organizing and the average know-how of the interview participant when it comes to the Scrum framework theory. The following sections (5.1.1. & 5.1.2.) review the overall findings of the interviews before diving into specific data presentation and analysis of the challenges.

Table 2

Interview Participant	**Role**	**Team Size**	**Country**	**Industry**	**Years of Scrum Experience**	**Scrum Certified**
P1	Scrum Master	6	Sweden	IT & Security	2	No
P2	Scrum Master	5	Sweden	Business Intelligence	5	Yes
P3	Scrum Master	4 & 7	Germany	Furniture	2	No
P4	Scrum Master	12	Sweden	IT & Security	1	Yes
P5	Scrum Master	6-7	Sweden	IT & Security	10	Yes
P6	Scrum Master	4 & 9	Sweden	Digital Book Publishing	2	Yes
P7	Scrum Master	4, 5 & 5	Germany	IT & Video Management	9	Yes
P8	Scrum Master	14	Sweden	Furniture	1	No
P9 F. Wendt	Agile Coach	various	Sweden	Agile Coaching	15	Yes
P10	Scrum Master	5	Spain	Automation	3	No
P11	Scrum Master	8	Sweden	Bank	1	No
P12	Scrum Master	7 & 8	Denmark	Transport and Logistics	3	Yes
P13	Scrum Master	6	Sweden	Consultancy	3	Yes

5.1.1. Definition of Self-organizing Teams

All interview participants were asked to provide their own definition of a self-organized team. An overwhelming majority placed emphasis on the decision-making autonomy of such teams. An interview participant referred to the self-organizing team's ability to set boundaries for themselves in order to choose what they can or cannot work on. This echoes the definition of another participant, Fredrik Wendt, a Scrum.org instructor and agile coach who chose not to remain anonymous. He referred to the self-organizing team saying that:

> *It would primarily be that the team members themselves organize how to collaborate, how to accomplish something and how to organize their work. That could be how to organize their workspace, how to organize their time, how to organize who does what, or in what way they do it* […] *self-organization is all about no one else telling them how to accomplish some usefulness.*

Many others have directly referred to the idea of no managers directing the self-organizing team or dictating their decisions. Almost all participants stated that in self-organizing teams planning and strategizing happens within the team as opposed to being given a plan devised by the top management. Thus, the freedom to make decisions and choose how to work best and deliver value was the most common theme in the definitions.

The second most common theme in the definition was the idea that challenges and problems are solved in the team and by the team. An interview participant placed emphasis on the notion that self-organizing teams should "come up with different ways to unblock themselves, even when it comes to technical problems, rather than wait for somebody else to push them."

The final common theme in the definitions provided by participants was the shared accountability that self-organizing teams are characterized by. Several participants referred to the fact that team members should be able to pick up tasks among themselves and that the accountability of the outcome should belong to the self-organizing team. In essence, most participants echoed an interview participant's definition of a self-organizing team as they identified it as follows:

> *A team where pretty much any member of the team can take up any task whatsoever and can do the whole chain from problem identification or development opportunity identification to code requirement analysis and developing testing, then finally implementation and reviewing. It is a team where we do not actually require a lot of outside help and we can deliver the things that we together with business owners feel are important.*

Summing up, an overview of the participants' definitions revealed that the three most crucial elements of a self-organizing team are:

1) **Autonomy to make decisions:** Planning and deciding how to work is a function of the self-organizing team that should not be threatened by the organization's top management

2) **The autonomy of problem-solving:** The team has the tools to resolve conflicts and technical difficulties and is capable to address such problems

3) **Accountability belongs to the whole team:** The team is responsible for the outcomes of its choices and thus they fully own their work process.

Finally, to sum up, A participant provides an analogy that summarizes the way a self-organizing team process looks like when they say:

> *There is a really great metaphor of the self-organizing team. And it's basically if you look at a roundabout, and how the roundabouts work, there are no traffic lights, they're just like, the cars are actually just like self-organizing to get in and out the roundabout, instead of the red and green and yellow light, and someone telling me what to do. That's where the accidents actually usually happen.*

5.1.2. Scrum Framework and Reliability Considerations

All interview participants were asked whether they have Scrum elements or artifacts that they have modified in order to suit their work process better. They were asked if they removed any of the Scrum events or artifacts and whether they adopted other agile tools or frameworks. This was done to check to what extent the teams are using Scrum and whether they are using a version that is too modified or restricted. The overwhelming majority kept all Scrum artifacts and events intact while adapting their lengths in ways that are in line with the Scrum framework. About half the participants have mentioned that their work process is a combination of Scrum with Kanban elements.

Very few participants' teams were observed to have elements that go against Scrum such as multiple product owners or an internal team leader assigning tasks. However, apart from one team that removed sprints in favor of a Kanban style, the overwhelming majority of participants aligned themselves with the Scrum events and artifacts without serious violations of the framework.

It is noteworthy that only a few participants are working in an environment where the organization has adopted a scaled version of Scrum. This means that the majority of participants have experience with facilitating interactions between their Scrum teams and other non-agile parts of the organization. This is significant in understanding the reason behind and magnitude of a number of the challenges discussed in chapter 3.

5.2. General Questionnaire Data

This section presents the questionnaire data not referring to the specific questions, but rather the respondents' situation and pre-existing factors. Reviewing the data of the respondents themselves might shed light on the reasoning behind the results.

Role within the Team

The total role selection adds up to 81, which exceeds the total amount of respondents (65). This is due to the opportunity to choose several roles, as a Scrum Master might also be a developer or business analyst. 65 responses indicate the roles of Scrum Master (46), Developer (14), and Product Owner (5), the more traditional roles of the Scrum team.

Approximately 28 % state that they are not directly within the team itself but rather someone overseeing the process. 6 respondents directly indicate some form of coaching role (Agile, Scrum trainer) and 5 a more managerial role (Product, Operational, Engineering, Quality, Devops).

Team Members:

Team Size	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**18**	**20**
Number of teams	1	5	10	13	19	15	20	8	5	2	3	1	2	2	1	1	2

Total number of teams: 113 (excluding 7-10 answer)
Number of teams within 3-9 members: 90 = 80 % marked in green (excluding 19/3 and 7-10). Inconclusive answers: 19/3 & 7-10

Years of Experience:

Total: 64

Years of Experience	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**20**
Number of respondents	3	7	11	7	11	5	4	3	1	5	2	2	1	1	1

Inconclusive answer: 72

Scrum Certification

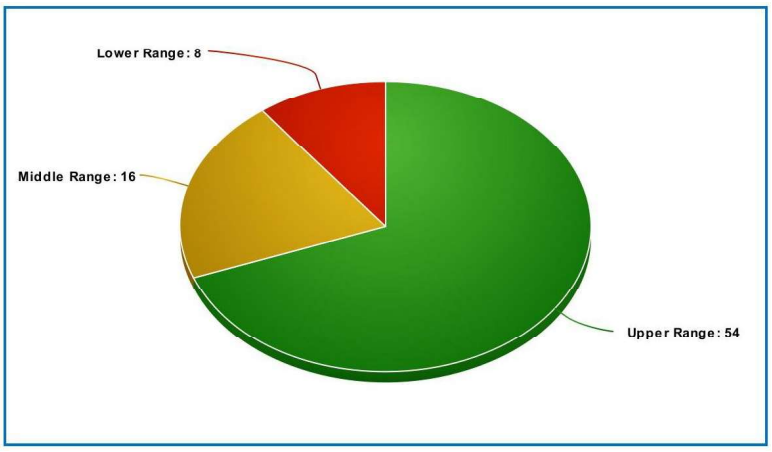
Total: 61
No: 14 (23 %)
Yes: 47 (77 %) – Mainly varieties of PSM and CSM

Representation of Participants

Europe	#	North America	#	South America	#	Africa	#	Oceania	#	Asia	#
Belgium	3	Canada	1	Brazil	1	South Africa	1	Australia	1	India	6
Denmark	4	USA	11							Vietnam	1
Germany	8										
Malta	1										
Norway	1										
Romania	1										
Sweden	14										
United Kingdom	1										

Following the Framework

Upper range: 35 positive with biggest respondent group of 30 on *Considerably* (52.63 %)
Middle range: 14
Lower range: 8 Negative with only 1 respondent indicating *Not at all*



Only 14 (14 %) works exclusively within the Scrum framework
The biggest framework influence from Kanban with 48 responses (74 %) with XP as the second option with 26 (40 %).

6. Results and Data Analysis of Challenges

This section will list all challenges in the same order as in section 3, report and analyze interview and questionnaire data for that specific challenge, one at the time. One core issue here is to determine whether we have found in literature is an actual problem in reality. We also evaluate the severity of the problem and whether our findings correlate with the claims in the existing theory. Every challenge section will end with practical recommendations based on participant experience.

6.1. Challenge 1 - Support from the Organization

Interview Data:

Most participants brought up this challenge on their own without being directed in the second section of the interview. An overwhelming majority have reported experiencing this challenge directly or at least had comments that reflect a lack of top management support.

No participants reported a lack of familiarity with this challenge. This means that every single participant was, at some point, in a situation where they felt their Scrum team is not being supported enough by top management in order to reach autonomy and self-organization. This makes this challenge the most discussed and the most common out of all the challenges in this paper.

Interview Analysis:

The lack of support from the organization is a challenge that manifests itself in various ways according to participants. There are also a number of reasons that possibly explain why a Scrum team is not being supported by the organization to be self-organizing. One way it shows itself is in the organizations disrupting Scrum events and thus threatening autonomy. A participant described this saying:

> *Sometimes managers come by and they ask a question. And since they are managers, it's not really a question. It's more of a command and that's very disruptive to the process when you already have agreed* [on it]. *And that goes for project managers as well. And that's a challenge especially for me as a Scrum master. I need to be firm and say: 'this is the process, we have to respect it.' We can change it, absolutely, but we have to agree on what we want to do.*

On the other hand, another participant referred to the lack of support showing when organizations fail to allow Scrum teams to own their decisions. They say that: "when you come higher up in an organization, they need to know what everyone is doing. So people

need to report things and suddenly when you report, it becomes a manager that decides and sets things and teams are not self-organizing anymore."

A common reason behind such behavior has been cited by most participants to be a lack of understanding of agile principles. On an organizational level, the management sometimes does not understand agile or why it has adopted it. One participant referred to this saying:

> *A lot of people have a very superficial understanding of agile frameworks. And that's a problem.* [...] *They say: we're going agile. And then they get people to help them, like consultants. And they can implement some parts of the framework, change names and change roles but there are no signs of any deep reflection going on. There's no behavioral change. There's nothing changing. So they just keep their old hierarchy structures, then nothing has changed. This means they have a superficial understanding of why they are doing agile.*

This means that the outcome is that Scrum lacks one of its core features, namely team autonomy and decision-making power. If the top management still wants to apply waterfall methods of planning despite the team's wishes, this means that self-organizing is hindered. This is emphasized by one participant who said: "one of the biggest challenges right now is that people want to have fixed deadlines for us. That's not the way it's supposed to work in this agile setup." Another participant also offers a similar experience saying:

> *A challenge I face is getting the buy-in from all levels in the organization. I've been in a project where I was responsible for setting up an agile development team. And we did, but then we were required to provide a fixed price and the deadline for when we could achieve certain requirements. So they expected a waterfall project anyway. So that kind of ruins the whole agile idea.*

Wendt's input on this topic was also in line with the notion that it is the biggest challenge in self-organizing. He reflects this when he says: "I think this is the most common issue I see. Some think you can introduce Scrum and a self organizing-team, but still have your own classic leadership style, and all classic types of hierarchies of decision-making." Thus, some reasons that lie behind this challenge are demonstrated as:

1) **A lack of understanding of agile principles and why agile is adopted:** Without understanding agile principles and why the organization is adopting them it will be difficult for the organization to know what it can or cannot do. It will not understand what actions aid the self-organization of the Scrum team and what actions hinder it.
2) **A disruption of the Scrum team's process:** The team's decision-making power is threatened when top management forcibly includes themselves in Scrum events

and attempts to influence the way the team works. This is something that could harm self-organizing.

3) **Lack of buy-in and waterfall Scrum:** This means that the organization is not fully on board with Scrum and therefore still demands rigid deadlines and extensive planning requirements. This leads to a combination of Scrum with waterfall planning which means that the team is not self-organizing at all and cannot reap the benefits of being an agile team that makes its own decisions.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 1	3	14	14	21	10	52 %
Severity	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 1	0	2	2	20	38	94 %

Above half of the respondents have experienced this challenge at a high frequency. In line with the interview data, an overwhelming majority of respondents reported that they believe this challenge threatens self-organizing either considerably or significantly. It is also noteworthy that no respondents at all claimed that this challenge does not affect self-organization. A dominating theme in explaining why this happens has been also in line with the most common explanation in interviews. This is the lack of understanding of agile and Scrum principles. It is described by a majority of respondents and one describes this saying that the "Management needs to be aware of how Scrum works and what the idea and concept behind Agile is."

The other common theme is the fact that Scrum becomes a soulless process when top management does not support team autonomy. This means that the management claims to want to change to agile but remains tied down by their traditional methods of leadership. This is described by one respondent as they say: "If there is no desire to change the culture, Scrum and Agile will be done in name only." Another reasonably stated that "any transformation needs upper management to understand the ups and downs."

Recommendations:

1) Educate top management on agile principles: On this topic, Wendt says:

> *Agile teams need to have enough formal authority and mandate to actually break existing rules. If we don't do that, we're going to hit the glass ceiling pretty soon. And then that's where it stops. So we need to have that formal*

> *mandate. And we start by educating* [the top management] *and changing their leadership style.*

Therefore, the top management must be coached to realize its role within the agile framework. Transparency about what actions help or hurt self-organizing is of paramount importance. This solution could help with a potentially ignorant top management that has good intentions for the business but does not understand that interfering with the team's decision-making autonomy jeopardizes self-organizing and consequently future performance. Supporting this, a questionnaire respondent claims that the "Scrum master needs to be a strong advocate for the team" and needs "to create transparency about how the actions of management are hurting the team."

2) Inspire strong PO's that know when to say no: Talking about this challenge, one interview participant says:

> *It's not easy to say 'no, we cannot do that. It's too much.' So it can be stressful and lots of pressure. And there you need to have a Scrum master. That's one of the Scrum Master's purposes; to help the team to say no and to take some of the hard discussions.*

Therefore, a solid collaboration between Scrum master and product owner was a common suggestion. The Scrum master should ensure that the team decisions and choices of how to work are respected by coaching the PO on saying 'no' and protecting the backlog. Questionnaire respondents reflect this saying that "the team should defend its decisions" and that "The team must decide how to do things. Period." These are defining features of the Scrum framework that the top management must learn to respect and embrace.

3) Inspire open dialogues between the team and top management: Some questionnaire respondents pointed out that the fault could be within the top management but also partially within the team. One respondent commented "Maybe management needs help to understand how they're not helping the teams with what they're doing. And also maybe the teams are not fully equipped to make well-informed decisions. So open the dialogue about these things and maybe play a little decision poker."

Therefore, an open dialogue where the needs of the team and the top management are discussed and made transparent should help each understand the motivations of each party's actions. This should also minimize conflict in the long run if done properly. By securing the organization's buy-in and viewing the top management as an ally interested in value creation and respecting autonomy, the Scrum team will be on its way to being self-organizing and high-performing.

6.2. Challenge 2 - Lack of Trust

Interview Data:

Only a handful of participants brought this challenge up without being directed to it in the second section of the interview. However, many participants conveyed the opinion that it can significantly harm the self-organizing of teams. Almost no participants voiced the opinion that a lack of trust has no impact on the teams self-organizing.

The majority thought that a lack of trust, be it between the team members or between the team and other teams or stakeholders has a significant negative effect on self-organizing. Most interview participants were able to recall a situation or a problem that they could easily attribute to a lack of trust between the parties involved.

Interview Analysis:

The interview data suggest that this challenge has a low frequency of showing up in the work of Scrum teams. However, the opinions of most participants reflect the view that this is a challenge that can severely harm a Scrum team's capability of self-organizing. This goes for individuals lacking trust for each other within or outside the Scrum team.

A common explanation for the severity of this challenge has been attributed to the relatively small size of teams within the Scrum framework. One participant refers to this commenting that "in Scrum, there should be small teams of about three to nine max. And if one of them loses trust in others and goes down in confidence, there is quite a big chance that the whole team will go down in confidence because it's a small group after all."

Another explanation behind this challenge that was commonly referred to was the varying levels of experience and competence within the team. In itself, this is not a negative thing to have. However, paired with a culture that does not enable transparent discussions and inspection of quality, a lack of trust can quickly develop between team members. Another participant describes their work process and how this problem can affect a self-organizing with the following:

> *I think it's a challenge that we as a team need to work on in order to make everyone feel comfortable with discussing quality goals. This is because obviously, you will have team members that are more competent than others. And they might want to take over in certain areas and they might not trust that the other ones will do as a good a job as them. So in that case, it's just about the communication and being able to say: 'maybe I should help you out with this' or 'Hey, could I take a look at this when you're done?' So I think overall it's both good and bad. A little bit of mistrust might be healthy in terms of quality checks. But still, obviously, you need to be able to trust each other and not be forced to feel like you need to double check everyone else's work.*

From this we can track the severity of this challenge to the following points:

1) The small size of Scrum teams: The small teams that are present within Scrum teams amplify the severity of this challenge. One alienated member that mistrusts other team members can be detrimental to the team's overall momentum and ability to self-organize. In short, one team member is already a significant percentage of a Scrum team when it comes to Scrum teams where the team should be 3-9 individuals.

2) Varying levels of competency: The varying levels of competency and experience, paired with different personalities can lead to some team members wanting to take the lead and take over tasks. This goes against self-organizing and agile principles.

3) Inability to balance criticism vs trust: Scrum teams need to be critical of work in order to succeed in making use of their *inspect* and *adapt* opportunities. Being overly critical could cause a lack of trust and a drop of confidence in team members and their competency. On the other hand, being overly trusting could mean the team is not critical enough to have transparent reviews of the work and development of their work process.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 2	7	25	17	10	4	51 %
Severity	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 2	0	5	4	26	28	86 %

The questionnaire results confirm the findings from the interview when it comes to the frequency and severity of this challenge. Half of the respondents of the questionnaire reported that they only rarely faced this challenge in their work or never. On the other hand, an overwhelming majority found its impact on self-organizing to be either significant or considerable.

Many respondents have stressed the effect of this calling it one of the most serious challenges when it shows up in a team or organization. Among these is one respondent who commented that when a lack of trust manifests itself you should "stop everything you do and address this. Trust is among the most important assets in any healthy work environment." Others have commented that it is difficult to resolve this challenge because "it takes time to build trust and create a safe space." Another pointed out the difficulty of having team members that possess "The courage to have an open and honest conversation with one another."

Finally, many respondents emphasized the difficulty of having team members embrace conflict and work out the problems of trust on their own. On this topic, one respondent has commented that "as long as the team has an environment where they are

able to work through trust issues themselves, the only danger is in shielding them from the impact." Therefore, problems of this type could also indicate that the team has a culture of avoiding conflict or a lack of transparency when it comes to issues of mistrust or quality checks.

Recommendations:

1) Get teams to engage in a dialogue with each other: An important recommendation when it comes to cross-teams trust is to raise the level of communication and bring the problem to the surface. This will ensure that teams do not disrupt each others' process. Among the Scrum values is respect and this should help breed a culture of trust between teams. A participant discusses this when saying that "you have to be respecting of other teams and so you cannot say to them 'we need you to do this now.' If you're not committed you lose trust between the teams."

2) Do something fun: A recommendation that a number of questionnaire respondents provided was to introduce fun activities in teams in order to build interpersonal relationships and inspire trust. One respondent wrote that one of the solutions can be as simple as having fun together. They simply add that "playing boule outside on a sunny day can create trust between people!" Another commented:

> *By having activities with the team so that the team gets to know each other on a personal basis goes a long way to getting teams to trust each other. When they know each other on a personal basis and have empathy for one another, they are more likely to become a trusting and high performing team.*

3) Pair and mob programming:

One concrete improvement measure, especially mentioned in questionnaire answers relating to this challenge, suggests taking up tasks together and doing programming in pairs or as a "mob." Fredrik Wendt explains mob programming saying:

> *It means we have one desk with two really big monitors, a keyboard, one mouse and four people. They're working on one problem at the same time together. So this is called mob programming. It's not just about programming. It's about doing the task as a 'mob' and dealing with complex problems.*

This recommendation was also brought forth by a number of questionnaire respondents. It tackles the varying competencies in the team and the problems that might arise when team members have varying levels of experience that cause some to feel the need to take over tasks. Interactions and helping team members coach each other about best practices of coding could help build trust and create bonds between developers.

4) Raise transparency about this and use *inspect* and *adapt* opportunities:

The problem of mistrust persists the more it is left without discussing it. Among the values of Scrum are courage and respect. Therefore, Scrum teams should be brave enough to bring up these issues and face these conflicts. Yet, they must be respectful enough to present it in a way that is not hurtful. A participant commented on this recommending:

> *Teams should be able to bring this up in retrospectives and maybe even daily standups. It's something that can really stop the flow of work and must be handled quickly by talking about it at least. Of course, this should be done in a way that respects the feeling of the people involved with the problem. It's important that you don't hurt trust even more by making people feel under review or attack.*

The importance of raising transparency about this problem was something that many questionnaire respondents brought up as one of the ways one might mitigate this challenge. One respondent wrote that one should "make transparent how this lack of trust is impeding the team's ability to self-organize through highlighting introduced risk and showing how it impacts value delivery." Another stressed that "one can implement and build trust by showing mutual respect and transparency about it."

6.3. Challenge 3 - Task Dependency

Interview Data:

About a quarter of the participants' team have reported that their organization has adopted a form of scaled Scrum. This is important to mention since task dependency is one of the more common problems in scaled Scrum teams. This is reflected in a comment by Wendt who called dependencies as "the single most common limitation for scaling Scrum." However, task dependency was still reported by both teams with and without scaled Scrum. Less than half of the participants brought this challenge up as a threat to self-organization without being directed by the second part of the interview. Almost all participants reported experiencing problems related to this challenge.

Only one participant believed that task dependency does not have a serious impact on the self-organizing capability of the Scrum team. This means that the challenge is something that is quite common and viewed as a threat to self-organization by the majority of interview participants.

Interview Analysis:

A common theme explaining this challenge is a different process and a lack of alignment between teams meaning that the Scrum team is left waiting on others before they can continue their work. A participant comments on this saying "It is very often you're

dependent on other teams. It can be either UX or other parts that need to be done before you can do your work as well. So that's a big challenge."

Therefore, a common problem is a lack of awareness of what other teams are doing or what the Scrum team is currently working on. Another participant referred to this saying "This is a severe problem. Dependencies make it difficult for us to effectively do our planning [...] we sometimes reject issues or reject taking in certain tasks in our sprint because they mean waiting for someone else to do the work. That is not a good way to work." Therefore, a second problem is a general lack of awareness on the part of the organization regarding what the Scrum team is doing and what it needs in order to fulfill a task.

A final common theme explaining dependencies was a clash between the work process of Scrum teams and non-agile teams in organizations that have not adopted a scaled Scrum. One participant discusses this saying:

> *It is tough to be agile and work with Scrum in an environment where there are a lot of other processes and a lot of other dependencies related to others that aren't working agile. It makes it very difficult to be able to actually plan and deliver what you deliver in an agile way.*

If a team's planning and tasks are limited to whatever the organization dictates due to dependencies, then it becomes difficult to call such a team self-organizing.

From this, one can assert that there are three culprits behind this challenge from the data collected:

1) Dependencies are outside the team: This means that the Scrum team cannot define its own process and choose what to work on. Other teams and circumstances control what the Scrum team can choose to execute when it comes to its backlog items.

2) Lack of awareness about the Scrum team's process: This refers to people outside the team being unaware of the Scrum team's process or what the team is currently occupied with. As a result, stakeholders do not understand what the team needs in order to accept a task or a user story.

3) Clashes between the Scrum team and non-agile teams: The difference in the process and work styles between Scrum teams and other non-agile teams in an organization can sometimes cause misalignment between them. Non-agile teams do not always understand the Scrum team's process as it is different from theirs. From that, dependencies between the two can arise.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 3	1	9	24	19	9	45 %

Severity	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 3	3	9	19	22	12	52 %

The data from the questionnaire is considerably in line with the previously discussed data from the interviews. Almost half of the questionnaire respondents experienced task dependency to a high frequency and over half of them believe this challenge harms self-organization either considerably or significantly. The problem of dependencies outside the team is commonly represented in the data which confirms that it is a significant threat to self-organization. A respondent comments on this saying: “I would suggest if an organization's current team make-up involves numerous cross-team or cross-department dependencies, it suggests those teams are not self-organizing at all, and change is needed on a deeper cultural level.”

An additional reasoning in line with this explanation is the idea that the team is not cross-functional to begin with and therefore, self-organization is consequently threatened. This is described by a respondent who says that in such situations “A team might not be cross-functional enough to not be dependent on others outside the team. This impedes cross-functional teams as they might lack the expertise and therefore transparency required to determine the best approaches. It introduces complexities in an already complex environment.”

This means that the questionnaire respondents and interview participants ultimately agree that dependencies outside the team have a serious impact on self-organizing. Achieving cross-functionality is one of Scrum’s goals and it seems to be a prerequisite needed to solve this problem before self-organizing can be a goal.

Recommendations:

1) Attempt to keep dependencies within the team: Many questionnaire respondents recommended to “make sure that dependencies are within the team and change team composition to achieve that.” Another also recommended this saying that if the dependencies are within one team “it just means the team needs to find different ways of dividing up work items to minimize dependencies (Agile tools like User Stories are meant to help with this).” This ensures that the team, in essence, controls its fate and can plan according to the tasks available at hand. Consequently, self-organizing will be less threatened by working that way.

2) Ensure cross-functionality and proper team selection: If the team requires too much help to achieve its tasks and is being constantly blocked by dependencies, it could signify a lack of cross-functionality. Many questionnaire respondents pointed to this and suggested that the selection process for teams needs to improve in order to keep dependencies at bay. One respondent reflected this saying: “Cross-functional teams

resolve most of this issue, the internal dependencies between stories happens sometimes but need to be recognized and planned accordingly." Another questionnaire respondent recommended that you should "change team composition" and adopt tools like pair and mob programming discussed previously in 6.2.

3) Raise transparency about dependencies: According to some interview participants and questionnaire respondents, when dependencies between teams are somewhat unavoidable the best thing to do is to raise transparency. Inspiring active dialogues between teams can help in this respect. "when it comes to cross-team dependencies, there are different ways like having Scrum of scrums were like different teams, the scrum masters can meet together and product owners can also meet together and discuss it." Wendt also discussed this recommending that the individuals responsible for the dependencies should be meeting constantly and discussing how to tackle dependencies. He recommends that one should:

> *put those in the same room instead, and have them talk about what the dependencies are. By doing this everyone becomes more conscious about the dependencies you get. And then you have a culture that supports teams in handling dependencies and a process that can deal with them effectively.*

6.4. Challenge 4 - Negative Perception of Scrum Events

Interview Data:

Almost half of the participants brought this challenge up on their own citing it as a threatening impediment for self-organizing. An overwhelming majority of participants have acknowledged that the perception of Scrum events as a distraction compromises the Scrum team's self-organizing nature.

Only one participant reported never running into this challenge. Also, only one participant reported that the opposite happened. This means that their team members readily embraced agile events and saw them as an escape from traditional product management styles where being overbooked with meetings is more common.

Interview Analysis:

Most participants asserted that a lack of understanding of the purpose of Scrum events is usually the culprit behind this challenge. About half of the participants brought up this challenge on their own before even being directed by specific interview questions about it. On this topic, a participant commented that "overwhelming meeting cultures can really get in the way of the team's delivery and the team's focus" and that some people misunderstand Scrum to be a framework promoting such meeting cultures. Most participants agreed that this is not a highly frequent challenge but almost all of them have

run into fellow team members who do not see value in Scrum events. Discussing Scrum events, Wendt added that

> *Some teams don't understand why they happen. They don't see the value in them because they don't understand the why, likely because if you have daily scrums, and you don't have goals, then why do you really need to collaborate? Why would you really need to collaborate if you don't have shared goals?*

According to many participants, a second common theme that might explain this challenge is related to the attitude and mindset of some team members, commonly developers. One participant commented that "it's not ideal for everyone to need to be that outgoing and have several meetings. Not everyone likes it." Due to being used to sitting and coding, some developers are not as engaged in planning or Scrum events. Another pointed to this stating that:

> *Some especially really good developers are sometimes not so interested in having meetings or Scrum events at all, or they just don't see the point in getting a joint-view of something and coordinating things together. Some really good developers only like sitting and doing what they're good at, which is coding.*

So to sum up, the two key findings and reasons behind this challenge are:
1) Lack of Purpose Understanding: Team members sometimes do not understand the purpose behind Scrum events and thus view them as a waste of time and energy.
2) Inclination towards traditional product development styles: Team members are sometimes inclined to work without planning or without the level of autonomy that is offered in a self-organizing team.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 4	5	23	14	16	4	45 %
Severity	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 4	8	12	11	23	9	51 %

A common theme in the questionnaire was the fact that participants pointed to a self-organizing team's needs to be flexible when it comes to choosing what meetings or Scrum events work for them. One participant commented that his/her team does not view Scrum events as "a strict wall. If a meeting doesn't make sense just don't have it." They continued adding: "For me and my team, Scrum is a bendable tool we use to make things

easier. We are not blindly following it." This coincides with data from the interview participants where a participant commented that:

> *a self-organizing team wouldn't really need a set framework to follow. It would understand the principles that agile wants to teach which are enforced by using Scrum or Kanban. But they should not specifically be locked down to just following such ceremonies. So I would say that a self-organizing team would use the ceremonies that give them value. But they don't follow a set framework just because it's a good framework. They do it because they have a specific need for it.*

Some responses were even as drastic as suggesting that Scrum is probably a poor fit for some teams where the individuals are only inclined towards development work and coding rather than planning. In line with this, some participants suggested that some developers "only care about coding and demanding more causes resentment" which could, in turn, lead to bigger problems threatening self-organizing.

Recommendations:

1) Coaching and Empowering: By far the most common remedy provided by participants in both interviews and the questionnaire for this problem is coaching the team on the value and purpose behind Scrum events. Around 68 % of questionnaire respondents mentioned coaching about purpose and value as a solution to this problem. Teaching the self-organizing team the 'why' behind each Scrum event has emerged as the single most recommended solution. Some have even gone on to recommend reminding the team of the purpose of each Scrum ceremony each time one starts. This recommendation is something that should be of course carried out by the Scrum master as the team member responsible for removing impediments.

2) Make team members participate in the solution and encourage suggestions: Many have also paired the idea of coaching with empowering team members to propose alternatives to Scrum events. Speaking about Scrum events, Wendt comments: "What you can do is to make sure that [team members] understand why we want to do [the Scrum events]. And then I say, if you figure out another way to do it, perfect." He went on to emphasize the importance of making team members feel included in deciding what Scrum events they integrate into their work process. The reasoning is that a consensus on a working style that comes from team member consensus is far more likely to sustain a feeling of ownership and understanding purpose.

3) Refine events according to team needs: Another common recommendation is to remember that an agile team should focus on delivering value and not simply following the Scrum framework. This means that the team should have the courage and commitment to organize the frequency of the Scrum events. Some have even suggested dropping the ones that are not useful to the team depending on their situation. One

respondent recommended a team should "adapt the ceremony timing and content to better fit the needs" while another said that you should maybe "Try to scale down on scrum activities and tweak to see what fits the team best. There is no model that fits everyone."
4) Demonstrate the benefits of Scrum vs. traditional management styles: An interview participant commented that "if implemented properly, Scrum has way fewer meetings than any other traditional organizational form'' and the team needs to see this. Thus, it becomes the Scrum master's responsibility to raise awareness about this and demonstrate this fact to the team. Some have also pointed to training where this specific point is brought up as a possible solution.

6.5. Challenge 5 - Geographic Issues

Interview Data:

A striking consensus from the interview data confirms that this is a challenge, with several participants claiming it to be one of the toughest. The majority advice against being dispersed, or suggest that you should co-locate to the largest extent possible. However, some upsides can come from it according to one participant who claims that "in one way, it was efficient because we had more hours than 24 hours a day. [...] But it was almost always that when it came to work, I had to read up on what had happened before. And you really need to have a good handle over it." This particular team worked across time zones and found ways to utilize time difference to their advantage. The same participant also states: "I think some in the team thought this was working good and other people, no they didn't want to work like this. So you have to find people that think this is a good process and they can work like that."

Interview Analysis:

The most common drawback described from being dispersed was the informal flow of information which one participant describes:

> *Mainly informal knowledge transfer, informal education and the everyday conversations that you have regarding certain parts of development or certain business-related items get lost a lot in the way because of those of us that are in one specific location. I mean, you talk* [during] *coffee breaks, talk* [...] *in the mornings, you don't always talk in meetings and having long session meetings might be a bit exhausting compared to just chatting over lunch or something.*

Another participant seconds this claiming that "nothing beats overhearing someone talking about something across the desk [...] and taking part in the discussion and things like that."

Considering that most participants were mainly Scrum masters, a certain emphasis was put on the difficulty in planning, since it is within that the tasks of a Scrum master to handle scheduling issues, which might be easier with a co-located team.

How the participant described their solutions to this challenge, mostly followed the theme of managing it, but as a necessary evil and in sub-optimal ways. Agile Coach, Fredrik Wendt, also recognize this as a big problem, claiming "it can be just different floors. That's where it starts" and also agreeing with communication being the critical factor.

> *[Communication] is slowed down. Or rather, you don't have access to the high bandwidth communication tools like walking up to a whiteboard and starting to draw. And so that's what you lose. But you might win personal preference. Like 'I want to be able to work from home - I don't want to commute for two hours today.*

To sum up, the 2 key findings and reasons behind this challenge are:

1) **The flow of informal information:** It is too natural a behavior to prefer high-speed verbal communication ahead of using digital tools. It is also very likely that, even if you plan to avoid this, a conversation about work naturally happens, even when not scheduled.
2) **When managed a few positive sides to it can be utilized**: Having dispersed teams, although with a higher strain on communication, can have its advantages.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 5	14	16	13	10	9	48 %
Severity	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 5	4	14	15	16	11	45 %

Reviewing the respondent percentages shows that this challenge is not as common as others, and points to it being somewhat severe, however that is reviewing the percentages alone. Reviewing the questionnaire answers provides a much steeper analysis of the situation. Eleven respondents suggest that graphically dispersed teams should be avoided if at all possible. Others describe being dispersed as unfortunate but necessary given certain circumstances - "Colocation is nice to have when you can find people/competencies close by. But this is not how the world works these days. So learn to live with this minute problem". However, one might interpret this group of respondents in another way. By simply advising not to use graphically dispersed teams without providing some form of a solution the practitioners indirectly indicate that the challenge might be quite severe after all.

The solution-oriented respondents emphasize the strengths of the Scrum framework tools and the importance of communication to handle this challenge. Several answers are technical with a focus on certain tools to use in organizing such as video conference and screen sharing software. The answer with the most consensus, after the over-represented “do not”, is that you should “either go 100 % remote or 100 % local. Do not mix if you want to keep self-organization high”. The core of this notion is, that even if only a few members are located elsewhere, the whole team must organize on the same terms, otherwise, exclusion and information loss is inevitable. Some respondents claim to be working exclusively remotely and seem to have less of a problem with this - “We are an entirely remote company so it is built into our culture. Geographic issues can be huge for companies that do not have it ingrained in their culture and communication methods become critical”. One interpretation of the data could be that teams that work co-located, or only partially dispersed perceive this challenge as more severe than those who have been forced to truly adapt with one respondent claiming that it “doesn't impact a team’s ability to self-organize but rather shows the extra need for it.”

Recommendations:

1) Do not get stuck in the middle: A solution not present in the interview data, but mentioned several times in questionnaire answers, where more teams experienced the challenge, is to completely execute one or the other. If you do not have the possibility to have the whole team co-located, treat the whole team as if it was dispersed. The team will have to adapt as a team and find a suitable form of communication. Having to use the common communication tools even for more casual conversations reduces the risk of shortcuts leading to exclusion.

2) Prioritize building relationships and enable meeting forums: As stated above by a questionnaire respondent, being dispersed might only increase the need for being self-organized. Working in close proximity in a co-located team provides endless opportunities to adapt to each other, and increase mutual understanding of personalities and working styles. Working in a dispersed team, you get less of the natural relationship building that happens by itself, relationships that the interview data indicates as an important pillar in self-organizing. Extra measures need to be taken to reach equivalent relationship levels or more.

A solution described by Fredrik Wendt refers to either arranging off-site opportunities or digital equivalents providing opportunities to meet for relationship building. Other participants and respondents second this, urging team members to meet as much as possible. One should put emphasis on why it is important for all team members to be included and figure out together what communication tools, be they chat/video tools, screen sharing or whatever may be suitable to your particular situation. A solution selected by the team is more likely to succeed.

6.6. Challenge 6 - Multiple Projects

Interview Data:

A few participants pointed to multiple projects as a threat to self-organizing without being directed by the interview questions. Many have suggested that focus suffers when team members are working on unrelated tasks. About half of the participants have claimed that a team with multiple projects could reflect a culture where team autonomy and self-organization are not enabled.

A number of participants reported never experiencing this challenge due to the fact that their team works only on one product or have long product development cycles. Almost all participants agree that multiple projects, at least in theory, can be problematic for a self-organizing team. Only one participant reported experiencing no problems due to this challenge.

Interview Analysis:

One participant discusses this challenge saying:

> *If one person is working on project A and Project B. That's not good. You will always be behind and you double-plan and over-plan and together it will not work. One of the values of Scrum is focus and these distractions that keep coming to the team all the time might the biggest impediment that we have to fight every day.*

The consequences of such a challenge are quite negative and apparent. In their explanations of why this challenge occurs, participants commonly referred to a lack of understanding of Scrum on the part of stakeholders in the organization.
Another participant referred to this problem saying:

> *Our organization often has a lot of projects going on and they want to switch between them. That's a really common challenge. This context switching severely harms the team's performance. If the team naturally adopts context switching on its own that's okay, but when it comes from the outside it's really harmful. I think that's a way of showing that many organizations don't do or understand agile, or they don't really understand the negative results of what happens to self-organizing teams when you context switch.*

Another common theme participants brought up when explaining the reason for this challenge was a general lack of transparency in the organizing. This is described by one participant who said that "the Scrum Master has the job of working with the product owner continuously on the backlog to make priorities apparent to stakeholders. That helps with this challenge."

Therefore, a lack of understanding of the team's internal prioritization process might cause stakeholders to push tasks onto the Scrum team causing it to switch contexts. This is something that could threaten both their performance and self-organizing.
So in a sense, the participants point to two main reasons behind this challenge
1) A lack of understanding of Agile values: The dangers of having split-focus is not something that is clear to stakeholders sometimes which threatens the agility and self-organization of the Scrum team.
2) A lack of transparency when it comes to backlog priority: Stakeholders might not be aware of what the team is doing or why it is doing something. This means they might feel it is not a problem to impose work on the Scrum team leading to context switching.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 6	8	11	15	19	10	46 %
Severity	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 6	9	4	8	20	22	66.5 %

Almost half of the questionnaire respondents have indicated experiencing multiple projects often or all the time coinciding with upper range percentages. Also, a majority of them have described its effect on self-organizing as considerable or significant. This highlights that this is a quite common challenge with potentially severe impacts on the Scrum teams' focus and, in turn, self-organizing. Very few participants have reported never experiencing this challenge which highlights how common and recurring something like this is in the life of a self-organizing team.

This is in line with the findings presented in the interview where a majority experienced this challenge as well. Echoing the interviews, a common theme questionnaire respondents brought up was a lack of a "clear priority between the different projects from the management group." This ties into a lack of transparency and understanding of what the Scrum team is doing. Due to the focus of the questionnaire on finding solutions and formulating recommendations, the majority of respondents did not state a clear reason behind this challenge.

Recommendations:

1) Coach stakeholders on agile values: A number of questionnaire respondents urged Scrum master to inspire " support and respect for Scrum at the C-suite level" to tackle this. One participant wrote that Scrum masters should coach "the organization about the Scrum values and how to pursue them." Another even added that Scrum masters should "educate the ones who are responsible for this and invite the teams to protest it." This is

in line with the Scrum framework since the Scrum master is responsible for protecting the team from disruptions that threatening the Scrum value of focus.

2) Raise transparency regarding backlog prioritization: This challenge highlights a potential lack of transparency in the organization as to what the Scrum is doing or how it intends to self-organize and prioritize tasks. A solution is a fruitful collaboration between the product owner and the stakeholders to show them the team priorities and make them understand why they should respect them. Agile coach Fredrik Wendt commented on his work as a product owner previously and facing this situation saying:

> *We had a list of 15 types of work. So what we did is we made it very transparent and clear that these are the different types of work that can come in. And if this happens, it is priority number one, like customer downtime. If the product is not working, we're going to do that first. And then we made this very clear so we can go talk to stakeholders of those different types of work. Before this, they didn't know about our list and they didn't know how to relate to one another. So when one customer or stakeholder came with a problem and we were already doing something further up on the list, this lower stakeholder got upset. But now when I made this transparent to them that we're prioritizing work this way, I asked: do you understand? And then they understood the process. This way you at least get rid of that tension, frustration, and conflict.*

This means that a proper prioritization via collaboration between Scrum master and the product owner is key. Then it becomes necessary to raise awareness about this prioritization to make stakeholders understand how the team self-organizes, what types of work it can accept and when.

3) Unify context for sprints when possible: A reasonable and common comment that many interview participants and questionnaire respondents gave was "don't do it," "Stop assigning people to multiple teams," "unify focus" and similar. However, sometimes the business needs dictate context switching. In that case, the team might benefit from a strategy of at least unifying context per sprint. This means that sprint 1 can be dedicated to product/task A and Sprint 2 would be dedicated to product/task B, then they rotate accordingly. Wendt commented on this solution provided by him saying: "if you find this as a recurring pattern, perhaps that solution actually makes everybody happy and makes the stakeholders happy. And we can at least focus inside those sprints instead of our focus being split all the time."

4) Break down teams to retain focus: This is a solution that came primarily from questionnaire respondents. One respondent recommended that an organization should "reduce context switching for the team. Maybe split the team into smaller teams." If the competencies in the team allow for a cross-functional second team to be self-sufficient, this might mean two focused teams instead of one team with a split-focus.

6.7. Challenge 7 - Personality Clashes

Interview Data:

The majority of the participants recognized this challenge and it was touched upon by almost half of the participants before arriving at that specific topic in the second section of the interview.

For some, it was evidently an ongoing challenge they were currently working with, described by one participant as "We have a lot of personality clashes in our team, hence the two-day planning" while others approached it in hindsight, as a challenge that has been overcome, described by another participant as:

> *Looking at personal experiences from looking at my own team and the people I'm working with* [...]*, we've found our way eventually. So from our side, we don't really have that issue anymore. I mean, obviously* [from] *communication with other Scrum masters that I know, they have a bigger issue with this.*

Although some did not experience the challenge currently or personally, similarities to the participant description above could be found in most interviews, as all participants presented with the topic, confirmed its existence, regardless if it was a big challenge for them or not.

Interview Analysis:

Given that this challenge is more of a relational than a technical one, different approaches to the cause of the problem were found in the interview material, depending on the interpretation of the question. A common answer tied to personality clashes is described by a participant: "I think the most common problem is that some people want to work by themselves and some people want to collaborate more" and summed up by Agile coach Fredrik Wendt:

> *I think there are two kinds of people, they are either tennis players or* [...] *football players. A tennis player is an individual. They strive for perfection in everything they do because they know they're on their own. And then there are football players* [who] *strive for the perfection of the whole team.*

One participant goes further by claiming that "a lot of egos also come in play. Developers just naturally tend to have this ego and they don't want to also ask for help whenever they know that something is really tough for them."

Interestingly the most commonly referenced cause for personality clashes derives from the Scrum core feature of cross-functionality. To be able to be agile Scrum teams

want to have as many relevant competencies as possible present within the team. But different competencies and backgrounds, the data shows, can also lead to differences in preference and way of approaching a problem. A participant describes "One back-end developer, for example, has a very different way of thinking of things than the front end developers, and he's constantly challenging how we work with the API's and the databases" while another participant describes a clash with team members saying that "they were just too different and I think much of it is related to two people coming from these different business areas".

A closely related factor that also affects preferences and can lead to clashes is cultural differences as one participant explains:

> *We have some team members who are from India who have one way of handling information or just having a discussion. And then we have some from Croatia who have a different way,* [and] *we have some Swedes, who want to have a consensus and everyone needs to be on board.*

Both these diversity aspects are initially supposed to be strengths of the Scrum framework but they can evidently also be challenging to maneuver in order to get the desired outcome.

To sum up, the two key findings and reasons behind this challenge are:

1) **Individual VS. collaborative members:** Certain individuals are more inclined towards teamwork than others and some put their own ego before the team's best interest.
2) **Cross-functionality inviting differences:** The promoted nature of having a diverse competence base within the team also invites diversity in background and preference which can lead to differences of opinion and way of working.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 7	1	18	31	13	2	47 %
Severity	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 7	5	18	14	19	9	43 %

Reviewing the respondent percentages does not show a strong preference either way, in terms of frequency or severity, although the challenge is clearly present. This may be due to two reasons.

One clue lies within mentions in our qualitative data, where many interview participants refer to this challenge as a spectrum rather than one time events. Several interview participants mention the maturity of the team as a determining factor and seven

questionnaire respondents argue on the same line saying: “I believe conflicts are inevitable; the team needs to overcome them to gain in maturity; communication is essential, and formalizing some 'feedback giving sessions' can be a way to help people learn to trust and respect each other.” Three respondents reference *storming*, from Tuckman’s stages of team development (Gilley, Morris, Waite, Coates & Veliquette 2010), specifically as a part of the team process. Thus, the percentages of frequency can be explained by different interpretations of what point in time the respondent should refer to, and that they might interpret it as sliding scale, therefore going strongly for the middle option.

If the representation from the quantitative data regarding severity, translates to the quantitative data, approximately half of the interview participants considered this to be a big challenge, while the other half had either overcome it or experienced it through others. Therefore, one explanation could be that the respondents who experience the problem might deem it more severe than those who did not to the same extent.

The questionnaire answers did not offer much more description of the problem itself but were more focused on mitigating the challenge, in line with the intended design of the questions. Most commonly referenced is communication and coaching, both as a team and 1-on-1 between individuals at conflict, but there is also an emphasis on teaming interactions. Some respondents argue this to be an impediment for the Scrum master to solve while others suggest you should “as much as possible, allow the conflict to resolve itself. A truly self-organizing team will be stronger as a result of navigating conflict themselves.” Some refer to coaching in relation to the Scrum framework and that differences in understanding of the mindset needs to be sorted out, as they can lead to clashes as well.

A rather unsentimental approach clearly stated in seven of the answers is in regards to the team composition itself and that you should “remove toxic individuals” or restructure, split up or in other ways reorganize in order to mitigate conflict.

Also, in contrast, there are several answers arguing in line with the statement that “because there's a lack of conflict, it's hard to be self-organizing. There must be conflict in order to make decisions and get on the same page within the team” and that conflict can be a good thing that actually helps self-organization.

Recommendations:

1) Conflict resolution and Team building: The single most common recommendation for dealing with personality clashes is to make use of the Scrum events to improve the team both in the short and long-term. Using ceremonies such as the retrospective, to highlight conflict and utilize the whole team as a conflict solving resource could help solve the specific conflict but also help increase the team’s ability to handle conflict for the future. There are also several tools suggested such as *4 colors* (My personality 2019),

Belbin's Team Roles test (Belbin.com 2019), *Covey's 7 Habits* (Covey 2004) and the *5 whys* (Serrat 2017), etc.

While the scientific validity of some of such tools can be questioned, they still are useful in providing the team with insights into each other's differences and a common language to discuss, each other's differences. Utilizing the Scrum master role to do 1-on-1 counseling and coaching can also be a contextual solution. One important thing frequently mentioned in both interview and questionnaire data is to work on building the team, not only within the framework and when facing conflict, but in the early stages and throughout the team's life cycle.

2) Change team composition: One solution brought up when faced with severe personality clashes within a team was to exchange and replace members or disband and reform teams altogether. Having the right members in the team does not only refer to competences but also compatibility and mindset. An interview participant emphasizes this saying: "I think that persons who were tremendous developers in other settings might not be the best in a Scrum team. It requires a new skill set." Exchanging members or rearranging teams is a reactive solution to a problem that may have started outside the team, with the recruitment process. Why does self-organization start once already within the team and not when forming it?

Fredrik Wendt, an Agile Coach, presented a solution of self-selection of both team and team members. Since a lot of the data suggest that some members are less inclined to be suitable for this type of team than others, it may lead to more probable alignment if team members actually get to choose to be in the Scrum team rather than be assigned to it. Wendt explains:

> *It's very common in most organizations that people don't select which team they're working in. They don't select themselves. Someone else pushes you into a team. So in those organizations, this is a more common problem. Whereas if I actually selected to be in this team, I'm more likely to also deal with these interpersonal relationships better because I selected them. So the solution to this is to stop pushing people into teams,* [and] *rather supporting people in their own selection of where to work.*

This solution might require some flexibility within the organization as a whole. As the analysis in 6.1. shows, this is not always attainable for all teams. The solution Wendt suggests can also be adaptable to already existing teams that are recruiting new members, or maybe have to replace a member as suggested above. Wendt exemplifies this from his coaching experience with an organization:

> *They had three new people coming in, in a rather short time. But they were not part of the self-selection. So the people that were already employed they selected into teams, and then they selected their buddies. So they kind of all decided for them - 'actually we think we're going to put this person into our team, I think it is a good fit'.*

Although finding tools and ways of building an interpersonally invested team, that may have the possibility to work around these challenges, the data shows that simply not all members are as naturally inclined to be able to do this. Therefore a self-selection solution might provide a better starting point for teams.

6.8. Challenge 8 - Division of Tasks

Interview Data:

Only a few of the interview participants brought this challenge up without being directed to it in the second section of the interview. Despite this, only a handful reported that this is a problem they never experienced. This means that the majority of Scrum teams represented in the study faced problems with selecting proper tasks for individuals based on their expertise or willingness.

The correlation between self-organizing and this challenge was clear to most participants. The majority acknowledged that self-organizing can be threatened if the team is not capable to collectively divide tasks in a way that helps them retain focus and deliver releasable increments at the end of their sprints.

Interview Analysis:

A common explanation for this problem is a lack of cross-functionality in the team leading tasks to go to individuals unprepared or unwilling to take them on. One participant represented this point of view and commented that "the delegation of work is a never-ending issue. But if you have big allocation issues, then you have a team that is not properly skilled or cross-functional." Another echoed this point of view saying that "the dream scenario is having a cross-functional team where everyone can pick up some tasks. That's not always the case." This means that the lack of access to all competencies necessary to deliver the increment could be a reason why a task is being picked up by team members who are incapable of or unwilling to do it.

A second explanation brought up by a participant was that poor estimation practices can lead to the problem of poor task division within the team. He described this explanation of the challenge commenting:

> *Sometimes one person, perhaps a newcomer or someone who has never done anything in that field probably will set a very high number. And a person who has done it a thousand times will set a very low number. Of course, the person who has done it many times will win that discussion because he knows this is easy for him or her. So that actually can be a problem if the task gets allocated to a person who might take a longer time than expected and that can be a negative thing.*

From this we can ascertain that two main explanations could show why division of tasks between team members can be a problem:

1) Lack of cross-functionality: The lack of competencies enabling the team to self-organize could mean that tasks could end up with unwilling or incapable members. The opposite could happen where a team member is too experienced and thus wastes time on a too easy task due to a lack of cross-functionality.

2) Poor estimation practices: Estimations that are not meaningful or helpful to make the teams understand the weight of a task can cause this challenge. If the team does not understand what is needed to fulfill a task or roughly understand how much effort it will need, improper selection of tasks could be a threat.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 8	6	31	11	12	3	59 %
Severity	**Not at all**	**Slightly**	**Moderately**	**Considerable**	**Significantly**	**Majority Share**
Challenge 8	11	17	13	13	9	44 %

The questionnaire reflects the data of the interviews showing that this indeed is a challenge of low frequency. More than half of the respondents have rarely experienced this challenge or have not experienced it at all. An interesting distribution is seen in the opinions of the severity of this challenge when it comes to self-organization. No clear consensus is reached which can come from the fact that a big percentage of the respondents have not experienced this challenge and thus are thinking in terms of theory or opinions, rather than reflecting on personal experiences.

Questionnaire respondents focused on offering solutions to this problem and little was provided on the reasons why this challenge arises. A common theme that reverberates through many responses is the fact that problems in self-organizing could also come about when hard tasks automatically go to more experienced team members. A respondent offered this point of view when they said they believe that: “knowledge and skill silos are a huge no in a well-functioning team. Maybe a person is more suitable for a given task but this doesn't mean that anyone else picking this item will fail catastrophically.”

Recommendations:

1) Emphasize cross-functionality: A questionnaire respondent emphasizes the “cross-functional knowledge sharing during the execution of the sprint.” This becomes possible if the team has all the competencies to ensure many people can tackle any task

the team intends to accomplish. Constant inspection to ensure cross-functionality to revise team composition was another recommendation some questionnaire respondents offered to solve this problem.

2) Encourage learning opportunities: Tasks or user stories that are too difficult can cause a short-term drop in productivity if picked up by junior team members. However, in the long run, it ensures team growth and skill development. One interview participant commented on this recommendation saying:

> *When you have a really tricky story and you have to allocate it to junior developer it perhaps takes a little bit longer for that person to do it. But then usually, that person becomes more of a specialist in that specific version. So going forward, and he's gonna be the experienced developer in that area. So it's a really efficient way to share knowledge, and that everyone can progress and learn new things. So that contributes to everyone getting better.*

3) Pair and mob programming: Related to the previous recommendation, pair and mob programming should help with this challenge by providing less experienced developers chances to learn from others. This provides opportunities for coaching and raises the level of collaboration between team members in the long run. A questionnaire respondent explains this solution saying:

> *This may also be a good thing if you have new members in the team that want to learn more. So instead of A) Leaving the easy tasks for the worst coders or B) Letting the worst coders fail miserably trying to complete complicated tasks, I propose C) Doing pair programming, so the best coders can pair up with the worst coders on complicated tasks, thereby making the worst coders good coders in the longer run.*

6.9. Challenge 9 - Drop in Quality and Technical Debt

Interview Data:

This challenge is among the less represented challenges in the data collected from the interviews. Only two participants brought this challenge up on their own without being directed to it in the second part of the interviews. Almost half of the participants did not experience this problem at all and thus had very little input on the reason behind it or how to solve it.

Only a handful of participants classified this as a problem with a threatening impact on the self-organizing aspect of Scrum teams. More than half of the participants reported that they have measures in place to prevent it and do not find that it harms their ability to be self-organizing in a severe way.

Interview Analysis:

A common theme explaining the threat of quality drop or technical debt is a lack of transparency in tracking such problems. While this challenge was not very frequently discussed, the ones who discussed it agreed that transparency about its existence is of paramount importance. A reason behind this is that the quality goals of the sprint were not as clarified or refined as the team needs them to be in order to produce releasable increments. A participant talks about sprint goals and the threat of having unclear ones when they say: "if you're not aggressive in your definition of them, quality goals can be threatened and you can experience a lot of technical debt."

Thus, the main reasoning found in the few discussions about this challenge and its effect on self-organizing teams was:

1) A lack of clear quality goals: Without consensus on the level of anticipated quality of an increment, a team could be left with a lot of technical debt. This could harm self-organizing in the future since the team will be doing troubleshooting and backtracking to solve errors in previous codes rather than planning for the future.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 9	0	16	27	14	7	42 %
Severity	**Not at all**	**Slightly**	**Moderately**	**Considerable**	**Significantly**	**Majority Share**
Challenge 9	3	14	12	20	15	54.5 %

Just below half of the respondents only experienced this challenge occasionally. Looking at the data we can see that the majority of respondents do not believe that this is a challenge with a high frequency. This is in line with interview data since many participants there reported opinions that are in line with questionnaire respondents.

When it comes to the impact of this challenge, above half of the respondents thought that it can have a considerable or significant impact on self-organization despite its low frequency. This indicates that quality drops and serious technical debt that harm self-organizing are threats that do not arise very frequently but have the potential to derail the process of planning, collaboration and consequently self-organization.

Recommendations:

1) Clear quality goals and definition of done: An interview participant comments that quality goals are only compromised when:

> *You don't have a proper definition of done. The definition of done should dictate your standard of quality. Meeting the definition of done by an agreed DoD as the standard of quality forces stakeholders to understand the cost of work. So if they want too much done in a short period of time, this could mean poor quality.*

Therefore, clear definitions of done ensure that everyone understands how the increment should look like at the end of a sprint. This reduces potential conflicts since it manages expectations as well. Questionnaire respondents have also recommended this saying that the team should "stress that the definition of done has to be met for each submit" and "develop and adhere to a proper Definition of Done."

2) Raise transparency about technical debt: A recommendation provided mainly by questionnaire respondents was to raise awareness about potential quality drops and to make technical debt as visible as possible. One questionnaire respondent described this recommendation saying:

> *The trust between the Scrum team and its Stakeholders (and managers) is crucial. In order to keep the team able to be self-organized, we need to keep a high-level of trust inside and outside the team. If the trust is broken, the "command & control" management style is more likely to happen. My advice: preach transparency (inside and outside the team) and try to remind your teams of the mid/long term goals. It is okay to fail, to miss a sprint Goal, what is important is to inspect and adapt to try to improve.*

3) Coach individuals to prioritize long-term outcomes: A recommendation that is closely related to the previous one is to place the focus on long-term goals. Quality drops and technical debt could be a reason of rushed work due to pressure to meet sprint goals. This could mean that there is a failure to see the bigger picture and how self-organizing can be threatened if the team is only repairing technical debt in consequent sprints. Questionnaire respondents have recommended evaluating situations and prioritizing the future of the self-organizing team. This was emphasized by a respondent who said that a Scrum master should: "Get everyone in a room. The team. Managers. Stakeholders. Come to an agreement that Quality needs to be the top priority for the software we develop and show that the cost of poor quality far outweighs the cost of delay almost every time."

6.10. Challenge 10 - Lack of Valuable Estimations

Interview Data:

All participants recognized the practice of estimations, and all of them use or have used, them. A majority of them claim it to be difficult to do estimations accurately. A participant explains that "I don't feel like they're hard to do if you know they're going to

be wrong. We can just guess and I think that's how it should be. We only do it for ourselves. If a project manager takes our number and runs away, he or she is in trouble" while another participant expresses "it's just a currency, like just make certain you make an agreement in the team, that when you really do an estimate - estimate."

Several participants claim this to be a big challenge, explained by a participant saying "time estimation is difficult. It is really difficult because when you start to investigate. Even if you do prototypes and spikes or whatever, there will always come new things when you start to dive into things that you didn't think about or see" and by another as "I would say that that is the biggest challenge, no doubt. And I mean, some people are more inclined to pick up what the idea itself is. And some people think it's way too vague and not clear enough what an estimation actually is."

Interview Analysis:

A challenge related to estimations, besides them being difficult to use within the team, is to use it to communicate with other parts of the organization. One participant made a representative argument of this:

> *The biggest challenge right now for the companies and organizations has to come from a project-driven organization where you measure everything* [...] *Everyone wants to measure things by time. They want to know 'When can we get this'. And the biggest challenge we see right now is that people are generally quite bad at estimating how much time* [things] *takes. But what we are good at is estimating the relatives, as in the complexity points of it, so how complex this is compared to another specific task* [...]. *I think to change people's mindset regarding measuring it by the complexity instead of the actual time it takes is a really big challenge.*

Although the majority of the interviews report this as difficult, they also describe it as a consistent part of their work process. A participant argues that "if the team spends a lot of time on estimation, then there's something wrong, I think because then the team doesn't really know that the value of their work. A highly skilled team already knows what they have in the pocket". Fredrik Wendt argues "so that part of discovering about the work is sometimes useful. But it's also a premature optimization because you don't really know what you need to do until you start doing it. There's a waste in that process." and expands by explaining that a range is "an honest thing" that is much more likely to be correct, than setting a certain date which, according to him, is "the best way to be wrong." Conclusions about the background of this challenge are presented in the following section. This is due to the fact that more relevant data was presented by questionnaire respondents with regards to this challenge.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 10	2	26	17	14	4	44 %
Severity	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 10	10	16	19	13	4	42 %

Reviewing the questionnaire data, a clear majority claim this to not be a problem. A common approach to answering these questions was to explain the purpose of estimates, best summed up by the answer:

> *Scrum expects estimates to be inaccurate to a certain degree. The purpose of estimation as a practice is to generate the disagreements, thereby drawing out the various perspectives [ideally] represented by the team, which hopefully leads to a deeper understanding of the work item being estimated.*

Another respondent expands on that notion with "Accuracy is not important. Statistical variability (predictability) or non-bias is more important. That being said, if there is systematic bias over a long period of time, the team needs to address this problem".

The majority of the answers that address a solution to this put an emphasis on the intended inaccuracy for the sake of discussion. Alternatively, they simply explain that you should use estimations when there is a need while others claim there is no need for them at all. If this data show that estimations are not a challenge, however, can be discussed. There seem to be a very clear and uniform understanding of the topic, given that many answers are aligned, which does not suggest that this was unknown or unclear to the respondents. On the contrary, judging from four *separate* variations of the answer "#noestimates", it is evident that this topic has been *well* discussed, considering that it is trendy enough for a hashtag.

Several of the answers, after all, treating this as a challenge address this as a maturity issue - something that a team will outgrow given that the purpose is understood and that the outcome of reliability over time is the real goal. This was explained as:

> *This depends on the team maturity: for a starter team, the importance is to get a global understanding, more than calculating precise estimations (therefore I sometimes used T-Shirt sizes for estimation). The main purpose of the estimation should be the discussion around it that clarifies the subject. In a later phase, when the team grows in maturity, estimations can be used to measure velocity, and then it's accuracy can be helpful.*

Considering the respondents' answers to estimation as a challenge, and that this seems to be a maturity issue mainly, it is possible that this has more to do with the attitude towards the challenge than the actual challenge itself. The common description of this challenge as a maturity issue could possibly be an effect of our respondent group. 77 % of the respondents have some or several forms of certification, and therefore formal training as well as several years of experience (see 5.2). This could mean that many respondents have not had to deal with this challenge for some time. A speculation could be that our more experienced respondent group consider this to be a challenge for young or inexperienced teams, and therefore have a somewhat distorted view of the challenge.

To sum up, the 2 key findings and reasons behind this challenge are:

1) Estimations are intended to foster discussion, not immediate accuracy:
The intended use for estimations is not to reach an immediate consensus, but rather to highlight the differences in perception of the task. This allows for the team to make the differences in perception visible by discussing and explaining their view, thus, leading to a better understanding. It is more used for visibility and transparency than for reliability. What it may lead to, given that you as a team use this practice for an extended period, is that you reach statistically useful data from your earlier estimates.

2) Estimations could be a useful tool, but also a challenge, for a younger team:
Given that a new team might need tools to understand each other's work process, and have a better chance of self-organizing, estimations as a means of fostering discussion could be useful. But using estimations, a tool meant to be uncertain, in a setting that already has uncertainties, could be challenging which is why the purpose of estimations need to be made clear.

Recommendation:

1) Make the purpose of making estimations clear:
Explain that estimations are not intended to lead to an immediate consensus, but rather a tool to air the differences you find and make sure to discuss why there are differences. This will lead to a more shared understanding of the tasks. It is also important to emphasize the purpose for people outside the team and try to avoid the fallacy of making estimations about time only.

2) Do the form of estimation that works for your team or do not do them at all:
Different teams prefer different methods, although estimation seems to be a well-known and common practice at some point in a team's life cycle. Discuss the form that works best for your context, be it "planning poker", complexity points, ranges or, as some argue, simply avoid estimations altogether if it is not helpful to you. The important part is shared understanding.

6.11. Challenge 11 - Lack of Individual Accountability

Interview Data:

During the interviews, this was not experienced as a severe challenge by any participants. Most described it as non-existing or easily dealt with. An interview participant described some upsides to the topic:

> *It leads to a discussion about performance in a team. And that's just how we want to address it. The team is the smallest part of the team and the individual is not. And the reason is, is the team or supposed to deliver the increment the definition of done increment. And the team needs to have the skills to do that. And if the team doesn't have the skill to do that, they won't deliver the increment in the sprint.*

Only a handful of the participants brought this challenge up on their own with regards to a team's self-organizing efforts. Many participants have reported that they theoretically understand how this challenge might threaten self-organization in Scrum teams. However, only a few have experienced or reported having run into these consequences as a first-hand experience. Many participants were actually positive about this challenge and found little problems dealing with it in their job as a Scrum master. This point of view was reflected by a participant who described it saying "so far, I haven't really seen a case where something goes wrong, and the team blames a specific person. Usually, it's quite open [...] and it's up to the scrum master to make sure that the team spirit is high enough and the trust level is high enough."

Interview Analysis:

One participant described experiencing this challenge saying:

> *You should act like a team and take praise and responsibility as a team. But of course, in reality, there are always specialists in the team and they will have to deal with things that other teammates cannot relate to. And since they do that, they will always be personally responsible for some things. You should work against it, I think, but it's inevitable. So that's a problem because it's the same thing as not having a team in those respects. If that person leaves the team, it does not have anything.*

This does not really describe the issue of individual accountability but still highlights some dependency issues (further discussed in 6.3) that might cause that type of conflict, although this particular team did not experience it. The interview participants that had anything to say about this challenge all addressed what could be done to solve it, ranging

from team building measures to replacing members not performing up to standard, but not in concrete ways. They seem to perceive this as a natural part of enhancing performance thus not leading to isolated measures for this particular challenge.

Apart from the management factor, most participants referred to the maturity of the team as an important factor regarding this challenge. Conclusions about the background of this challenge are presented in the following section. This is due to the fact that more relevant data was presented by questionnaire respondents with regards to this challenge.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 11	13	24	12	11	3	59 %
Severity	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 11	15	14	21	10	3	56 %

Reviewing the questionnaire data, it is apparent that this challenge is not very common, but when present it can be of great severity for a team described “if this became a problem in a team, it would significantly impact the team by creating friction between team members which would destroy their ability to self-organize.” Most answers seem hypothetical, and not personally experienced, although the severity is still recognized. Only one answer seems to personally experience this - “Devs don't care about committed work. They only care about the absolute best intellectual work they're capable of regardless of deadlines”. One other respondent shares: “usually in these kinds of teams you see "hero behavior", where the usual suspects are being asked to fix things or to think along with someone. If you address that out in the open and raise awareness about the harm that is doing in the long term, it will solve itself.”

The single most referenced concrete cause for this challenge, only mentioned once in the interview data, is top management affecting team dynamics by using traditional incentive systems. One respondent describes that:

> *Like in the old Soviet Union, some managers want to appoint "employee of the month". This is usually counter-productive, as you create a culture of heroes instead of a team culture. So fight this Soviet-style behavior. Instead, make the managers appoint "team of the month". Make the managers understand that culture eats strategy for breakfast.*

Other causes for this challenge, if present, relate to the team’s maturity and are mostly referenced as an early phase problem, soon to go away with better awareness of agile mindsets and coaching.

To sum up, the 2 key findings and reasons behind this challenge are:

1) An opportunity to learn and identify weaknesses: The most outstanding point raised in the interview material was the already existing discussions and countermeasures in regard to performance. These discussions allow for improvement and detecting where the team needs to get better, but no teams experienced that it was a problem.
2) Management as the biggest threat: In interviews hypothetically discussed, and in questionnaires clearly experienced, reward systems and incentive programs were deemed to be the main cause of this challenge.

Recommendation:

1) Pair and Mob programming
Pair and Mob programming (see 6.2) is a way to deal with complex issues. But it can also be a way of co-creation and co-learning that might be suitable for less complicated tasks as well. Solving problems in pairs, or more, allows for a shared understanding and team members can coach each other. The mindset of this recommendation is summed up by a questionnaire respondent - "Promote collaboration among team members. The shared responsibility could be a positive thing if team members care about their peers. Naturally, a team member wouldn't want to let their peer down if they have a good working relationship with them."
2) Avoid individual rewards
Since this is not always a root cause that the team can affect directly towards each other it is of great importance to take a stance against organizational structures that threaten the team. This recommendation is best summed up by a questionnaire respondent:

> *Coach your corporate management to stop pushing individual awards and do more team awards. In the corporate world, individuals will probably always be judged and rewarded (i.e. pay raises, promotions) based on their individual contributions. But if the management will start viewing those individual contributions in terms of how they contribute to the team's dynamic and success, it changes the individual's goals. We also do team rewards such as outings, catered lunches and, gift cards for all when they meet interim goals.*

6.12. Challenge 12 - Delayed and Changing Requirements

Interview Data:

Only a handful of participants brought this challenge up on their own when asked about challenges that threaten self-organizing in Scrum teams. An overwhelming majority have reported experiencing a change or a delay in the requirements of an increment to be delivered at the end of the sprint. However, more than half of the participants have described this challenge as a normal occurrence in the life of an agile team.

A few participants pointed to this challenge as a problem that can derail self-organizing if it is not dealt with properly. On the other hand, some participants identified it as a common recurring theme for agile teams and therefore should not be classified as a negative or severe challenge.

Interview Analysis:

The general vocabulary describing this challenge often included words like "irritating, frustrating or annoying" and similar. According to many participants, teams are very familiar with this challenge. Consequently, it was described as an annoyance rather than a major threat to self-organizing. A participant reflected this when commenting on this challenge saying:

> *It doesn't really affect our effectiveness. I would say it affects our mood and our general feeling in the group more. Sometimes if you're working towards something and then someone comes up and changes it, some get quite annoyed. So it's more of a mood killer than an efficiency killer.*

Another common theme regarding this challenge lies within possible misunderstandings when it comes to working within the Scrum framework. Some participants stated that changing requirements are only a problem for teams that are stuck in a traditional waterfall style of work and view requirements as something that should be defined before work starts. This is something that obviously goes against the lightweight nature of Scrum. Wendt added on this topic that "if you look at the sprint as a promise, then you get a problem when this happens. But if you look at the sprint as a time-limited period for exploration while trying to achieve a business outcome, then the exact requirements are not needed."

A rare problem mentioned by a handful of participants is sprint goals becoming obsolete due to changing requirements that force teams to drop items from their sprint backlogs. On this topic, a participant described this problem as follows:

> *I think what's most frustrating is when you hear about a product scenario, and you are told 'this is really important, you need to deliver it on time.' Then we are all struggling working really hard to meet a tight deadline. And then you suddenly hear: no, this is not important anymore.*

This highlights a problem with top management since it shows they are stuck in a traditional project management mindset where they dictate what should be delivered when. This is something that can definitely threaten self-organizing. This is especially true when it leads to the obsolescence of sprint goals which in turn leads to frustration and demotivation for the Scrum team.

From the analysis, we can draw the following conclusions and understand the reasons behind this challenge in the following points:

1) Changing requirements are an outcome of agile and a feature of it: The challenge is there because of Scrum's emphasis on inspecting and adapting using an empirical approach that encourages feedback and stakeholder input. Looking at it that way, changing or delayed requirements are not a direct threat to self-organizing but rather a regular normal part of it. As a participant puts it: "changing requirements, that's something you have to live with. It's part of agile."

2) Not a big threat to self-organizing but a source of frustration: Changing requirements are a cause for mood shifts and possible frustration in teams. The problem is not merely related to efficiency but also the general atmosphere of the team.

3) Approaching Scrum with traditional project management mindsets: Expecting requirements that are rigid and defined from the start of a project highlights the fact that team members are struggling with agility. They are stuck in a waterfall mindset of project management and thus, the sprint is being mistakenly viewed as a rigid promise to the top management or client, rather than a vessel helping the self-organizing team choose how to deliver value.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 12	3	12	18	21	9	47.5 %
Severity	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 12	16	9	12	16	10	41 %

Participants have reported a relatively high frequency of experiencing this challenge. Almost half of the participants either experience it often or all the time. This cements the idea that changing and delayed requirements are a regular part of agile. On top of that, just below half the participants believe that changing requirements should have no effect on self-organization whatsoever (this was reflected in the free text answers). This highlights the fact that many respondents have stated that when it comes to requirement change or delay "the team needs to understand that this is something that happens, there is nothing wrong or risk coming from it." This is echoed by another respondent who even went on to say that

> *Agile teams welcome changing requirements, even late in the development process. This is actually a feature of Scrum and not a bug. If a team's ability to self-organize is threatened by changing or delayed requirements, I would suggest that the real problem likely lies in corporate culture or something deeper.*

This coincides with interview data where many disagreed that changing requirements are negative. Another respondent even commented that "Scrum is designed for this" in reference to changing requirements. Another commented that "A team's ability to timely respond and adapt to changing requirements, and its ability to take on challenges and learning new things as they progress, is a key benefit of a self-organizing Scrum Team."

Again, only a handful referred to sprint cancellation as a possible result of this challenge. However, the majority of respondents placed this challenge within the domain of the responsibility of the product owner, not the Scrum master. This is reasonable since changing requirements or acceptance criteria should directly affect the product backlog which is owned solely by the product owner. The Scrum master also plays a role in coaching and making sure that changing requirements are properly dealt with. However, most recommendations involve the product owner in one way or another due to the nature of this challenge.

Recommendations:

1) Make the team embrace changing requirements:

An interview participant describes this challenge saying:

> *I think a lot of developers complain too much about changing requirements. I'm a developer myself and I understand the feeling. But they should maybe try to understand why they are angry about a change in requirements [...] If changing requirements threaten a team's self-organizing then it is not very agile.*

In essence, the solution with such a team is for the Scrum master to coach the development team on the principles of agile. A questionnaire participant commented: "This is what scrum/agile does. It should cater to needs being discovered later. This should not be an issue for a team although it can be annoying." Therefore, if changing requirements are a problem this might suggest that the team does not understand the rules of the game and the principles agile advocates for. Coaching them to see requirements as a natural part of agile is one of the most common recommendations provided by many participants.

2) Make everyone understand the "why":

Another recommendation that many participants provided was raising the level of communication between the development team and client/organization so that

a) The development team understands the logic behind the changing requirements
b) The organization/client understands the cost behind the changing requirements

An interview participant adds:

> *If there's a lack of trust between the scrum team and the stakeholders in the rest of the organization, changing requirements could be regarded as almost offensive. But if there's already abundant trust between the team, stakeholders and product owner a change in requirements will not be an impediment, but a challenge for the team. And the team will understand why the change of requirements came. Trust and collaboration ensure that changing requirements won't be an issue.*

This means that a Scrum team that collaborates with stakeholders will understand that all parties are invested in the success of the product. From that, an understanding of the "why" should help the team cope with the changing requirements.

A number of questionnaire participants have, however, pointed to another issue that must be made transparent to the organization/customer, namely the cost of the changing requirements. One participant commented: "Make sure that the client or other parts of the organization know the cost of changing a requirement. Make clear that "we can change this, but then we won't have time to do that." Another commented that you should "always coach clients, explain what the costs of late/delayed changes are." Thus, an active dialogue between the team and stakeholders will help both understand the 'why' and the consequences of the changing requirements.

3) Clear definition of ready: A problem with changing requirements or new tasks that questionnaire participants pointed out is the unclarity of these tasks or a violation of an existing definition of ready for the team. Some have recommended the should not "accept anything that does not adhere to the "Definition of Ready". Another commented that "Definition of Ready [...] helps the team and product owner keep the product backlog relevant and prioritized, making sure new acceptance criteria are agreed upon and stories are ready for sizing." Therefore, by having a clear and agreed upon understanding of what backlog items are ready to be started, the team prepares itself for changing requirements. More importantly, it understands which changing requirements it can take on and which ones they should potentially push to a subsequent sprint.

6.13. Challenge 13 - Excessive Documentation Requirements

Interview Data:

Very few participants brought this challenge up before being directed to it in the second part of the interview. Less than half thought it could constitute a problem threatening self-organizing. A number of participants, on the other hand, asserted that documentation is part of their work and a normal task that has little influence on their self-organization. No participants have identified this challenge as a serious threat to self-organizing. However, still, around half of the participants have reported experiencing running into

this challenge at least once in their various assignments as Scrum masters. A fair number of the participants had a positive attitude towards this challenge and seemed to have a plan on how to tackle it.

Interview Analysis:

A common theme discovered was that most participants' teams simply view documentation as a part of the work and not something extra to it that could hinder them. This is reflected by a participant's comment on the topic when they talk about documentation saying:

> *My understanding is that it's a piece of work to do and nothing but another piece of work that you have to do. If the organization needs it and if you are in that kind of an environment where documentation is very important, then it has to be done. And then you just plan for it like any ordinary work.*

Another common theme was that the Scrum masters understood that documentation in itself is not a problem and does not go against the Scrum framework. Documentation that generates no value and occupies the self-organizing team is the problem. An interview participant reflects this opinion when they say: "It's one of the principles of agile that you should not document just for the sake of documenting. But that doesn't mean you should never document. If there is a need for it, absolutely." Therefore, from the data, the most common reason behind the problem of too much documentation lies within teams seeing no value or logic behind the documentation they need to produce.

The second common theme explaining this challenge was the mindset of the parties requiring the documentation being too aligned with traditional project management rather than agile. A participant goes on to explain this problem saying that an organization behaving like this: "is a project-oriented organization where you have maybe a waterfall mindset of simply documenting everything. That's a tendency and a cultural thing you need to move away from."

Summing up, the two main findings explaining the problem and how the participants relate to it are:

1) Documentation is just a piece of work: A fair number pointed to documentation being a routine task they have to carry out and plan for.

2) A non-agile mindset of the organization: Purposeless documentation could indicate that the organization is emulating a traditional style of project management rather than a product development framework that enables team agility.

Questionnaire Analysis:

Frequency	Never	Rarely	Occasionally	Often	All the time	Majority Share
Challenge 13	14	21	19	7	2	56 %
Severity	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 13	15	14	21	10	3	46 %

The majority of the respondents reported having experienced this challenge either occasionally or rarely. This indicates that it is a low-frequency problem and aligns with the data from the interviews. Almost half of the respondents believe that too much documentation affects self-organization only slightly or not at all. This could be explained by the fact that many respondents have questioned the correlation between documentation and self-organizing. The relation lies within the non-agile mindset of organizations demanding pointless documentation which takes the team away from planning and self-organizing.

However, a great number of respondents took this section of the questionnaire as an opportunity to explain that documentation is not a problem in itself like most interview participants said. One participant said that "documentation does not go against agile. Agile does not discourage documentation, it just encourages documentation when needed and appropriate." Another commented that "a common misunderstanding of Agile is that it eschews documentation. The Agile Manifesto merely states that documentation is valued less than working software, which is not to say that it is not valued at all."

The questionnaire also confirmed that the problem can be tracked to a non-agile mindset of the top management. Just below half of the respondents reported that this could be at the core of the problem. One respondent voiced this opinion saying that requiring too much documentation of Scrum teams "implies that top management does not buy into Scrum as they do not understand it, so the team might ultimately fail."

Recommendations:

1) Include documentation in the definition of done: Wendt adds on this: "you need to document to the level needed. It's just a matter of getting it done. And it should be put as a part of the definition of done to have the right level documentation." Questionnaire respondents agreed with this to a great extent where one said "development teams must work with stakeholders to set expectations under their Definitions of Done to include things those stakeholders may value (such as documentation) with an understanding that such changes may involve trade-offs in scope"

Therefore, by raising transparency about documentation and including it in the definition of done, teams become more aware of it and less distracted by unnecessary documentation.

2) Do not demonize documentation: Another recommendation for self-organizing teams is to be compassionate towards requesters of such documents. One should try to understand the logic behind the request and collaborate to find the right level of documentation. An interview participant describes a story of success using this when they say:

> *We had some new developers starting who wanted to know how we do things. They said: we feel kind of stressed out about not having anything written down. So what we did then was that we took the time to actually sit down, write it down and discuss how our process looks like. So the solution was to sit down and talk about it. And it turns out though we worked in a team for a while, we didn't really know how the process looked like as well. So it was beneficial for both of us.*

3) Coach organizations on agile mindsets and opportunity cost: This recommendation came mainly from questionnaire respondents who proposed the Scrum master should "educate top management and explain the importance of adhering to the Scrum guidelines." One respondent wrote about the opportunity cost of having too much documentation saying: "There is never too much work. Just a shortage of funds. Your problem - you choose. I can write about the functionality or you can have it. Possibly not both." This means that the requester of the documentation should be coached to understand agile principles and see the opportunity cost of demanding too much documentation. The correct level of documentation should be decided with the help of the team and not simply decided by the top management, otherwise self-organizing is threatened.

6.14. Challenge 14 - Different Perceptions of Communication Software

Interview Data:

This challenge relates to disagreements regarding the usefulness of communication software or a perception that such software can threaten the self-organizing nature of Scrum teams. Only a few participants brought this challenge up on their own without being directed towards it. The general perception of communication software in participants was overall positive.

Only a few participants reported that they themselves are unsatisfied with the communication software they use. About half of the participants reported experiencing someone else viewing communication software as something with a negative impact on

the self-organizing team's process. Two of the participants even reported that they are in the middle of attempting to convince their team to abandon their communication software in favor of physical boards.

Interview Analysis:

An important point to analyze in this data is the fact that all our participants are Scrum masters. This means that the input only reflects one role out of the three possible roles in the Scrum framework. A participant commented on communication software saying that: "Developers hate it. Jira is a great tool. It's Scrum masters, manual testers and managers that really like it. Some developers just want to work. They think it just takes lots of time from their work." Therefore, one must be wary of a possible discrepancy between the opinions of Scrum team members.

Another participant reflects that it varies based on personalities in the team saying: "we have people like it, and people who don't, and people who prefer physical boards." Given the cross-functionality of the team, one must be critical of the positive data that shows most Scrum teams viewing communication software in a positive light. A comparison between the views of the different team members views on this issue was not possible nor included within the scope of this study.

Another common theme explaining this challenge is too much focus or reliance upon the communication tools in a way that overcomplicates the tracking of work. Wendt discusses this saying:

> *If you can't do it physically on paper, what does that tell you? That's kind of the general first question I ask if I get into a coaching situation. And that typically means we don't have a clear enough focus. We have too many things going on. But what's the purpose? If you can't group all of this up, and try to summarize it so you can represent it physically and write it down simply, you have a focus problem. You don't know what outcome you're looking for. And then the tool is never going to help you.*

Therefore, we can summarize the two main problems or reasons behind this challenge as:
1) Different personalities depending on business expertise: Some roles such as the Scrum master and product owner are concerned a great deal with artifacts and are highly in touch with them. Developers, on the other hand, might be more used to coding without having to seek out information on communication software. Different personalities because of cross-functionality also mean that inevitably some members will appreciate the software while others will not.
2) Relying on communication software too much: Too much emphasis on digitizing communication and overusing communication software could mean that the team has too many tasks or items to tracks. This means that the problem potentially goes beyond the

specific software and is more related to the team's work process or the corporate culture of the organization.

Questionnaire Analysis:

Helpfulness	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 14	7	3	5	20	20	73 %

An overwhelming majority of respondents reported using Jira for communication in their Scrum team(s). This makes it the most used communication software for our respondents. Slack comes in second place where above half of the respondents use it. This is important to know in order to understand what the experience of most respondents relates to in terms of software when they talk about software usefulness in self-organizing.

A majority still viewed communication software as something that either considerably or significantly aids the self-organizing of Scrum teams. The distribution of the questionnaire respondents is more representative of Scrum teams and their various members than the interviews. However, the questionnaire respondents are definitely still mostly comprised of Scrum master who represent 71 % of the respondents. This means that a comparison between the perceptions of different roles in the team becomes something that is not feasible. While the data points to an overwhelming appreciation of communication tools, one must be aware of the fact that some Scrum masters admit themselves that they appreciate these tools while developers, for instance, might not.

Recommendations:

1) Consensus on tools and constant review: An interview participant commented on disagreements about communication software usefulness saying:

> *The important thing is that the team gets why the tools are there for them. The team should agree together on what they use. If that is not possible from the start, then this is something that should be reviewed. This is what sprint retrospectives are for, in order for the team to improve its tools and make sure it all works for them.*

Therefore, a team should use its opportunities of *inspect* and *adapt* in Scrum in order to discuss the tools and make their own decisions on what to use and what not to use. This topic should always be revisited to make sure the utmost transparency about it is ensured.
2) Look beyond the tool to see the bigger picture: Disagreement on communication tools is a risk but a bigger risk could be an exaggerated reliance on them. Scrum teams should set up constant checks in order to ensure that software is not the cornerstone of Scrum. Collaboration and communication are of course the purpose and the team should

choose how it works best to be self-organizing, be that with or without communication software.

However, the inability to move forward without software could indicate bigger problems such as too many tasks or projects to track. This can affect focus as discussed previously. On this point, it might be worth following the suggestion of a few Scrum masters who advocate for physical boards instead of software. In the end, the team should be given the choice to pick between physical boards, software or a mixture of both. However, a Scrum master should give the team the option to make that choice and see the reasoning behind it.

6.15. Challenge 15 - Varying Stances on Agile Training

Interview Data:

Only a handful of participants have brought the lack of agile training or a different perception of its importance as a threat to self-organization before the second part of the interview. About half of the participants acknowledge the importance of formal agile training for self-organizing. Almost half of them did not think it was important at all. A few participants did not have any input on the topic. It is noteworthy that 8 out of 13 of the participants had undergone some form of formal agile training and have some certification in it.

The relation between years of experience working with Scrum on the one hand and having agile certification on the other was also examined. Almost all candidates with 3 or more years of experience have undergone some form of agile training.
Finally, the interviews aimed to confirm whether there is a significant clash between the team and the organization in terms of valuing agile certification. This means that one of the two parties would value certification more than the other and friction would ensue. This specific problem was not found to be a serious or common issue when talking to participants.

Interview Analysis:

Most candidates have referred to certification as a positive but not a necessity for proper team self-organizing. A common theme here is that certification is a facilitator and help for the Scrum master in guiding the team towards self-organizing. However, on-the-job experiences and mentorship have been brought up as useful alternatives. An interview participant refers to the effect of formal agile training in facilitating work process saying:

> *If all team members are certified in Scrum, for example, and we decide that we work according to Scrum, then that would be way easier because everyone would know what we are supposed to do and why. But the challenge I face several times is that I need to*

decide what activities we do or we need as a team. We need to decide together what activities to do and discuss why they are useful. But sometimes people need to be educated first in Scrum, the purpose of Scrum, agile and the purpose of each activity.

Analyzing the data, one must be wary of the different levels of experience of the participants and whether they are biased due to having or not having the certification themselves. This might explain why there are responses like the one above claiming agile training is highly important to self-organize while others claimed it was not important at all.

A final point to analyze is the discrepancy between organizations and teams in valuing formal agile training. This is something that has been claimed in the theory but is very rarely represented in any of this study's interview.

From this, three conclusions can be drawn about this challenge:

1) Formal agile training facilitates self-organization but is not a prerequisite: As discussed by a number of participants, certification raises awareness of Scrum values and best practices. This facilitates self-organization but it is not impossible to have self-organization without the presence of certified individuals according to most participants.

2) Discrepancies between teams and organizations were not found: The challenge of certificates being valued by the team but not the organization or the other way around was not highly represented if at all by any participants.

Questionnaire Analysis:

Importance	Not at all	Slightly	Moderately	Considerable	Significantly	Majority Share
Challenge 15/1	25	9	12	11	6	55 %
Challenge 15/2	14	16	12	16	4	48.5 %

More than half the respondents believed that certification is not at all needed in order to achieve self-organization. Also almost half of the respondents reported that they believe their organizations only value certification slightly or even not at all. This explains the lack of discrepancy between team and organization in valuing formal agile training. As the data suggests.

It is, however, still understandable that a fair number of respondents reported that their organization value certification highly. One must be wary of bias here due to the fact that a number of the respondents are agile coaches and potentially come from agile organizations where agile certification is almost a given.

Recommendations:

1) Focus on a growth learning mindset instead of certification: This recommendation was one brought up by agile coach and Scrum.org trainer Fredrik Wendt when he says:

> *Having a mindset of growth is key in Scrum. This is because you are going to inspect and adapt. And if you don't like that, if you don't like changing and challenging yourself, you're not going to like [Scrum's] approach. So that mindset is more important than having the perfect skills from certifications. Even though Scrum.org has three levels of Scrum mastery, it's not a guarantee that you actually know how to apply it in real life and that you can balance short-term versus long-term results.*

Therefore, a mindset of embracing change, iteration and group learning can be way more important to self-organizing than simply having undergone some formal agile training.

2) Find a balance: Another problematic factor that can affect self-organizing teams is placing too much focus on certification. Therefore, a balance must be reached between theoretical studies and practical work and team development. This should be a balance that is reached through the collaboration of the team with the top management to make any potential problems transparent in this respect. An interview participant provided this recommendation and added "I think it's very good to have certification. However, I think that making teams or organizations go too deep into theory and that is maybe not the right way to go. But being very practical and playful, and engaging people, that could never go wrong."

7. Discussion

In this section, we discuss the findings and highlight important aspects of the data that were not extensively covered in other sections. The section starts by presenting a discussion about the validity and reliability of data. Following that a discussion about the data and interrelated challenges is presented. We will also relate the findings to the research questions, which are presented in 1.3 and also repeated below. Finally, a discussion about the new findings and their significance for self-organizing teams follows.

Q1: How much do the challenges to self-organizing acknowledged in the literature reflect and align with the practical experiences of Scrum practitioners?
Q2: How frequent are these challenges in reality and how severe is their impact on a Scrum team's effort to self-organize?
Q3: What are some possibly generalizable solutions and recommendations that Scrum practitioners offer with respect to each of these challenges?

7.1. Theory VS. Practice Discussion

This section discusses the alignment or misalignment of challenges discussed in the review of the literature with regards to the experiences of Scrum practitioners in reality. This aims, in part, to answer research question 1 (1.3). As the data shows a close alignment between data and theory in many cases, only the topics that diverge will be discussed. The topics omitted can, therefore, be assumed to have high alignment between theory and practice. We also touch upon research question 2 (1.3) in discussions about the severity and frequency of said challenges.

Regarding Geographic Issues, theory (3.1 - 5) and data (6.5) align well, but additional relevant aspects can be found in the data. Previous research did not show an emphasis on the relational aspect of being dispersed, neither does it explicitly cover the flow of informal information, two main aspects brought up in the data.

The description of previous research regarding Delayed and Changing Requirements (3.1 - 12) put more emphasis on the severity of the challenge than what was actually experienced in the data (6.12). Although frequent experiences of the challenge are apparent in the data, a large portion of respondents perceive this to be a part of working with agile, and not a challenge.

The presentation of previous research regarding Lack of Individual Accountability (3.1 - 11) does not at all reflect the core of the problem experienced in the data (6.11). The data suggest that the main cause for this challenge comes from the

organization and reward systems disrupting team priorities, rather than from within the team itself.

Research presentation of Effective Estimations (3.1 - 10) aligns fairly well with the data outcome (6.10) but for the maturity aspect where data suggest this challenge to be represented mostly in new teams.

The data on Division of Tasks (6.8) brings forth one aspect that previous research (3.1 - 8) does not - that the challenge is closely linked to poor estimation practices done by the team.

The research description of the cause of the Multiple projects challenge (3.1 - 6) aligns well with data (6.6). Data highlights the additional aspect of a lack of transparency when it comes to making stakeholders aware of backlog priorities which might be an additional cause.

Regarding Support from Organization, previous research (3.1 - 1) highlights that top management might be hesitant and prefer safe options, which data (6.1) does not.

Discussing the challenge of Documentation Requirements research (3.1 - 13) aligns well with data (6.13) in terms of causes, but not necessarily in severity, as the data show that this is not perceived as much of a challenge.

7.2. Discussion of Findings

From the findings, one can observe a variation in the frequency and severity of each and every challenge that could threaten self-organization. However, an important point to raise is the potential connections or causality between those challenges. In essence, some of the challenges on the reviewed list of challenges could be a result of another challenge on the list. For instance, poor division of tasks could be attributed to a lack of trust, which in turn can be traced back to a lack of support for the Scrum team by the organization. That is why we believe in the importance of making the best out of the *inspect* and *adapt* opportunities offered by the Scrum framework.

Raising transparency about the challenges and bringing any potential conflicts to the surface can help teams stay on top of these challenges and guide them to understand the root causes better. In this sense, the strength of root cause analysis tools and open dialogues between Scrum teams and stakeholders is of huge importance. Scrum is a framework that is reliant upon empiricism, validated learning and iteration. This means that Scrum teams that capitalize upon their retrospectives and are utilizing their *inspect* and *adapt* chances properly can become more effective at tracking interdependent challenges.

The third research question (1.3) of this paper was about providing recommendations on how to solve these challenges. However, it is impossible for teams to solve the challenges without having the self-awareness and transparency required in

order to identify they are facing the challenge in the first place. Therefore, future Scrum teams can benefit from having the list of challenges in the literature compared to the experiences of Scrum practitioners, which corresponds to the first research question (1.3) of this paper. It would be, thus, positive for Scrum masters to acquaint themselves with these challenges and collaborate with their teams to find connections between them.

By tracking the connections and causality between challenges, one can implement the proper recommendations related to the challenges and also keep all stakeholders informed about the struggles of the Scrum team. This creates an inclusive and collaborative environment where everyone can pitch in about the causes of the challenges and also their proposed solutions for them. The recommendations in this paper are not to be treated as exhaustive remedies to the challenges, but like Scrum, they provide frameworks and tools that can help Scrum masters combat these challenges more effectively.

We acknowledge that Scrum teams consist of various personalities and this means that a one-size-fits-all solution is not possible for many of, if not all the challenges. For example, as previously discussed, an interview participant was facing a problem of new developers demanding documentation about the process and the work of the Scrum team. Instead of fighting the documentation, as some have recommended, this Scrum master actually embraced it. By writing the process down, all members learned more about it. When presented with the same dilemma, other Scrum masters could be protective of their process and rather not grant the wishes of the developers for documentation. Instead, practices of pair programming, coaching and mentoring could be used. None of these two approaches are right or wrong in absolute terms. We believe that you have to fine-tune your tools and decide your approach to the challenge based on your knowledge of the personalities on the team. Since Scrum is centered around small teams where the individual gets a lot of say, this becomes something of paramount importance. In short, the tools and recommendations offered should be implemented paired with the Scrum master's knowledge of their team's resources, personalities and interpersonal traits.

From the list of challenges and analysis of the data, we can see that some challenges have a closer correlation between them. Others do not. For instance, agile certifications do not directly or reasonably connect with problems regarding personality clashes. On the other hand, poor division of tasks can be the direct cause for dependencies between team members and even others outside the Scrum team. It is noteworthy that the two top challenges identified in this study can be identified as the direct cause of each and every challenge on the list in certain situations. The first, which relates to a lack of support from the organization, directly undermines the autonomy and self-organizing nature of the Scrum team. Personality clashes can arise due to that. A lack of trust can be a natural outcome. In the section about the lack of individual

accountability, a lack of support from the organization was actually identified as one of the prime causes of the challenge.

Going through the list of challenges, it is easy to connect each and every challenge to a lack of support from the organization. This confirms the severity of this challenge to self-organizing and explains why it deserves special attention by Scrum practitioners in our point of view. Lack of trust is a challenge that behaves similarly. This means that all of the challenges, even lack of support from the organization itself, can be traced back to a lack of trust preventing the top management from letting go of control and allowing Scrum teams to self-organize.

7.3. New Findings

Using the GT approach elements (as described in 4.1) when analyzing the data collected for possible solutions to the challenges also provided the opportunity to extract new challenges present in the data that the literature review does not cover. These new challenges might be applicable for future research and are presented below.

1) Lack of availability of proper personnel

A new angle found, that threatens self-organizing and some of the core concepts of Scrum in regards to cross-functionality, is simply a lack of access to competent employees. Given that self-organizing and the solutions to several of the challenges discussed, depend on utilizing the competencies within the team, or improving the lack thereof by new hires or replacement, this challenge might be particularly difficult.

An interview participant claims that “the greatest problem that we're facing into truly becoming self-organizing would be that we have this key personnel dependency still” and another explains that “ right now in the region of Malmö, Helsingborg, Copenhagen, the Skåne south-west part, we have a specific [challenge]. Basically, every large company is seeking the same kind of competency and resources. It's a stress to have enough people to do what you want to do.” Although not a challenge experienced by many, this could pose some difficulties for some teams attempting to be cross-functional and self-organizing.

2) Project Focus VS. Product Focus

Rather than a new challenge, this finding is described as a possible cause for many of the challenges discussed in this paper. It is described here because of its possible coverage of multiple topics. The notion is presented by Fredrik Wendt and suggests that product management might “look at just output” instead of assessing “outcomes and impact” and that in the end, the product is what you produce, while the process is how you produce it. If they “rather look at just output - ‘did we perform according to schedule according to plan’ then they are no longer creating the boundaries for people to self organize”. Instead, they focus on factors like “did we release on time? Is that the most important thing? Let’s

say [we released] two months earlier, but failed to deliver what people wanted. What's the value of that?"

Wendt went on to explain that "they eventually realize that a project mindset is more about internal elements. This means that people are being engaged, they like the work, they communicate and they understand what they're doing and why". This is of importance, but "no one wants to buy people having fun. I buy the [popular product] because it looks good. I like the interface, or I hate it." It can be a problem when "everything is assigned into projects, people lose track of the product. Are customers happy when they use the product? Does it make life better for them? Instead of asking these questions, they focus on: we need to deliver this in within six months, because that's when the project budget ends."

3) Mentor VS. Coach

Another common mistake Scrum masters make according to Wendt is that they often take on the role of a mentor instead of really being a coach - "If I'm a mentor, which is where I think a lot of Scrum masters start, and sadly also stop [..] and if I'm mentoring this person or this team, I use all of my wisdom, experience, tools, techniques, tricks and frameworks, etc. And I try to get it into their heads". This behavior can really limit self-organizing when Scrum masters have an approach such as "You don't need to think about a solution to this problem. I have it. You don't need to self organize. I can give you three options and you just pick one." It is not about the Scrum master not having the necessary skills or experience to provide solutions, and sometimes it is necessary to mediate and facilitate. But as Wendt explains it a coaching approach could be "I don't know how to get there either. But I believe that you have all the skills, powers and the wisdom needed to figure that one out. [...] They own the agenda, they know exactly what to do [...] I just mirror and reflect - I ask powerful questions."

8. Conclusion

This paper presented 15 challenges that can affect the self-organizing nature of a Scrum team. By using a Mixed method and elements of Grounded theory we have investigated whether, how and to what extent, these challenges are experienced by Scrum teams in reality (research question 1 - 1.3), and what can be done to possibly mitigate these challenges (Research question 3 - 1.3). An analysis of interview data and a presentation of the questionnaire responses demonstrated an overview of the severity and frequency of these challenges with respect to their effect on self-organization (Research question 2 - 1.3). By analyzing data from interviews and questionnaires, surveying expert practitioners, we have generated recommendations based on this expert experience relating to these challenges. Our ambition with this study has been to determine whether we, by using both inductive and research-based approaches, can capture solutions to complex challenges and be able to generalize experience to offer actionable advice.

To conclude our research on the challenges Scrum teams face in self-organizing, some aspects found are worth mentioning in particular that not only are relevant to one but several of the challenges we already have generated recommendations for, and altogether might help clear up the bigger puzzle of self-organization in this context.

The single biggest cause found in the experience based data regarding these challenges comes from the organization itself with a lack of support, understanding and suitable supporting structures. Many other challenges could stem from a lack of support from the organization which means that this challenge deserves extra attention. Without the support of the organization, a Scrum team might face severe problems that hinder their self-organizing capabilities. This problem stretches out to affect many functions of the Scrum team and can be one of the direct causes for other challenges discussed in the paper. One of the most detrimental issues here is that top management does not understand the principles of Scrum nor their purpose in implementing it.

We believe that this threatens two values that are central to the Scrum framework, namely commitment, and focus. Without having support for autonomy and empowerment of the self-organizing team, it is difficult for the team to commit to business goals and feel a sense of ownership of their process. Also, without understanding the purpose of the top management's action, a Scrum team will struggle with focus and self-organizing will suffer. The result is a hybrid between Scrum and waterfall methods of project management which is something that does not capitalize on the benefits of either. Therefore, it is our point of view that, by securing top management buy-in, ensuring their support and understanding of Scrum, many of the challenges listed as threats to self-organizing can be at least partially avoided. Additionally, it can be seen as ironic that

one of the biggest hindrances to self-organizing teams stems from the lack of *external* support.

One of the core features of the Scrum framework, namely cross-functionality, seems to be particularly important for providing tools for self-organization. Optimizing who is in the team is crucial to be able to effectively deal with challenges, regardless whether they relate to diversity, complexity, planning, unforeseen circumstances, workload, adaptability or even location. Ensuring that all competencies needed are present within the team, that members are set on collaborating and being prepared to experiment with team formation are of paramount importance for having a team that can actively solve challenges *as a team*.

Another key to the puzzle is a growth mindset and a constant strive for an understanding, both of agile principles and of the context it is implemented in. A team needs to find tools to make their different understandings of processes, tasks and each other visible, and utilize those differences for learning and improving. Different competencies cannot be fully utilized if all team members stay within their own lane, or comfort zone. It is in the meeting between different experiences and views where collective, not individual thinking, can be used to a greater sum than the separate parts. Coaching others on Scrum principles and mindsets are important, but it should not necessarily be a top-down process only. Team members should coach each other, improve weaknesses by using strengths to solve problems together and utilize cross-functional aspects, to stay cross-functional, even in new settings. Pair or mob programming have been reported to be a useful method for solving more complex tasks, but also to share and utilize experiences from different areas of expertise. These have been the recommendations of several Scrum masters participating in this study, trying to provide coaching and advice, both to their own teams and to others by sharing experiences with each other, through forums, slack groups or through this study. As a final piece of advice from Agile coach, Fredrik Wendt:

> *Unfortunately, not a lot of Scrum masters coach each other. And that's something we introduce to the organization. You as scrum masters know that coaching is powerful, right? They say: yeah. we know. Are you doing it to yourself? No. Why aren't you? You know how useful it is and what it does to people and their performance.*

Bibliography

Akif, R. and Majeed, H., 2012. Issues and Challenges in Scrum Implementation. *International Journal of Scientific & Engineering Research,* 3(8), pp.1-4.

Bergman, G., 2011. The Scrum Story. *Lean Magazine.* Available at: http://leanmagazine.net/scrum/the-scrum-story/ [Accessed May 24, 2019].

Boehm, B. and Turner, R., 2009. *Balancing Agility and Discipline: A Guide for the Perplexed, Portable Documents*. Addison-Wesley Professional.

Brooks Jr, F.P., 1995. *The Mythical Man-month.* anniversary ed..

Brown, N., Cai, Y., Guo, Y., Kazman, R., Kim, M., Kruchten, P., Lim, E., MacCormack, A., Nord, R., Ozkaya, I. and Sangwan, R., 2010, November. Managing Technical Debt in Software-reliant Systems. *In Proceedings of the FSE/SDP workshop on Future of software engineering research (pp. 47-52).* ACM.

Caracelli, V.J. and Greene, J.C., 1993. Data Analysis Strategies for Mixed-method Evaluation Designs. *Educational evaluation and policy analysis*, *15*(2), pp.195-207.

Carroll, B., 1999. A Template for Accelerating the Development of Self Managed Work Terms. *National Productivity Review*, 18(4), pp.21–28.

Cho, J., 2008. Issues and Challenges of Agile Software Development with Scrum. *Issues in Information Systems,* 9(2), pp.188-195.

Codabux, Z. and Williams, B., 2013, May. Managing Technical Debt: An Industrial Case Study. In *Proceedings of the 4th International Workshop on Managing Technical Debt* (pp. 8-15). IEEE Press.

Cohn, M., 2004. User stories applied: For Agile Software Development. *Addison-Wesley Professional.*

Cole, R. and Scotcher, E., 2016. *Brilliant Agile Project Management: a Practical Guide to Using Agile, Scrum and Kanban.* Pearson UK.

Covey, S.R., 2004. *The 7 Habits of Highly Effective People: Powerful Lessons in Personal Change*. Simon and Schuster.

Cusumano, M.A. and Smith, S.A., 1995. Beyond the Waterfall: Software Development at Microsoft. *MIT Sloan School of Management.*

Deemer, P., Benefield, G., Larman, C. and Vodde, B., 2010. The Scrum Primer. *Scrum Primer.*

Denning, S., 2016. Agile's Ten Implementation Challenges. *Strategy & Leadership,* 44(5), pp.15-20.

ForbesInsights, 2017. Achieving Greater Agility: the Essential Influence of the C-Suite. *PMI.* Available at: https://i.forbesimg.com/forbesinsights/pmi/achieving_greater_agility.pdf. [Accessed May 24, 2019].

Gilley, J.W., Morris, M.L., Waite, A.M., Coates, T. and Veliquette, A., 2010. Integrated Theoretical Model for Building Effective Teams. *Advances in Developing Human Resources,* 12(1), pp.7-28.

Gregory, P., Barroca, L., Taylor, K., Salah, D. and Sharp, H., 2015, May. Agile Challenges in Practice: a Thematic Analysis. *In International Conference on Agile Software Development* (pp. 64-80). Springer, Cham.

Hamel, G., 2017. First, Let's Fire All the Managers. *Harvard Business Review.* Available at: https://hbr.org/2011/12/first-lets-fire-all-the-managers [Accessed May 24, 2019].

Highsmith, J.A., 2002. *Agile Software Development Ecosystems.* Addison-Wesley Professional.

Hoda, R. and Murugesan, L.K., 2016. Multi-level Agile Project Management Challenges: A Self-organizing Team Perspective. *Journal of Systems and Software*, *117*, pp.245-257.

Hoda, R., Noble, J. and Marshall, S., 2011. Developing a Grounded Theory to Explain the Practices of Self-organizing Agile Teams. *Empirical Software Engineering*, *17*(6), pp.609-639.

Holmes, J., 2011. Agile Prerequisite or Pipe Dream? *Training Journal.* P.69-71.

Holmström, H., Fitzgerald, B., Ågerfalk, P.J. and Conchúir, E.Ó., 2006. Agile Practices ReduceDistance in Global Software Development. *Information Systems Management*, 23(3), pp.7-18.

Jeldi, N.P. and Chavali, V.K.M., 2013. Software Development Using Agile Methodology Using Scrum Framework. *International Journal of Scientific and Research Publications*, *3*(4), pp.1-3.

Jia, Y., Larusdottir, M.K. and Cajander, Å., 2012, October. The Usage of Usability Techniques in Scrum Projects. In *International Conference on Human-Centred Software Engineering* (pp. 331-341). *Springer,* Berlin, Heidelberg.

Kniberg, H., 2015. *Scrum and XP from the Trenches.* Lulu.com.

McKenna, D., 2016. *The Art of Scrum: How Scrum Masters Bind Dev Teams and Unleash Agility.* Apress.

Melnik, G. and Maurer, F., 2003, August. Introducing Agile Methods in Learning Environments: Lessons Learned. *In Conference on Extreme Programming and Agile Methods* (pp. 172-184). Springer, Berlin, Heidelberg.

Moe, N.B. and Dingsøyr, T., 2008, June. Scrum and Team Effectiveness: Theory and Practice. *In International Conference on Agile Processes and Extreme Programming in Software Engineering* (pp. 11-20). *Springer,* Berlin, Heidelberg.

Murugesan, L.K., 2016, December. Overcoming Challenges in Self-organizing Agile Software Teams. *In 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)* (pp. 1-4). IEEE.

My Personality Test, True Colours. *My Personality Test.* Available at: https://my-personality-test.com/true-colours [Accessed May 24, 2019].

Nerur, S., Mahapatra, R. & Mangalaraj, G., 2005. Challenges of Migrating to Agile Methodologies. *Communications of the ACM, 48(5),* p.72-78.

Pries, K.H. and Quigley, J.M., 2010. *Scrum Project Management.* CRC press.

Rigby, D.K., Sutherland, J. and Takeuchi, H., 2016. Embracing agile. *Harvard Business Review,* 94(5), pp.40-50.

Roper, K.O. and Phillips, D.R., 2007. Integrating Self-managed Work Teams into Project Management. *Journal of Facilities Management,* 5(1), pp.22-36.

Rubin, K.S., 2013. *Essential Scrum: A Practical Guide to the Most Popular Agile Process.* Addison-Wesley.

Schwaber, K., 2004. *Agile Project Management with Scrum.* Microsoft press.

--, 1997. Scrum Development Process. *In Business Object Design and Implementation* (pp. 117-134). Springer, London.

-- , 2010. What is Scrum. *Scrum Alliance.*

Schwaber, K. & Sutherland, J., 2017. The Scrum Guide. *Scrum.org.*

Scrum Alliance, Self-organizing Scrum Teams - Challenges and Strategies. *Scrum Alliance.* Available at: https://agilealliance.org/wp-content/uploads/2016/01/Self-organizing-Scrum-teams-Challenges-and-Strategies.pdf [Accessed May 24, 2019].

Serrat, O., 2017. The Five Whys Technique. *In Knowledge solutions* (pp. 307-310). Springer, Singapore.

Stellman, A. and Greene, J., 2014. *Learning agile: Understanding scrum, XP, lean, and kanban.* O'Reilly Media, Inc.

Stoica, M., Mircea, M. and Ghilic-Micu, B., 2013. Software Development: Agile vs. Traditional. *Informatica Economica,* 17(4).

Stray, V., Moe, N.B. and Hoda, R., 2018. Autonomous Agile Teams: Challenges and Future Directions for Research. *arXiv preprint arXiv:*1810.02765.

Takeuchi, H. and Nonaka, I., 1986. The New New Product Development Game. *Harvard Business Review,* 64(1), pp.137-146.

Tata, J. and Prasad, S., 2004. Team Self-management, Organizational Structure, and Judgments of Team Effectiveness. *Journal of Managerial Issues*, pp.248-265.

The Nine Belbin Team Roles. *Belbin.com.* Available at: https://www.belbin.com/about/belbin-team-roles/ [Accessed May 24, 2019].

Yeatts, D.E. & Hyten, C., 1998. *High-performing Self-managed Work Teams: a Comparison of Theory to Practice*, Thousand Oaks, CA: Sage.

Appendix 1. Interview Design

Intro - who we are, why, self-organizing teams.

Quantitative Questions:

- Do you have any form of agile formal training or certification?
- How many years of experience do you have working with Scrum?
- How many team members are there in your team(s)?

Interview Section 1: Questions

- What is your understanding of a self-organizing team?
- Has your organization adopted a scaled Scrum? If not -
- How does the team interact with other (Scrum) teams or others outside the scrum team?
- Do you have any external elements to the Scrum framework that you added to your work process?
- Do you have any elements that you purposely have removed? Why?
- Based on your understanding of "Self-organizing" what are the challenges you, as a team, face in order to achieve that?
- What are some of the recurring impediments in your work process that you have to solve?
- How do you solve them?

Interview Section 2: Topics to cover

- Support from the Organization
- Lack of Trust
- Task Dependency
- Negative Perception of Scrum Events
- Geographic Issues
- Multiple Projects
- Personality Clashes
- Division of Tasks
- Drop in Quality and Technical Debt
- Lack of Valuable Estimations
- Lack of Individual Accountability
- Delayed and Changing Requirements
- Excessive Documentation Requirements
- Different Perceptions of Communication Software
- Varying Stances on Agile Training

Appendix 2. Questionnaire Design

Scrum Team Challenges in Self-organizing

Thank you for filling out our survey!

We are two students at Lund University, studying the M.Sc. in Management, and we are investigating what challenges might affect the efforts of Scrum teams to self-organize. We want to know how you experience these challenges and how one might try to solve them. With the help of your experience we hope to generate practical advice for other teams who might face these challenges.

Researchers:
Ahmad Hemaily - ahmad.elhemaily@gmail.com
Boman Lyngfelt - bomanlyngfelt@gmail.com

We will ask you to rate how you perceive the challenges, but you will also have the option to provide what you think might be a solution for each challenge. If you answer all optional questions the survey might take up to 20 minutes, but you are of course allowed to skip the ones that you do not have any input on.

We greatly appreciate you taking the time to share your expert knowledge with us!

* Required

1. **I am a**
   Check all that apply.
   - [ ] Developer
   - [ ] Scrum Master
   - [ ] Product Owner
   - [ ] Other: _____

2. **Please indicate if you are someone outside the Scrum team that oversees the team process**
   Mark only one oval.
   - ◯ Yes
   - ◯ No

3. **Number of members in the team (icluding all roles) - If you are part of several teams; write the number of members in all of them [eg. 6, 7 & 8]**

4. **Years of experience working in a Scrum Team**

5. **Do you have any form of Scrum Certification? If Yes, which?**

6. **What country are you currently employed in?**

Challenges

In the next sections, we ask you about challenges Scrum teams may face, how frequent they are and how severely they may affect a team's self-organization. Finally, we ask you for recommendations on how these challenges may be solved.

7. **How closely does your team follow the guidelines of the Scrum framework?** *
Mark only one oval per row.

	Not at all	Somewhat	Moderately	Considerably	Meticulously	Not sure
Options	◯	◯	◯	◯	◯	◯

8. **Does your team use tools or elements from another agile framework?** *
Check all that apply.

- [ ] No
- [ ] Kanban
- [ ] XP
- [ ] Other: _____

Challenge: Personality Clashes

This challenge relates to conflicts resulting from different personalities in the Scrum team.

9. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency	◯	◯	◯	◯	◯

10. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity	◯	◯	◯	◯	◯

11. **What could be done in order to solve this?**

_____

_____

_____

_____

_____

Challenge: Scrum Events

This challenge relates to some team members having the mindset that Scrum events are time-consuming and take them away from work.

12. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency	◯	◯	◯	◯	◯

13. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity	◯	◯	◯	◯	◯

14. **What could be done in order to solve this?**

Challenge: Quality and Technical Debt

This challenge relates to teams rushing work in order to meet a sprint goal and thus compromising quality of the end-product. This means that the team has to go back and fix 'done' work due to possible errors or oversights.

15. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency	◯	◯	◯	◯	◯

16. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity	◯	◯	◯	◯	◯

17. **What could be done in order to solve this?**

Challenge: Individual Accountability

This challenge relates to the fact that accountability and credit belong to the self-organizing Scrum team. A team member's failure or success is attributed to the whole team, something that can be seen as unfair by some team members or yourself.

18. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency	◯	◯	◯	◯	◯

19. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity	◯	◯	◯	◯	◯

20. **What could be done in order to solve this?**

Challenge: Proper Task Allocation

This challenge relates to the self-organizing team struggling to divide tasks appropriately. Efficiency drops if an inexperienced team member is assigned a task that is too difficult. The same happens if a highly experienced member is assigned a task that is too easy for them.

21. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency	◯	◯	◯	◯	◯

22. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity	◯	◯	◯	◯	◯

23. **What could be done in order to solve this?**

Challenge: Task Estimation

This challenge relates to difficulties or disagreements about the estimation of tasks/user stories in the Scrum team. The outcome is highly inaccurate estimates or potential team conflicts.

24. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency	◯	◯	◯	◯	◯

25. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity					

26. **What could be done in order to solve this?**

Challenge: Multiple Projects

Retaining focus could become difficult for a Scrum team if the team is working on multiple projects at the same time.

27. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency					

28. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity					

29. **What could be done in order to solve this?**

Challenge: Delayed/Changing Requirements

This challenge relates to a change or delay of acceptance criteria or requirements from clients or other parts of the organization.

30. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency					

31. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity					

32. **What could be done in order to solve this?**

Challenges: Top Management Support

This challenge relates to top management not supporting the Scrum team in their efforts to self-organize. This could mean that the Scrum team's decisions are not being respected or that the organization is interfering too much with the Scrum team's work.

33. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency					

34. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity					

35. **What could be done in order to solve this?**

Challenges: Documentation Requirements

This relates to top management demanding too much documentation which is something that goes against the agile nature of Scrum.

36. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency					

37. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity					

38. **What could be done in order to solve this?**

Challenges: Task Dependency

This challenge relates to tasks being dependent on one another. This could be within the team, meaning an overlap of tasks/user stories. Alternatively, it could be that the Scrum team is waiting too much on other teams or individuals, which threatens self-organization.

39. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency					

40. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity					

41. **What could be done in order to solve this?**

Challenge: Lack of Trust

This relates to commitment and respect, two of the core Scrum values. Lack of trust between team members can be an impediment hindering the process of self-organization.

42. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency					

43. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity					

44. **What could be done in order to solve this?**

Challenge: Geographic Issues

This challenge relates to difficulties in self-organizing due to the team being geographically dispersed.

45. **How often do you experience this challenge?**
Mark only one oval per row.

	Never	Rarely	Occasionally	Often	All the time
Frequency	◯	◯	◯	◯	◯

46. **How threatening can this be to a Scrum team's ability to be self-organizing?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Severity	◯	◯	◯	◯	◯

47. **What could be done in order to solve this?**

Agile Training

48. **How important do you think agile certification is for self-organization?**
Mark only one oval per row.

	Not at all	Slightly important	Moderately important	Considerably important	Extremely important
Options	◯	◯	◯	◯	◯

49. **In your opinion, how much does your organization as a whole value formal agile training?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Extremely
Options	◯	◯	◯	◯	◯

Communication Software Tools

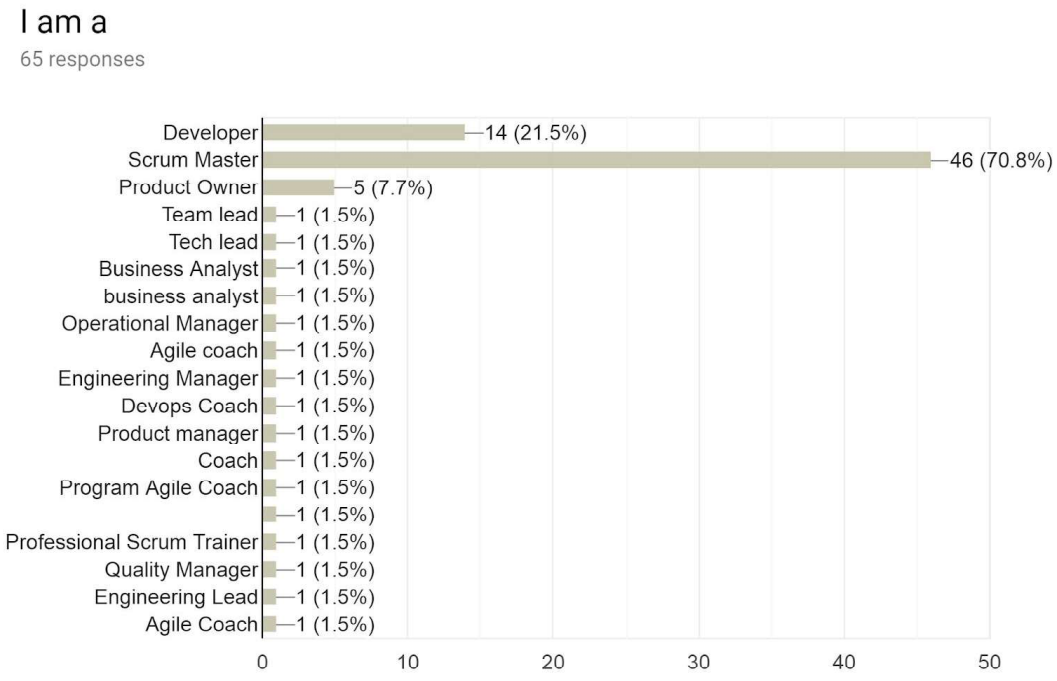
50. **What communication software tools do you use in your work?**
Check all that apply.

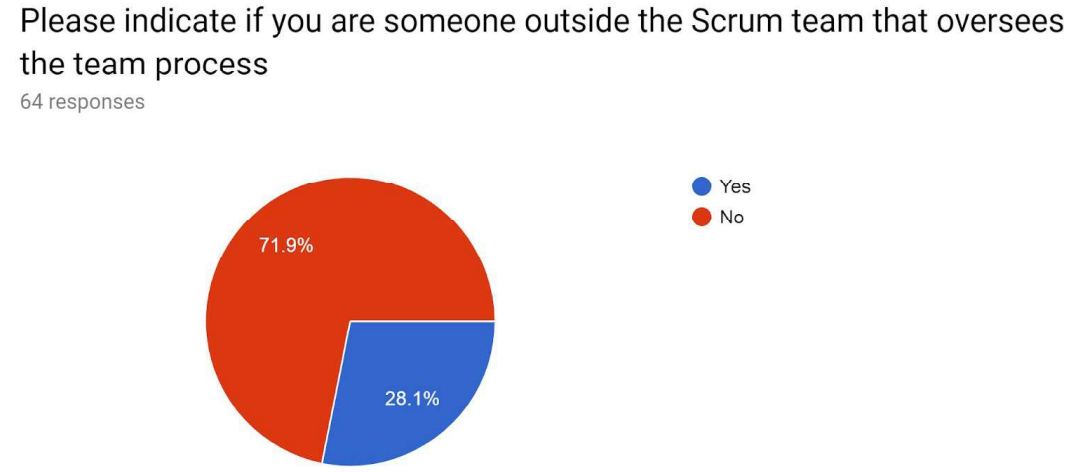
- [ ] Jira
- [ ] Slack
- [ ] Trello
- [ ] Google Docs
- [ ] Other: _____

51. **Do you believe they enhance communication and help you become a more self-organizing team?**
Mark only one oval per row.

	Not at all	Slightly	Moderately	Considerably	Significantly
Options	◯	◯	◯	◯	◯

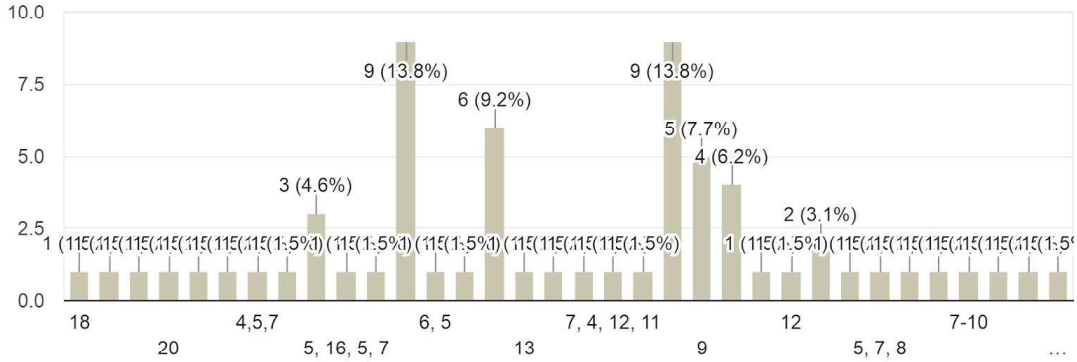
Appendix 3. Questionnaire Data





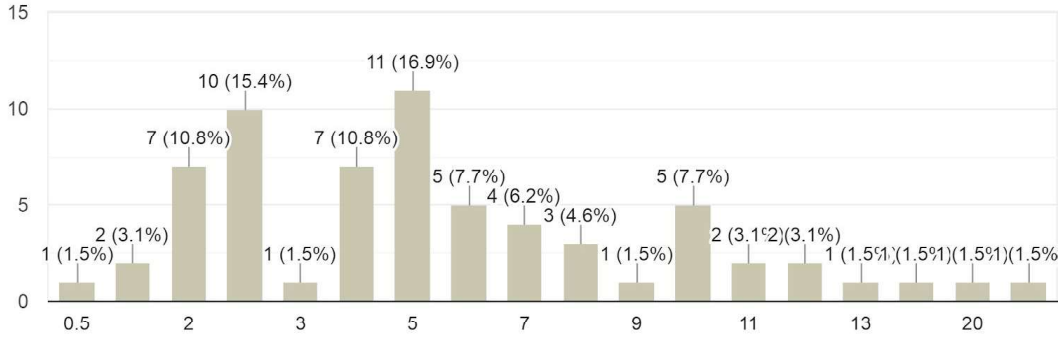
Number of members in the team (icluding all roles) - If you are part of several teams; write the number of members in all of them [eg. 6, 7 & 8]

65 responses



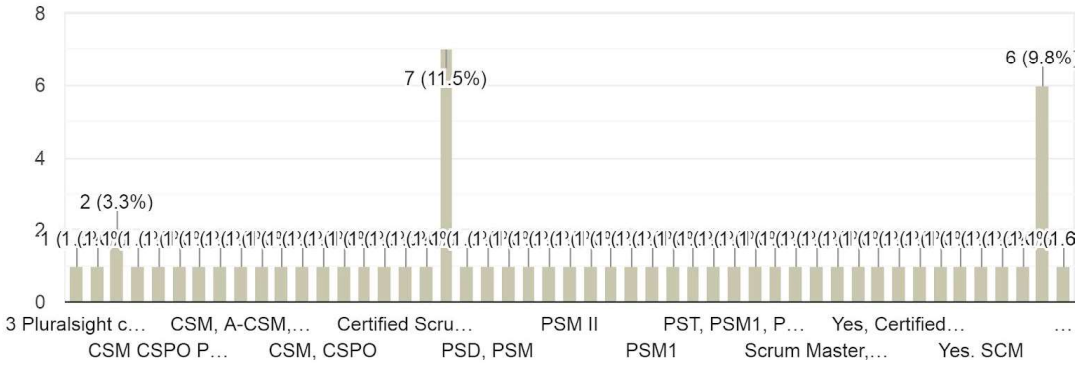
Years of experience working in a Scrum Team

65 responses



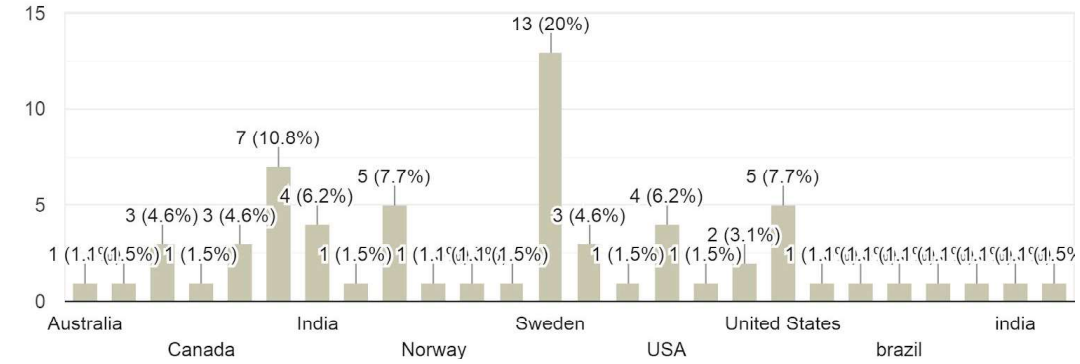
Do you have any form of Scrum Certification? If Yes, which?

61 responses



What country are you currently employed in?

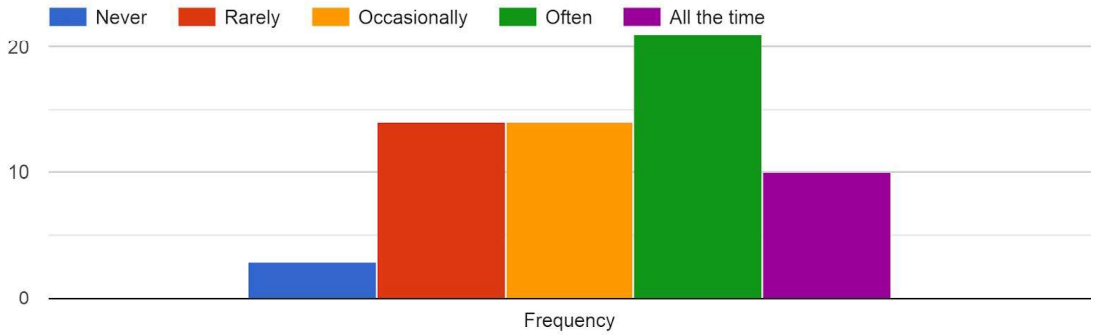
65 responses



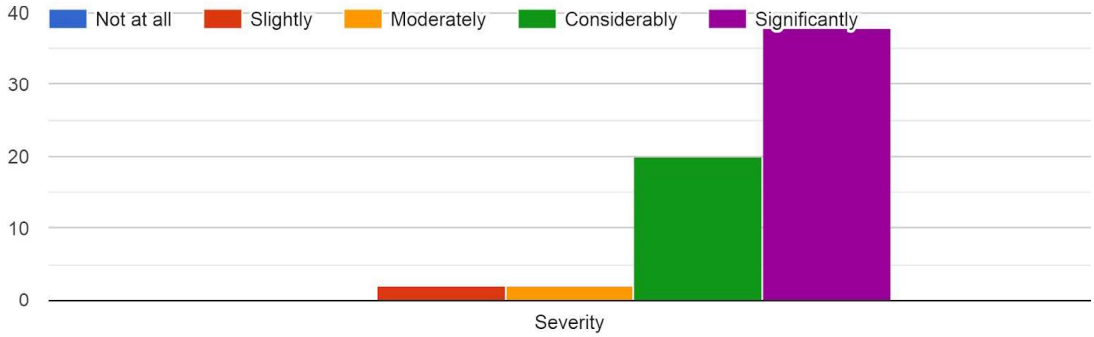
1. Support from the Organization

This challenge relates to top management not supporting the Scrum team in their efforts to self-organize. This could mean that the Scrum team's decisions are not being respected or that the organization is interfering too much with the Scrum team's work.

How often do you experience this challenge?



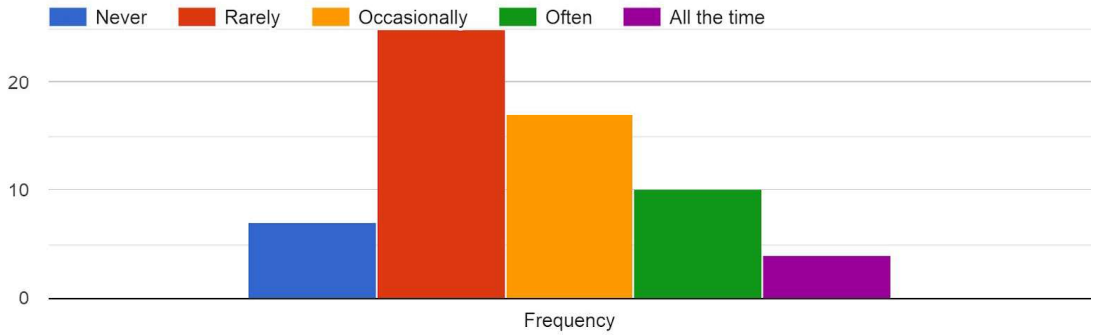
How threatening can this be to a Scrum team's ability to be self-organizing?



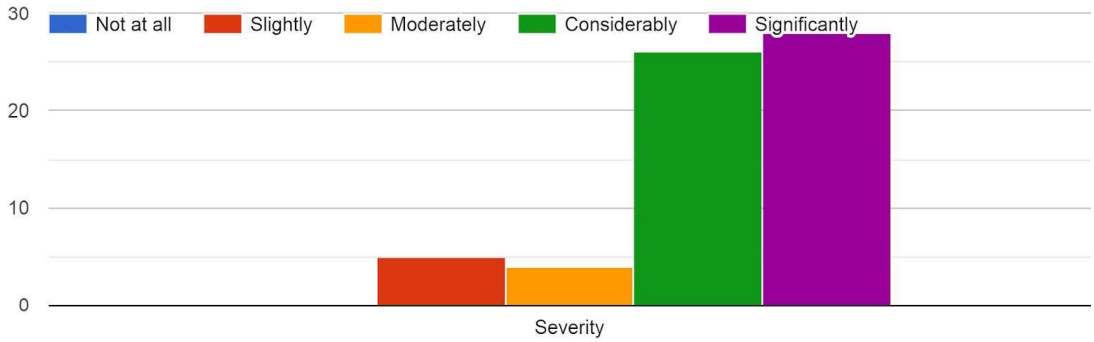
2. Lack of Trust

This relates to commitment and respect, two of the core Scrum values. Lack of trust between team members can be an impediment hindering the process of self-organization.

How often do you experience this challenge?



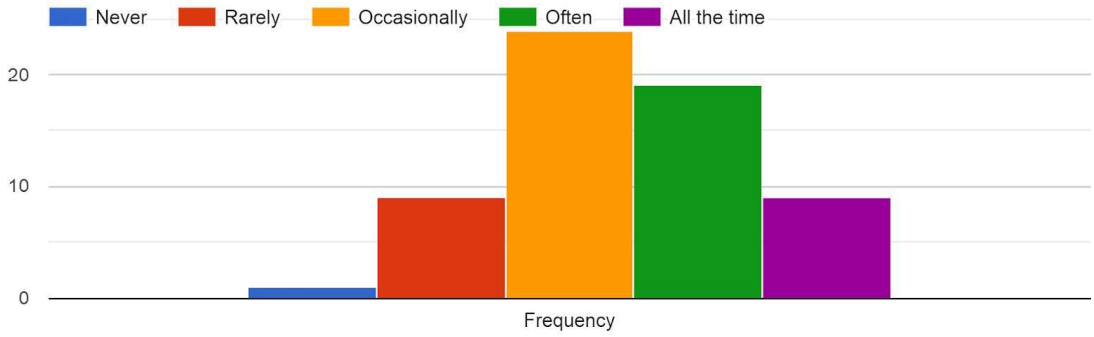
How threatening can this be to a Scrum team's ability to be self-organizing?



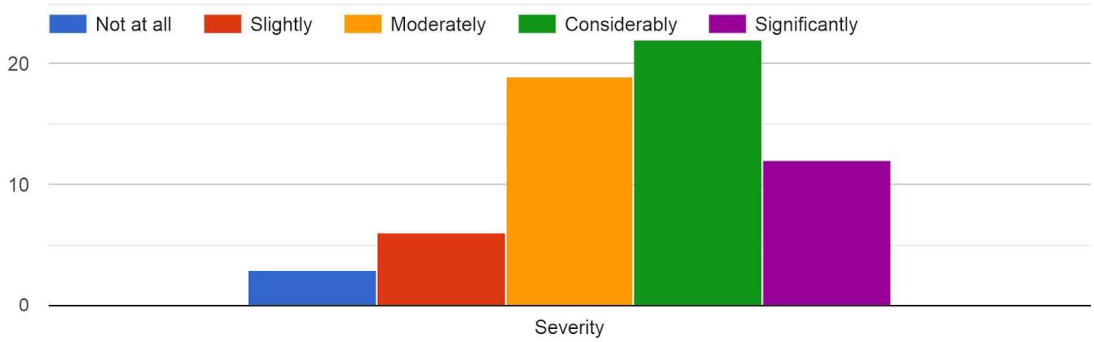
3. Task Dependency

This challenge relates to tasks being dependent on one another. This could be within the team, meaning an overlap of tasks/user stories. Alternatively, it could be that the Scrum team is waiting too much on other teams or individuals, which threatens self-organization.

How often do you experience this challenge?



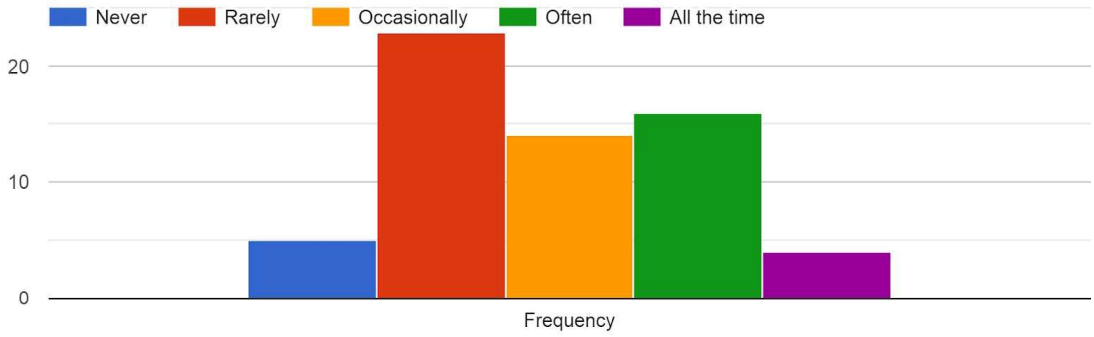
How threatening can this be to a Scrum team's ability to be self-organizing?



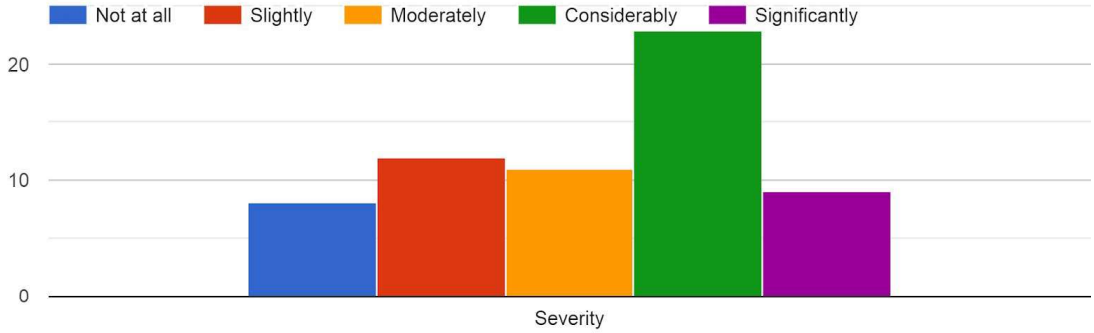
4. Negative Perception of Scrum Events

This challenge relates to some team members having the mindset that Scrum events are time-consuming and take them away from work.

How often do you experience this challenge?



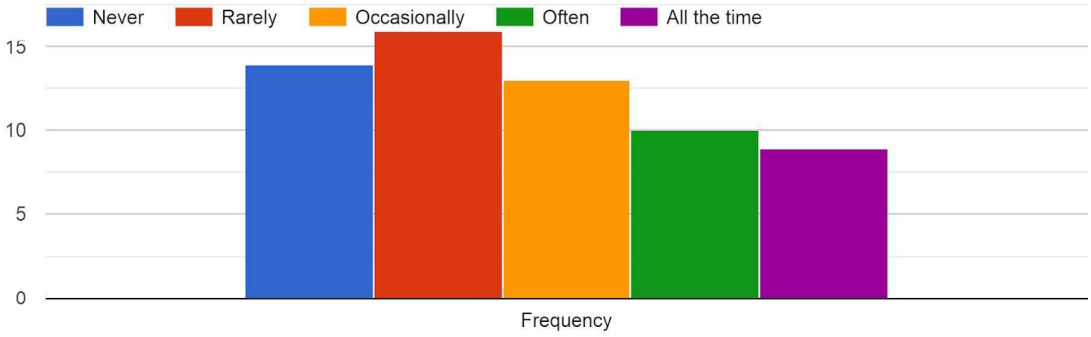
How threatening can this be to a Scrum team's ability to be self-organizing?



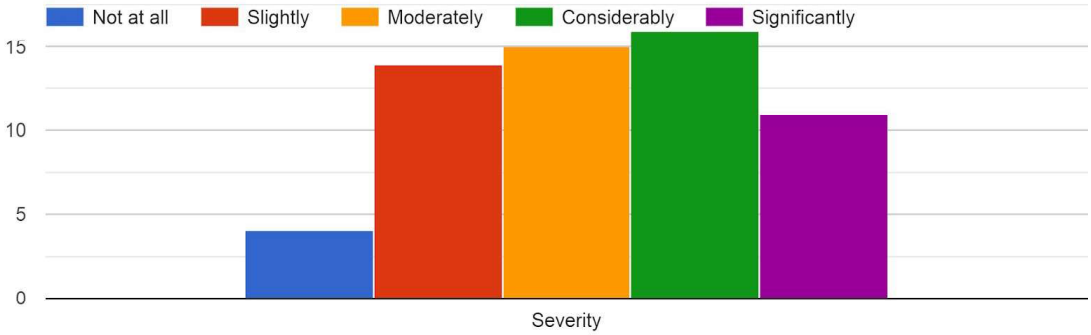
5. Geographic Issues

This challenge relates to difficulties in self-organizing due to the team being geographically dispersed.

How often do you experience this challenge?



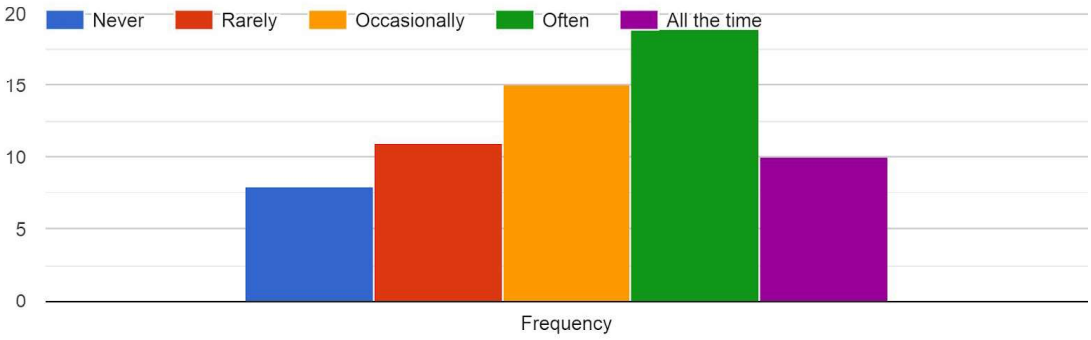
How threatening can this be to a Scrum team's ability to be self-organizing?



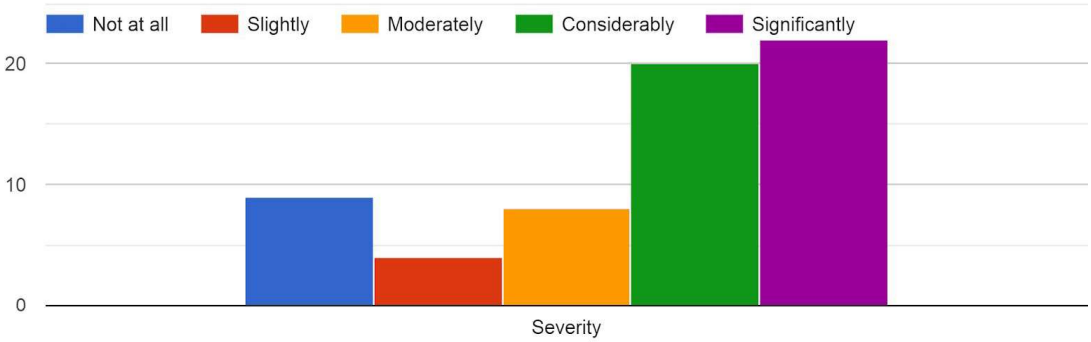
6. Multiple Projects

Retaining focus could become difficult for a Scrum team if the team is working on multiple projects at the same time.

How often do you experience this challenge?



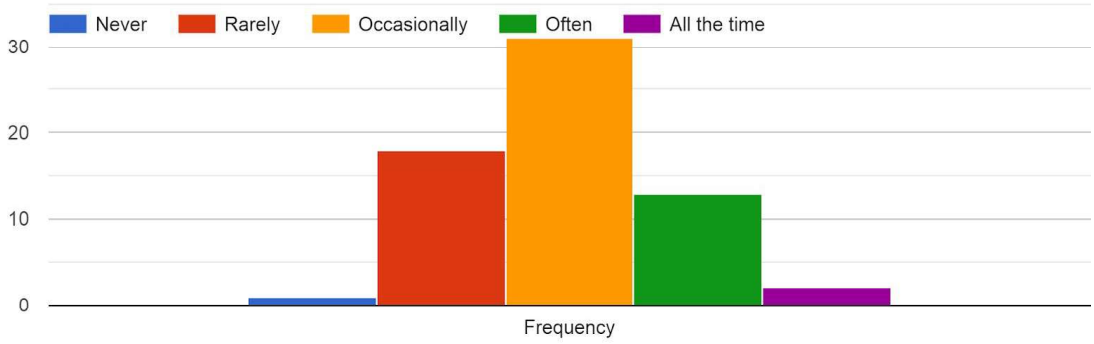
How threatening can this be to a Scrum team's ability to be self-organizing?



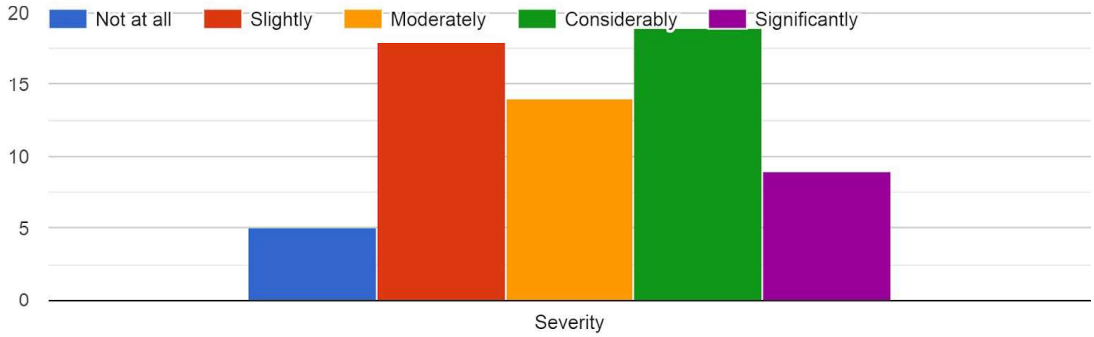
7. Personality Clashes

This challenge relates to conflicts resulting from different personalities in the Scrum team.

How often do you experience this challenge?



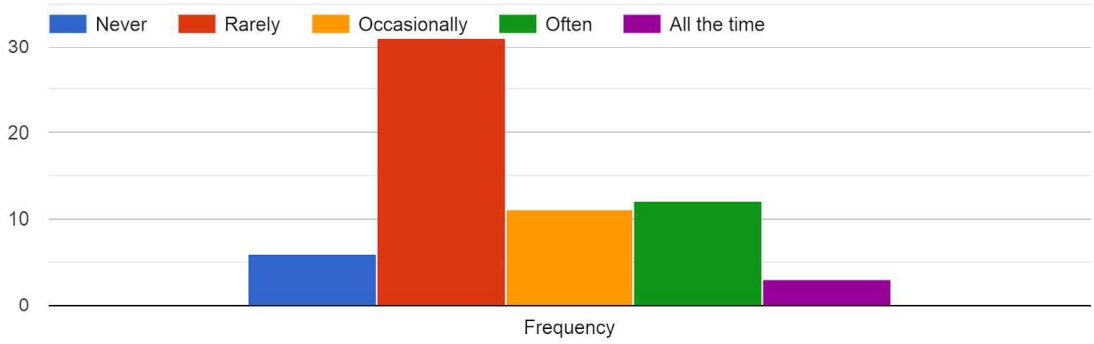
How threatening can this be to a Scrum team's ability to be self-organizing?



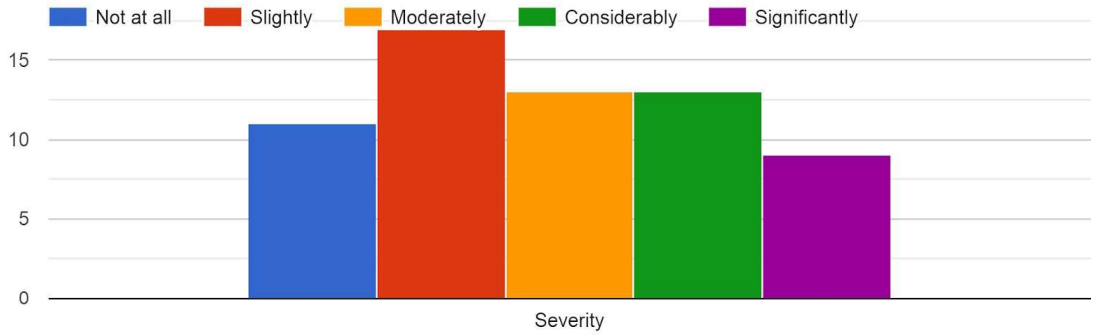
8. Division of Tasks

This challenge relates to the self-organizing team struggling to divide tasks appropriately. Efficiency drops if an inexperienced team member is assigned a task that is too difficult. The same happens if a highly experienced member is assigned a task that is too easy for them.

How often do you experience this challenge?



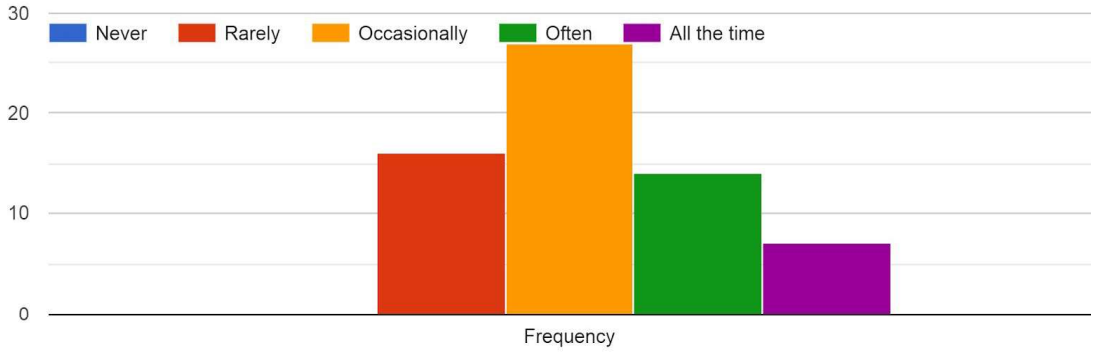
How threatening can this be to a Scrum team's ability to be self-organizing?



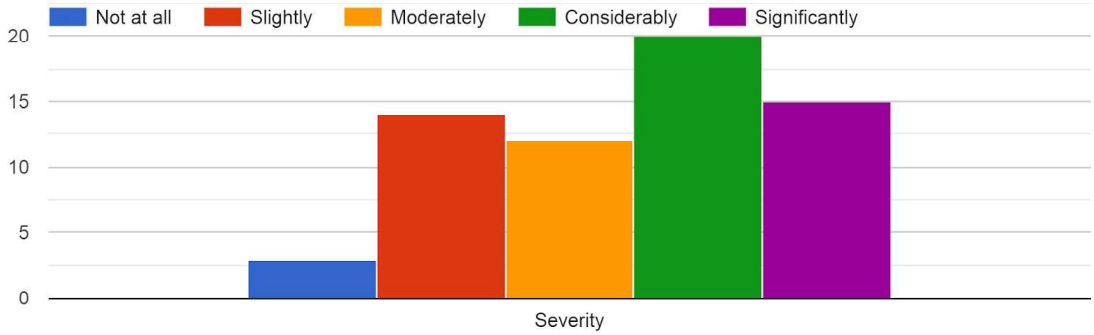
9. Drop in Quality and Technical Debt

This challenge relates to teams rushing work in order to meet a sprint goal and thus compromising quality of the end-product. This means that the team has to go back and fix 'done' work due to possible errors or oversights.

How often do you experience this challenge?



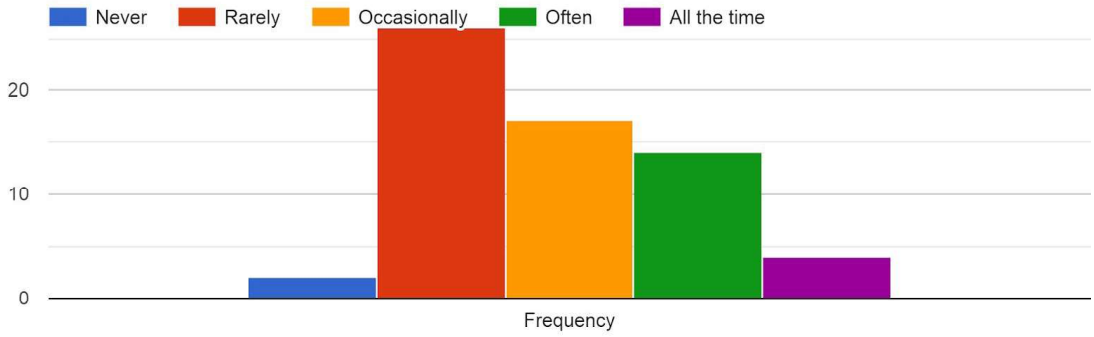
How threatening can this be to a Scrum team's ability to be self-organizing?



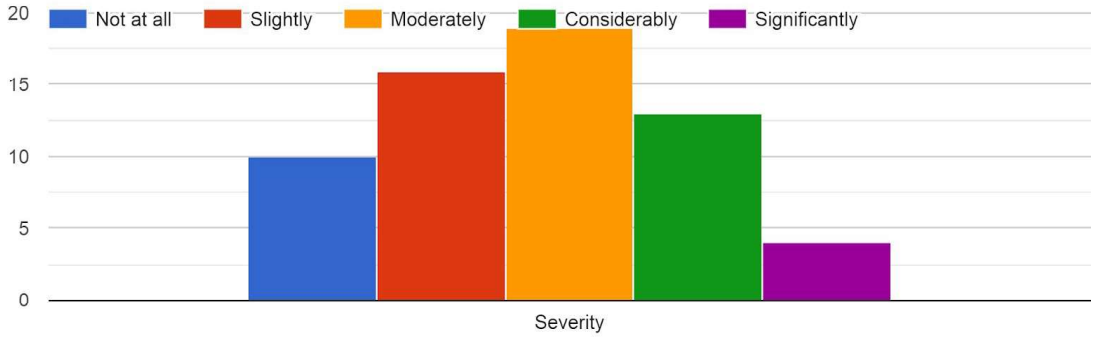
10. Lack of Valuable Estimations

This challenge relates to difficulties or disagreements about the estimation of tasks/user stories in the Scrum team. The outcome is highly inaccurate estimates or potential team conflicts.

How often do you experience this challenge?



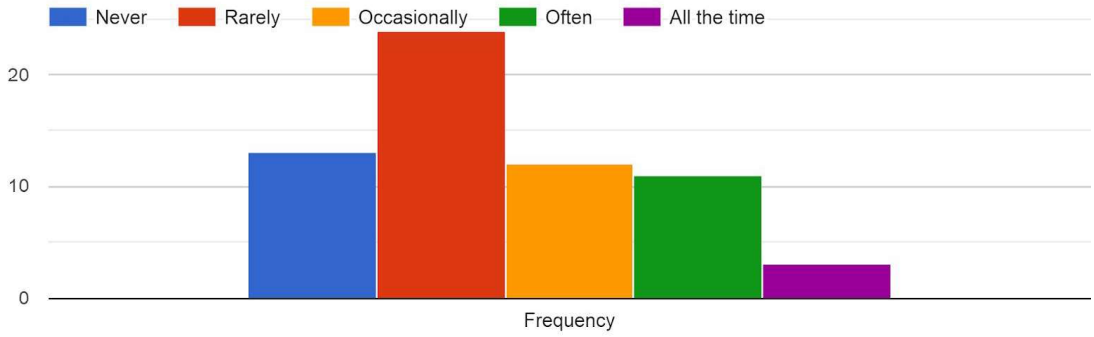
How threatening can this be to a Scrum team's ability to be self-organizing?



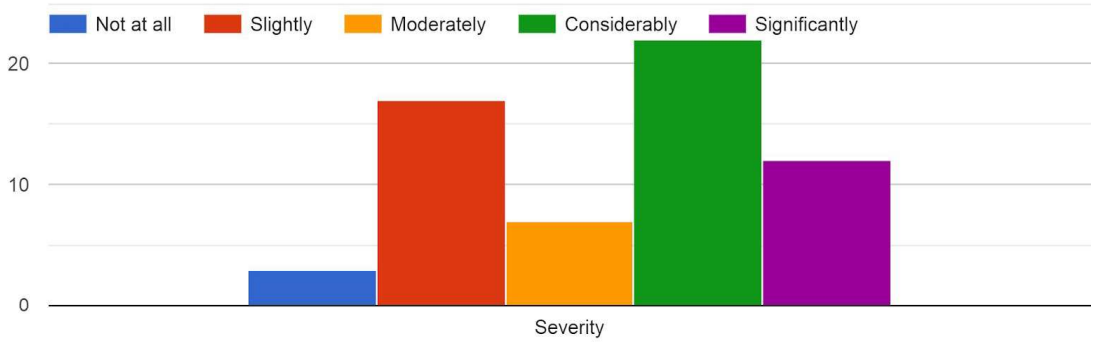
11. Lack of Individual Accountability

This challenge relates to the fact that accountability and credit belong to the self-organizing Scrum team. A team member's failure or success is attributed to the whole team, something that can be seen as unfair by some team members or yourself.

How often do you experience this challenge?



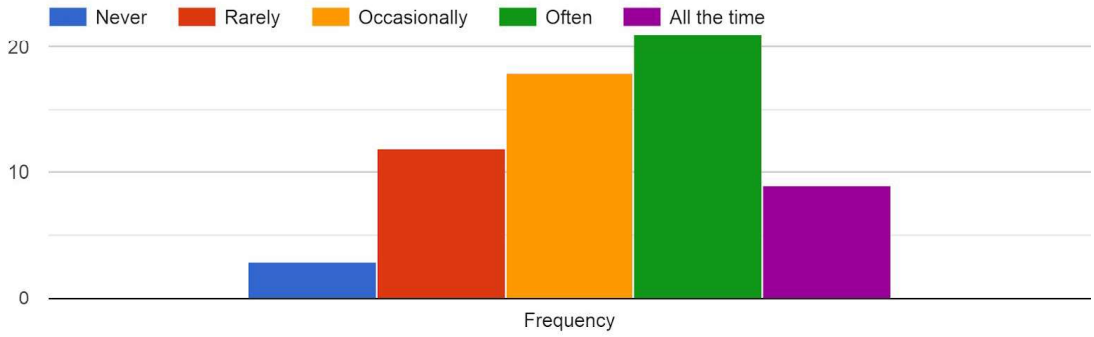
How threatening can this be to a Scrum team's ability to be self-organizing?



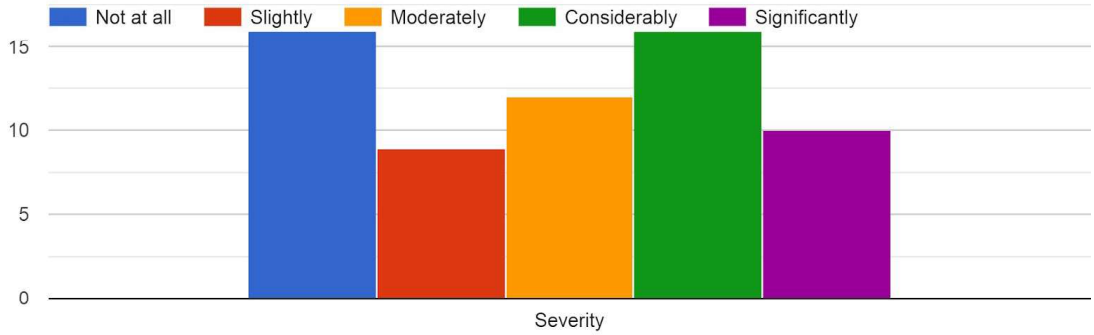
12. Delayed and Changing Requirements

This challenge relates to a change or delay of acceptance criteria or requirements from clients or other parts of the organization.

How often do you experience this challenge?



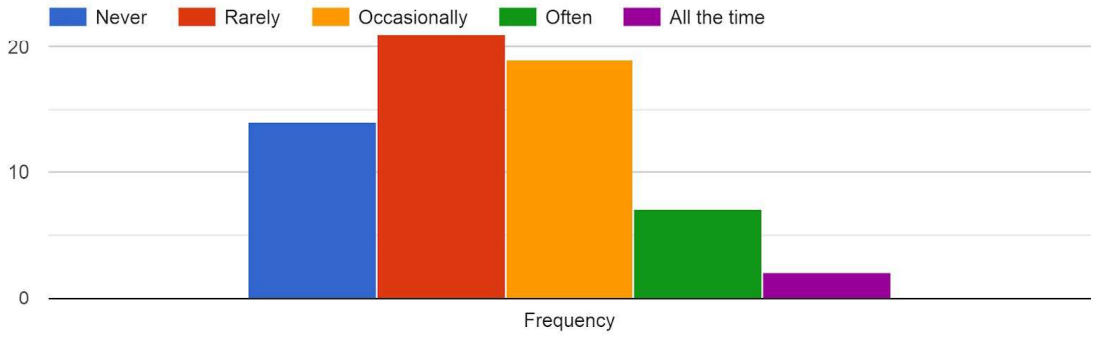
How threatening can this be to a Scrum team's ability to be self-organizing?



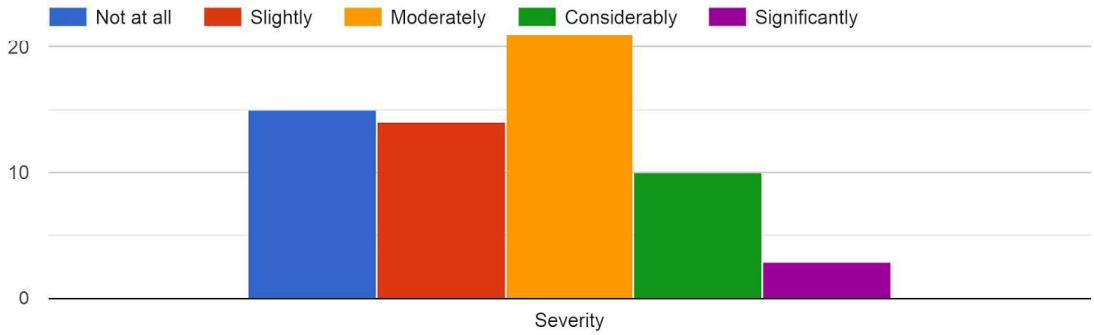
13. Excessive Documentation Requirements

This relates to top management demanding too much documentation which is something that goes against the agile nature of Scrum.

How often do you experience this challenge?



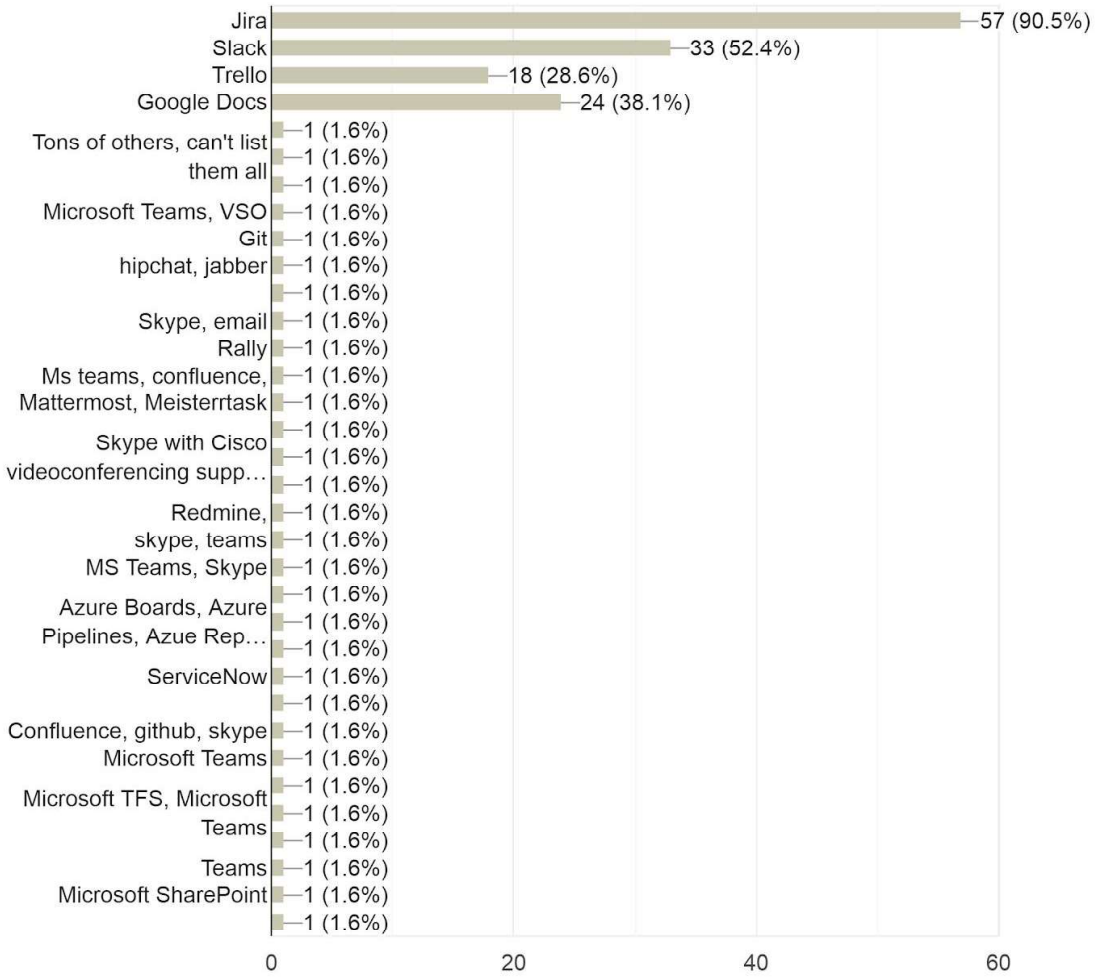
How threatening can this be to a Scrum team's ability to be self-organizing?



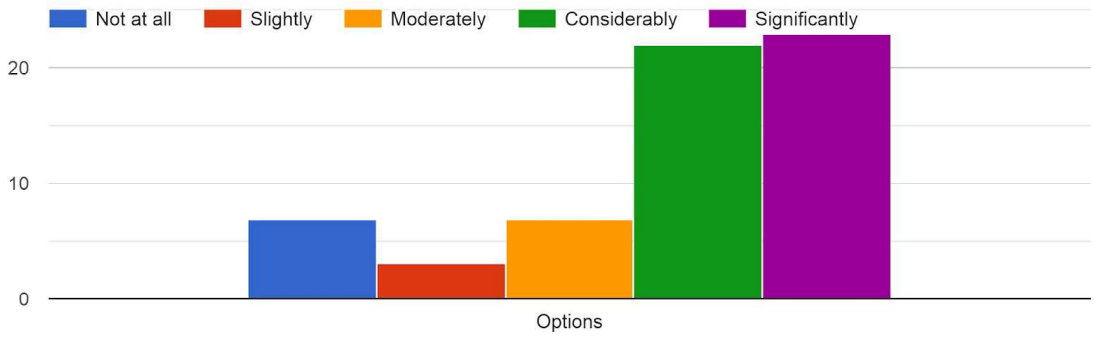
14. Different Perceptions of Communication Software

What communication software tools do you use in your work?

63 responses

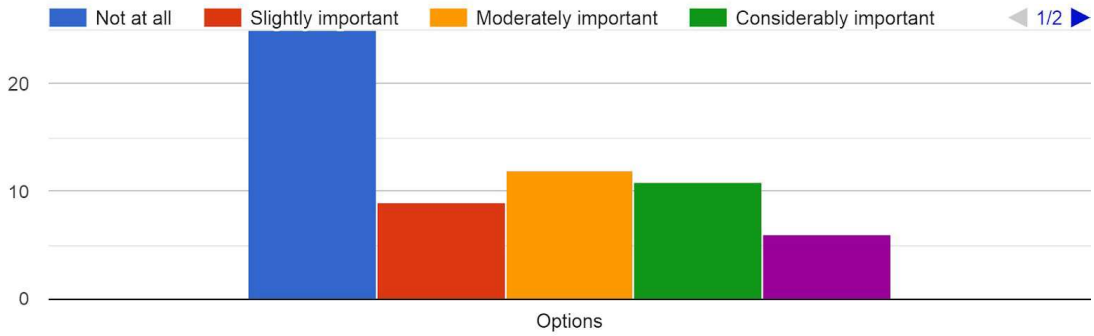


Do you believe they enhance communication and help you become a more self-organizing team?



15. Varying Stances on Agile Training

How important do you think agile certification is for self-organization?



In your opinion, how much does your organization as a whole value formal agile training?