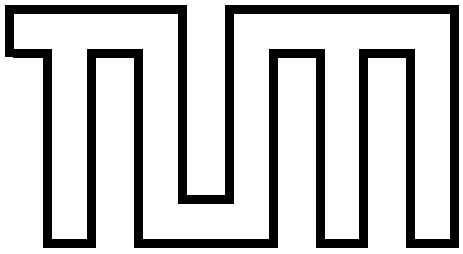


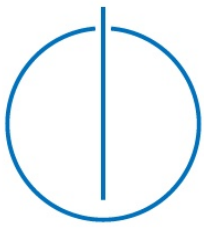
Technical University of Munich

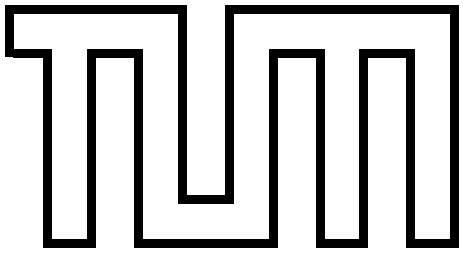
Department of Informatics

Master's Thesis in Information Systems

Applying the Teamwork Quality Model to an Agile Program of a Utility Organization

Manuel Styrsky





Technical University of Munich

Department of Informatics

Master's Thesis in Information Systems

Applying the Teamwork Quality Model to an Agile Program of a Utility Organization

Anwendung des Teamwork Qualitätmodells auf ein agiles Programm eines Energieversorgers

Author: Manuel Styrsky

Supervisor: Prof. Dr. Florian Mattes

Advisor: Phillip Schneider, M. Sc.

Submission: August 16, 2021

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Munich, August 16, 2021

(Manuel Styrsky)

Acknowledgments

First and foremost, I would like to thank my advisor Phillip Schneider for the excellent supportive guidance. You always had an open ear and willingly answered all my questions. You helped me from the beginning to the end.

In addition, I would like to thank Ömer Uludağ, who suggested the topic of the thesis long before I actually started. You were always able to help me professionally and always had good advice for me.

I would also like to thank Professor Dr. Florian Matthes for giving me the opportunity to write my master thesis at his chair of Software Engineering for Business Information Systems (SEBIS). You provided very valuable feedback early in the research process and during my first presentation, helping me to shape the topic of this thesis.

Furthermore, I would like to thank Nina Nürnberger and Kerry Walder as my supervisors and mentors from OPITZ CONSULTING Deutschland GmbH from the bottom of my heart. You accompanied me from the beginning, convinced the case study partner of our research project, and were always there for me - whether for feedback, new impulses, or moral support. Thank you very much for your valuable time.

In addition, I would like to thank all the people from the case study partner who supported me in the preparations, were available for interviews, or willingly participated in the surveys. Thank you all very much. Without you, none of this would have been possible.

Furthermore, I would like to thank Torgeir Dingsøyr and Gunnar Bergersen for their support, feedback, and interest in my work during my thesis writing. It was a pleasure to work with you together on this exciting topic.

Last but not least, I would like to thank all family members, friends, colleagues, and lecturers who inspired me, from whom I was able to learn, who constantly helped me, or who were simply there for me and always had an open ear. Without you, I would not stand where I am today.

Abstract

Agile frameworks and methods are gaining importance, especially in software development. Thereby, agility is a means of reducing risks in uncertain and volatile environments where changes are constantly taking place. In contrast to traditional management practices, agile frameworks and methods rely on self-organization and leadership instead of tight control mechanisms and rigid processes. This enables teams to react quickly and adequately to changes, even during project implementation.

What originally started with small teams and projects is now being used for large programs with multiple teams. This makes it increasingly difficult for managers to assess the program's performance and whether the program is on the right track. Since agile methods build on teamwork, one of the most critical factors for good project performance may be good teamwork. Teamwork quality in agile programs is the center of this research. Based on the findings in the literature on scaling agile, teamwork, and the Teamwork Quality model (TWQ model), this master thesis tries to apply the TWQ model to the program level of agile programs.

A case study and two surveys are conducted at a program of a utility company. In addition to the data from this utility company, the data is aggregated with data from a similar previous case study at a software development company in the finance sector. The results show that the TWQ model is applicable to the program level of large-scale agile software development programs. With this, TWQ seems to influence positively the success and performance assessed by team members. Furthermore, there is also a slightly positive influence of TWQ on the performance assessments of stakeholders. However, there is a negative effect of TWQ on the performance assessments of Scrum Masters, which was identified in previous research in a somewhat similar way.

Even though this thesis shows that the TWQ model can be applied to large-scale agile software development programs, more research is needed to confirm this with higher reliability and to explain the reason for the negative influence of TWQ on the performance ratings of Scrum Masters.

Contents

Chapter 1

Introduction

This chapter discusses the relevancy of team performance in scaled agile software development programs for theory and practice. In addition, the chapter discusses the specific research questions to be answered and describes the research approach that will be used to answer these research questions.

1.1 Motivation

Increasing Importance of IT

Information technology (IT) and software development are becoming increasingly crucial for companies. However, digitization [6] is nowadays having a disruptive effect on all kinds of companies, which results in the fact that continuing with analog business alone is often no longer economically competitive [89].
For companies, this is both a risk and an opportunity at the same time. While there is a permanent risk of being squeezed out of the market, many new opportunities can open up at the same time [88]. For example, digital platforms are a new business model [94], which is not possible without IT. In the field of Industry 4.0, IT is also indispensable in production and provides considerable potential for savings [88]. Marketing and sales can also benefit from IT to an increasing extent, thereby improving sales success while at the same time reducing costs through extensive automatization [90].
These are only a few of many examples and opportunities, which are driven by technological trends like distributed cloud, hyper-automation, and artificial intelligence [11].
Although the are many opportunities for digitization, companies need skilled personnel resources to adopt them.

Lack of Resources

The development of new software solutions is challenging for companies because there is a major shortage of qualified staff in the IT sector. Bitkom conducts representative studies in this field every year, surveying more than 800 managing directors and HR managers. In 2020, there were 86,000 open positions in the IT sector in Germany, whereby 70% of the respondents identified a shortage of IT staff. In 2019, there were even 124,000 vacancies, whereby 83% of the respondents identified a shortage of IT staff. Bitkom attributes the drop in 2020 to the effects of the corona pandemic [98] and assumes that the increase will continue in the following years. [41, 59]
This lack of resources can make it difficult for companies to participate in competitive markets that are subjected to rapid change.

Increasing Use of Agile Methods

Agility allows dealing with changes during project execution in a quick and successful way. According to the 14th Annual State of Agile Report [18], 95% of organizations now use agile software development methodologies like Scrum or Kanban. These high adaptation rates may, among other things, be due to the fact that projects executed using agile methods are significantly more successful than those executed using traditional project management approaches [53]. In project situations with growing software complexity and increasing uncertainties regarding both requirements and technologies, agile project management approaches are particularly effective, for example, in reducing software delivery times, dealing with changing priorities, increasing productivity, and improving software quality [18].
As the original agile methods are only designed for small projects with one team, various scaling agile software development approaches, frameworks, and practices emerged to benefit from the promising agile methods in larger projects. Uludağ et al. [87] identified over 20 scaling agile frameworks like Scaled Agile Framework (SAFe), Large Scale Scrum (LeSS), or Nexus in their study. The most widely used, however, is SAFe [18, 19]. This one and the other most important scaling agile frameworks are described in Section 2.3.

Research Gap: Performance in Scaled Agile Software Development

In the context of the increasing importance of IT for all types of companies with simultaneous resource scarcity, companies need to use the existing resources as efficiently and effectively as possible to develop good software solutions and remain competitive.
While team performance and its influencing factors in single teams has been extensively researched, there are only a limited number of scientific publications on what influences the performance and success of projects in environments in which multiple teams work together in close collaboration. With close collaboration, as required by agile methods, it is reasonable to

assume that the quality of teamwork also plays a decisive role in the success of agile projects. This assumption has already been verified with the Teamwork Quality model (TWQ model) for classic [36] and agile teams [49] in several studies, for example, these from Högl, Lindsjørn, and Dingsøyr. TWQ model describes the relationship between teamwork quality and project success (more details can be found in Section 2.4). Even so, the question remains as to the relationship between the quality of teamwork and the success of large-scale agile programs. Therefore, this thesis aims to investigate (1) whether the TWQ model can also be applied to large-scale agile programs, (2) how teamwork affects the performance and success of large-scale agile programs, (3) and which indicators of teamwork are particularly crucial for large-scale agile programs. Furthermore, the thesis shall explore (4) how a utility company sets up and performs a large-scale agile software development program.

1.2 Research Questions

To achieve the research goal outlined in the previous section, this master thesis aims to answer the following research questions:

Research Question 1:
How is a large-scale agile software development program performed at a utility company?
The answer to this question serves three purposes. Firstly, it should provide insights into how a utility company performs a large-scale agile software development program. Secondly, the answer will help to place the survey results, and the application of the TWQ model to the data in the context of the investigated large-scale agile software development program. Lastly, this should give interested third parties enough information about the case study partner to compare the results with further studies.
To answer this question, several semi-structured interviews will be conducted with Product Owners, Agile Coaches, and other key roles from the observed program at the case study partner.

Research Question 2:
Can the TWQ model be applied to the team level of a large-scale agile software development program of a utility organization?
The answer to this research question should ensure that the TWQ model is also applicable to the program of the case study partner. In addition, the significance of the individual indicators of TWQ in the TWQ model is investigated.
In order to answer this question, the data of the survey, which is conducted on the team level of the case study partner's program, will be analyzed using structural equation modeling (SEM) and lavaan in R (for more details, see Section 5.1.3).

Research Question 3:
Can the TWQ model be applied to the program level of large-scale agile software development programs?
This is the key question of this research. The intention is to find out to what extent the TWQ model can be applied to the program level of large-scale agile software development programs. The answer to this question should also provide information about how the indicators of TWQ relate at the program level.
To answer this question, the data from the surveys at the program level of this study and the one of Doepp [24] will be merged. The resulting larger dataset allows for more reliable results regarding the influence of TWQ on performance and success at the program level.

Research Question 4:
What are commonalities and differences between the TWQ models at team and program level?
This question intends to provide insights into the commonalities and differences between the application of the TWQ model on the program and team level by directly comparing the descriptive statistics as well as the results of the application of the TWQ model on the data from the two different levels.
To answer this question, the results of the analysis from the merged data from both studies, this one and the one of Doepp [24], at the team and program level will be used and compared.

1.3 Research Approach

In order to answer the research questions mentioned above, this thesis uses a mixed-methods exploratory research design [15, 83]. This approach combines a case study of a large-scale agile software development program at a utility company that wishes to remain anonymous and two surveys at the same program. Furthermore, the survey data of a similar study from Doepp [24] is added to have a more extensive data set, which allows for more reliable results.
The research approach combines a case study and a survey. It is divided into two parts: The first part consists of several semi-structured interviews conducted to understand how the program works and its setup. In addition, challenges and success factors are also investigated. This part of the research is used to answer the first research question.
The second part consists of the surveys conducted at the team and program level of the case study partner's program. The survey data is augmented with data from the case study by Doepp [24] at a finance company to obtain more reliable conclusions at the program level. Overall, the survey data is used to answer research questions two through four regarding the application of the TWQ model (see Section 2.4) to the case organization and the program level of scaled agile programs. Therefore SEM is used, as it allows estimating and testing correlations between dependent and independent variables as well as the hidden structures in between. Figure 1.1 visualizes this mixed-methods approach and shows which data is used for which research question.

The thesis describes the basic concepts, the research approach, and the results of the research and is structured as follows:
Chapter 2 describes the foundations in the field of agile software development and scaling agile software development, which are essential to understand the results of the research and their context. Chapter 3 reviews what relevant research has been conducted in the field of TWQ model, large-scale agile software development, and team performance so far. Chapter 4 describes the methodological approach and the results of the interviews with the case study partners and serves as a foundation for answering the first research question. Chapter 5 describes the methodological approach of the survey, the analytical approach of the data analysis, and the results of the data analysis, as well as the application of the TWQ model. This serves as the basis for answering research questions two through four. Chapter 6 discusses the findings of both the interviews and the survey, answers the research questions, identifies the limitations of these research findings, and embeds them in the context of other research. Chapter 7 summarizes the findings and shows where there are still research gaps that should be addressed in further research.

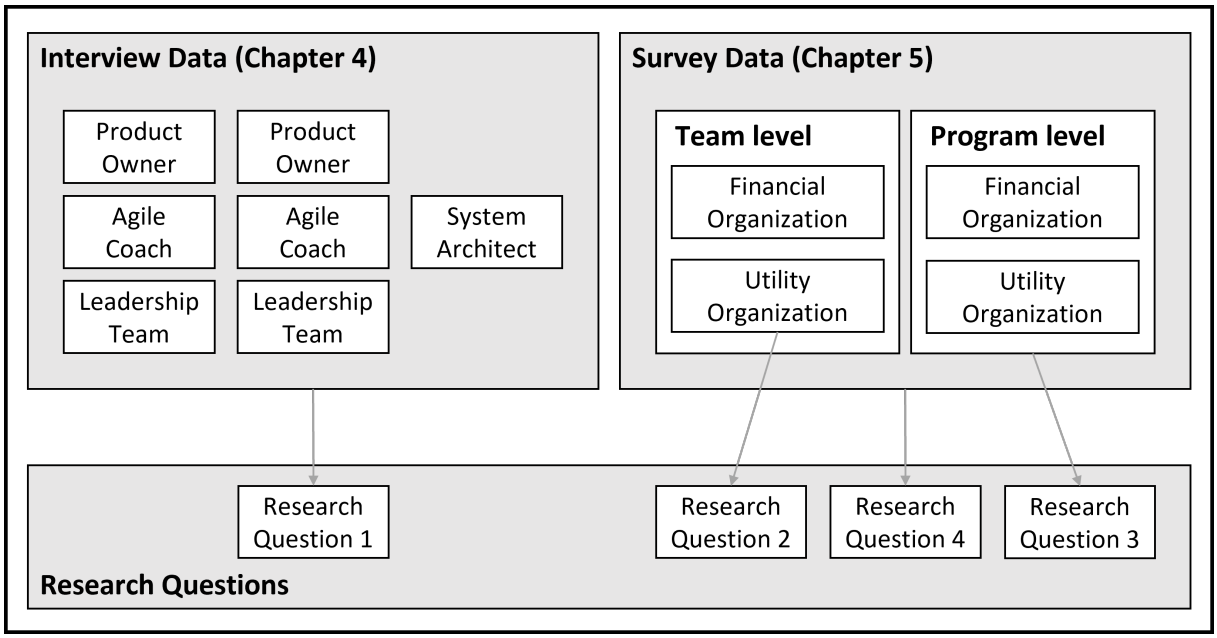


Figure 1.1: The visualization of the research approach showing which method and data is used to answer which of the research questions

Chapter 2

Foundations

This chapter describes and explains the foundations in the field of agility, scaled agile software development, and the TWQ model, which are necessary for understanding and interpreting the following chapters, the research approach, and the results.

2.1 Agile Software Development

Over a long time, traditional project management methods were used in software development. Popular representatives are the waterfall model [64], the spiral model [8], PRINCE2 [40], and in Germany, the V-Model [10]. These models are all plan-based and have limited ability to work with changes during project execution. All the requirements and technologies should be known before the project starts. Furthermore, the lead-time and the amount of rework is very high [14]. In project situations with growing software complexity and increasing uncertainties, these traditional models are less and less suitable (compare Figure 2.1). Instead, agile methods can be applied in settings with high uncertainties regarding requirements and technology [68]. Mersiono [53] states in his study that agile projects have a two times higher success rate than traditional executed projects.
This higher success rate is one of the reasons why a wide variety of agile methods and frameworks arose. The most common approaches are the Scrum framework (see Section 2.1.2) and Scrum-related methods [18]. Even though, earlier other methods like extreme programming [4], or the dynamic system development method (DSDM) [77] were often used too [17]. Even if most of these methods were formally described around 2000, the development of the agile methods begun much earlier (see Scrum in Section 2.1.2).
Currently, 95% of the companies are using agile methods, whereas, in 51% of the companies, more than half of their teams are agile [18].

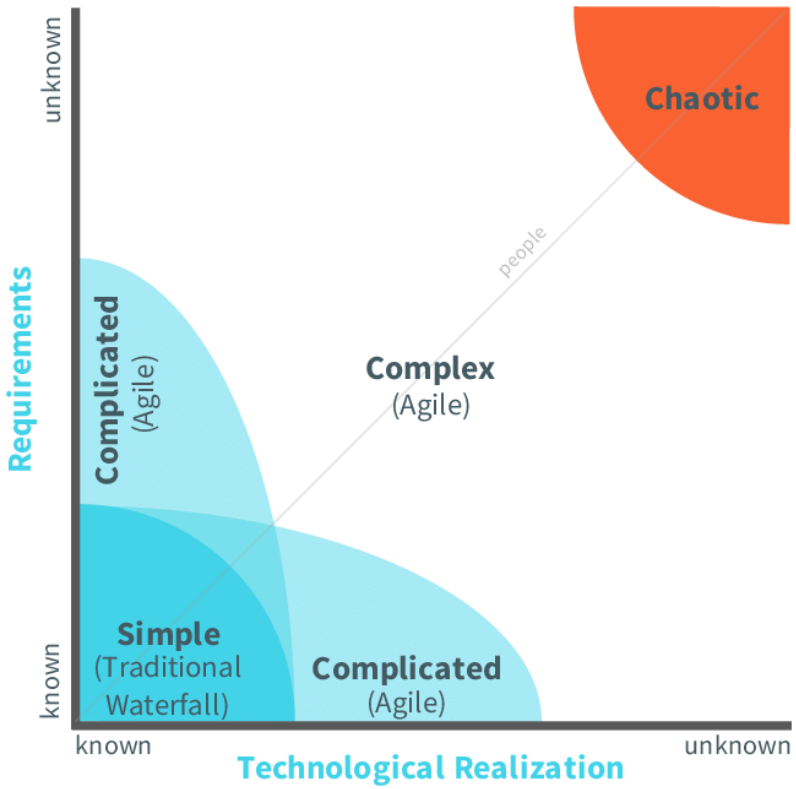


Figure 2.1: The Stacey Matrix [76] adapted to software development [86]

2.1.1 Manifesto for Agile Software Development

Note: All the information in this section is taken from the agile manifesto [54].
The agile frameworks and methods are based on values and principles. These are designed to guide teams when they need to make decisions and are the foundation for any agile implementation. The principles and values are described in the agile manifesto. It was created while 17 developers using different agile methods like Scrum, DSDM, feature-driven development (FDD), and many more came together to find common ground of their experiences and methods. This resulted in the agile manifesto - a set of 4 values and 12 principles.
The manifesto therefore states:

> We are uncovering better ways of developing software by doing it and helping others do it. Through this work, we have come to value:
>
> | | | |
> |---|---|---|
> | **Individuals and interactions** | over | processes and tools |
> | **Working software** | over | comprehensive documentation |
> | **Customer collaboration** | over | contract negotiation |
> | **Responding to change** | over | following a plan |

That is, while there is value in the items on the right, we value the items on the left more.

The 12 principles of the agile manifesto make the fore values more tangible: [54]

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity–the art of maximizing the amount of work not done–is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

2.1.2 Scrum

Since Scrum is the most widely used agile project management approach [18] and many agile scaling frameworks, such as SAFe [44], are based on it, this section describes the fundamentals of Scrum. Unless otherwise stated, the information is taken from the current version of the Scrum Guide [70].

Scrum is a framework to develop products in complex environments. Even if the framework is simple to understand, it is hard to master. This comes from the fact that it is incomplete by purpose, which allows users to use various processes, techniques, and methods.

History of Scrum

According to Schwaber and Sutherland, the development of the Scrum framework started in the early 1990s, and they presented it in 1995 at a conference [67]. Although the term Scrum itself came up even earlier in 1986 (Takeuchi and Nonaka, [81]), the first Scrum Guide was published in 2010 [71]. Since then, several minor and major changes took place in five updates. The latest update is from November 2020. Overall of the updates, the core of Scrum was not changed.

The Scrum Process

The Scrum process itself is iterative and incremental. This means that the work is done in several iterations, called *Sprints*. In each of these Sprints, a new so-called *Product Increment* is created, which creates incremental value. A Sprint starts with the *Sprint Planning*. Within this, the whole Scrum team collaboratively creates a plan on how to perform the work to be done in the next Sprint. This plan is called *Sprint Backlog*. Therefore, the following questions should be answered during the Sprint Planning:

- Why is this Sprint valuable?
- What can be done this Sprint?
- How will the chosen work get done?

Every day during the Sprint, the team meets for the *Daily Scrum*, which is a short, 15-minute time-boxed event. The purpose of this event is to inspect the progress and adjust the plan if necessary. At the end of a Sprint, the Scrum team meets with the stakeholders for the *Sprint Review*. Hereby, the Scrum Team presents the Product Increment, and together with the stakeholders, future adaptions are determined. The last event of each Sprint is the *Retrospective*. As a result, the whole Scrum team develops a plan to improve their quality and effectiveness. All the Scrum events are time-boxed, which gives them a maximum amount of time, which they must not exceed.
A visual representation of the Scrum flow is visualized in Figure 2.2.

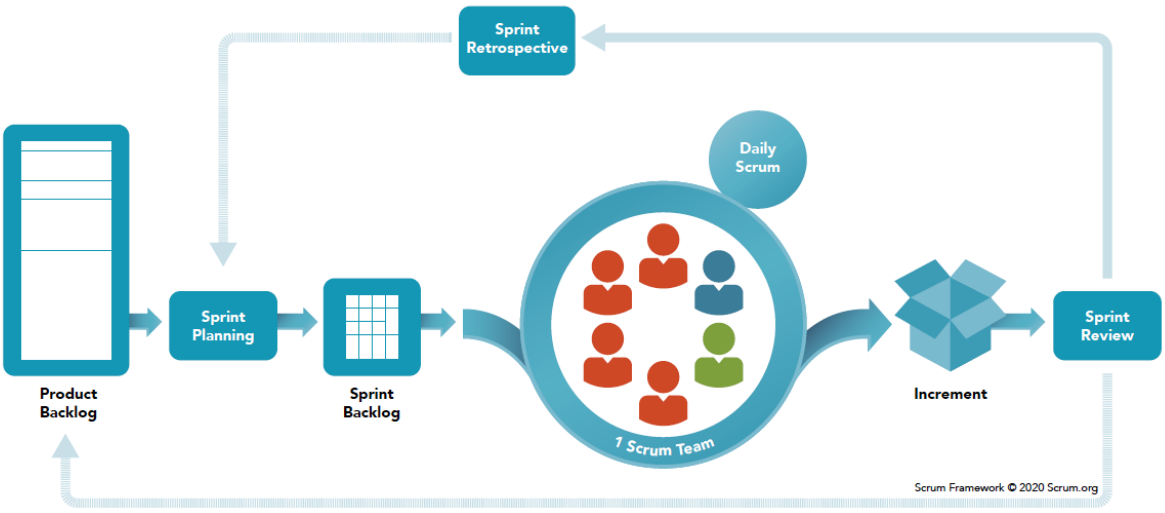


Figure 2.2: The Process flow of the Scrum Framework [72]

Accountabilities and Artifacts

Scrum defines three accountabilities (earlier roles [69]):

- *Developers* (earlier Development Team [69])
  The Developers commit to creating a usable increment in each Sprint. The needed skills may be broad and depend on the domain of work.
- *Product Owner*
  The Product Owner has to maximize the product's value and is accountable for effectively managing the Product Backlog.
- *Scrum Master*
  The Scrum Master is defined as a true leader and is accountable for establishing Scrum according to the Scrum Guide. He serves the Scrum Team, the Product Owner, and the organization in several ways.

Furthermore, there are three artifacts defined by Scrum which represent value. Each of the artifacts has a commitment. The artifacts and their commitments are:

- *Product Backlog*
  The Product Backlog is an ordered list of the work the Scrum team should provide. It is the single source of work, and it may never be complete. The commitment for the Product Backlog is the *Product Goal*, which describes the long-term objective for the whole Scrum Team.

- *Sprint Backlog*
  The Sprint Backlog is a set of Product Backlog items selected for the Sprint combined with a plan to deliver the next Product Increment. The commitment is the *Sprint Goal*, which is also part of the Sprint Backlog and serves as the objective for the Sprint.
- *Increment*
  The Increment is a step towards the Product Goal. It adds to all prior Increments. The *Definition of Done* is the commitment of the Increment and describes the quality measures it must meet.

Empirical Process Control

Scrum is based on empiricism, which asserts that "knowledge comes from experience and making decisions based on what is observed" [70]. Therefore, Scrum defines three pillars of the empirical process control that build on each other as follows:

- The first pillar is *Transparency*. Important decisions are based on previous states of the three artifacts. These and the process must be transparent to all team members and stakeholders to make good decisions.
- Transparency enables *Inspection*. To detect potential problems and risks early, the progress must be inspected frequently.
- Inspection enables *Adaption*. If the inspection reveals undesired process or product deviations, the team should adjust to minimize this deviation in the future.

All events of Scrum provide the opportunity for inspection and adaption.

2.2 Agile Related Concepts

There are many concepts that are indirectly as well as directly related to agility. The following section describes those concepts that are necessary for understanding the following chapters, but especially for SAFe which the case study partner uses.

2.2.1 Servant Leadership

There are different types of leadership in literature and practice. Agile methods often refer to servant leadership (see previous version of the Scrum Guide [71], or SAFe [44]). Hence, this section will describe servant leadership to get an understanding of its style and characteristics.

The term Servant Leadership was first coined by Greenleaf [31], who describes a servant leader as somebody who puts the needs, interests, and aspirations of others above his own ones and wants to serve first instead of leading or owning others. The motive of a servant leader is to

serve others to grow and to be more autonomous. [31]
As this definition of Greenleaf is not very accurate and leaves room for interpretation, many models appeared in the literature. Spears was one of the first to develop a model based on Greenleaf's findings. He differentiated ten essential characteristics of servant leaders [75]. The model of Russell and Stone [66] defines nine functional and 11 additional characteristics of servant leadership.
These two are only examples among others. Dierendonck [91] tries to conceptualize the different models and characteristics of servant leadership in literature in one overall model, which is visualized in Figure 2.3. His model defines the core characteristics of servant leadership as (1) empowering and developing people, (2) humility, (3) authenticity, (4) interpersonal acceptance, (5) providing direction, and (6) stewardship. However, the model goes even further. It also describes the antecedents (inked green), mediators (inked orange), and outcomes (inked blue) of servant leadership.

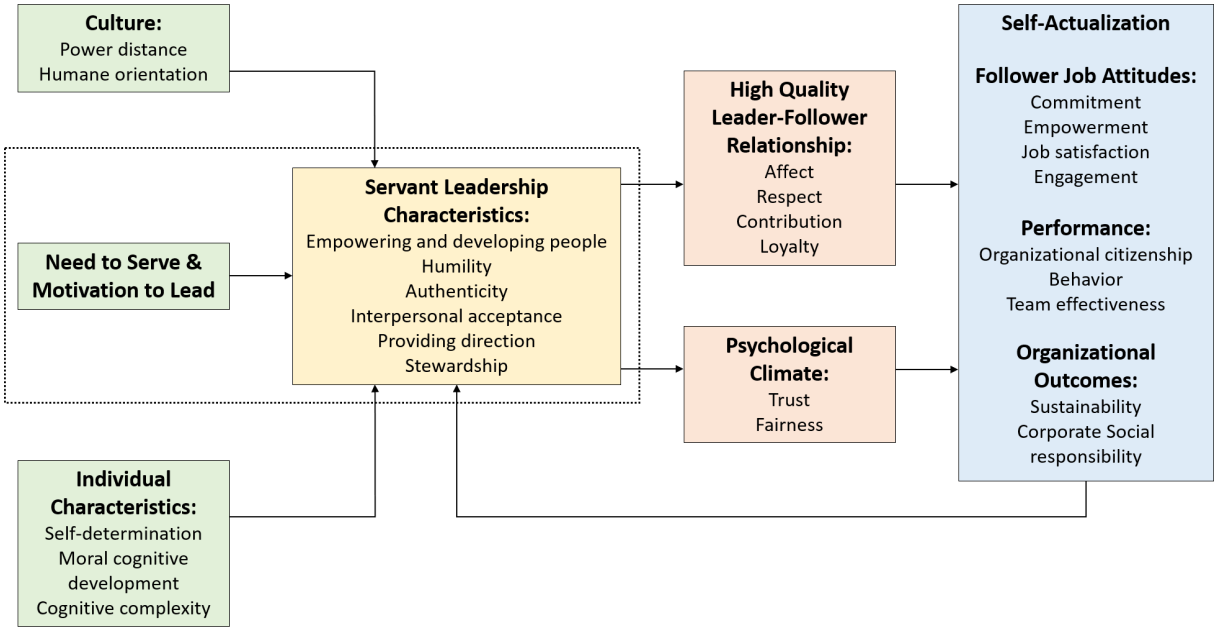


Figure 2.3: The conceptual servant leadership model of Dierendonck [91]
Coloring: green - antecedents; orange - mediators; blue - outcomes; yellow - characteristics
(Visualization recreated with PowerPoint)

2.2.2 Lean

The term Lean first appeared in 1991 in the context of Lean Manufacturing. Womack et al. [97] compared in their book Japanese and American companies. In 1996 they introduced the term Lean Management [96]. Later, in 2003, Poppendieck and Poppendieck [62] applied the concepts of Lean Manufacturing to software engineering. Petersen [60] compared the concepts of agile and lean software development. He found out that:

- Both, agile and lean, share the same goals: Respond to change, reduce waste, and focus on customers' needs.
- Both define similar principles to reach the defined goals. E.g., continuous improvement.
- In comparison to agile, lean does not define processes and only relies on principles and paradigms.

2.2.3 DevOps

There is no clear and unique definition of DevOps in literature. Lwakatare et al. [51] conceptualize DevOps as a model of the four dimensions collaboration, automation, measurement, and monitoring. This model is similar to the results of the literature review conducted by Erich et al. [26]. They clustered the literature in culture of collaboration between development and operations, automation of software processes, combined measurements of development and operations, sharing of information, and three more categories. Bass et al. [3], however, define DevOps as following:

> "DevOps is a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality."

All these definitions support the principles behind the agile manifesto (See the principles 1, 3, 7, 9 in section 2.1.1).
The most important cultural change, however, is the increase of collaboration between software development and operations. The most important technical practice is the use of continuous integration and continuous deployment tools. These tools can automate software processes like building, testing, and deploying the software and can furthermore enable the immediate placement of changes into productive software systems. [3]
According to the 15th Annual State of Agile Report, 74% of having already or are planning a DevOps initiative [19].

2.3 Large-Scale Agile Software Development

Initially, agile methods and frameworks like Scrum [70] or Extreme Programming [4] were designed for single teams. However, a single team has a limited amount of work capacity. One solution might be to add more people to the team, but then the individual performance decreases. Wheelan [95] investigated the effect that with increasing group size, the group productivity decreases. Also, Hackman [32] investigated that four to five people is the optimal team size based on the productivity of the group. Hence, in large projects, the project members must be divided into several teams. Of course, more teams also involve greater effort in terms of inter-team

communication and management roles. This might be a reason why the recommended team size in Scrum or the SAFe is greater than five [70, 45].

Neither the Agile Manifesto [54] nor the Scrum Guide [70] provide explicit instruction on how to scale Scrum or agile methods. The Scrum Guide only states: "If Scrum Teams become too large, they should consider reorganizing into multiple cohesive Scrum Teams, each focused on the same product. Therefore, they should share the same Product Goal, Product Backlog, and Product Owner." [70]
This lack of guidance is why there were plenty of new frameworks developed. Uludağ et al. [87] identified 20 scaling agile frameworks. Based on the 14th Annual State of Agile Report [18], the most used agile scaling approaches are SAFe (35%) and Scrum of Scrums (16%). Other frameworks like LeSS, Disciplined Agile Delivery (DAD), or Nexus have adoption rates below 5%.

The agile scaling frameworks build on the idea of Scrum. As the case organization makes use of SAFe, this will be described in detail in Section 2.3.4. The other mentioned frameworks are described briefly in the following.

2.3.1 Scrum of Scrums

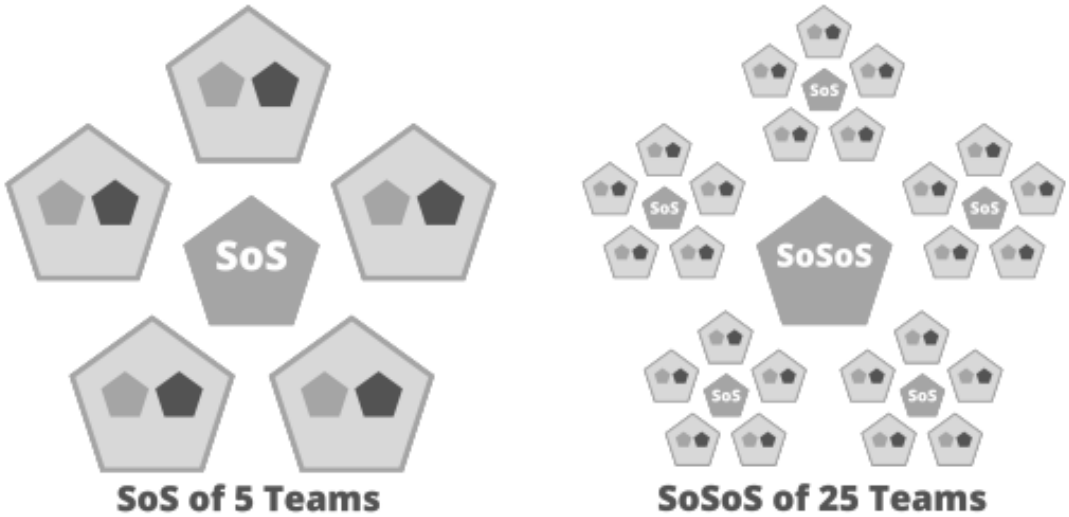


Figure 2.4: Scrum of Scrums constellations for different sizes [80]

The idea of Scrum of Scrums was first mentioned by Sutherland in 2001 [79]. Scrum of Scrums describes the idea of a larger team that consists of several teams and is described in the Scrum@Scale Guide [80]. Figure 2.4 visualizes the idea of several teams building a Scrum of Scrums. If there are even more teams, they build several Scrum of Scrums, which then build a Scrum of Scrum of Scrums.

2.3.2 Nexus

Schwaber, the other inventor of Scrum, developed the agile scaling framework Nexus together with his organization Scrum.org [73]. The main idea of Nexus is the *Nexus Integration Team*, which is an additional team that ensures that in each Sprint, an integrated product increment is generated [55]. Figure 2.5 shows the process flow of Nexus, which is very close to the Scrum process flow in Figure 2.2.

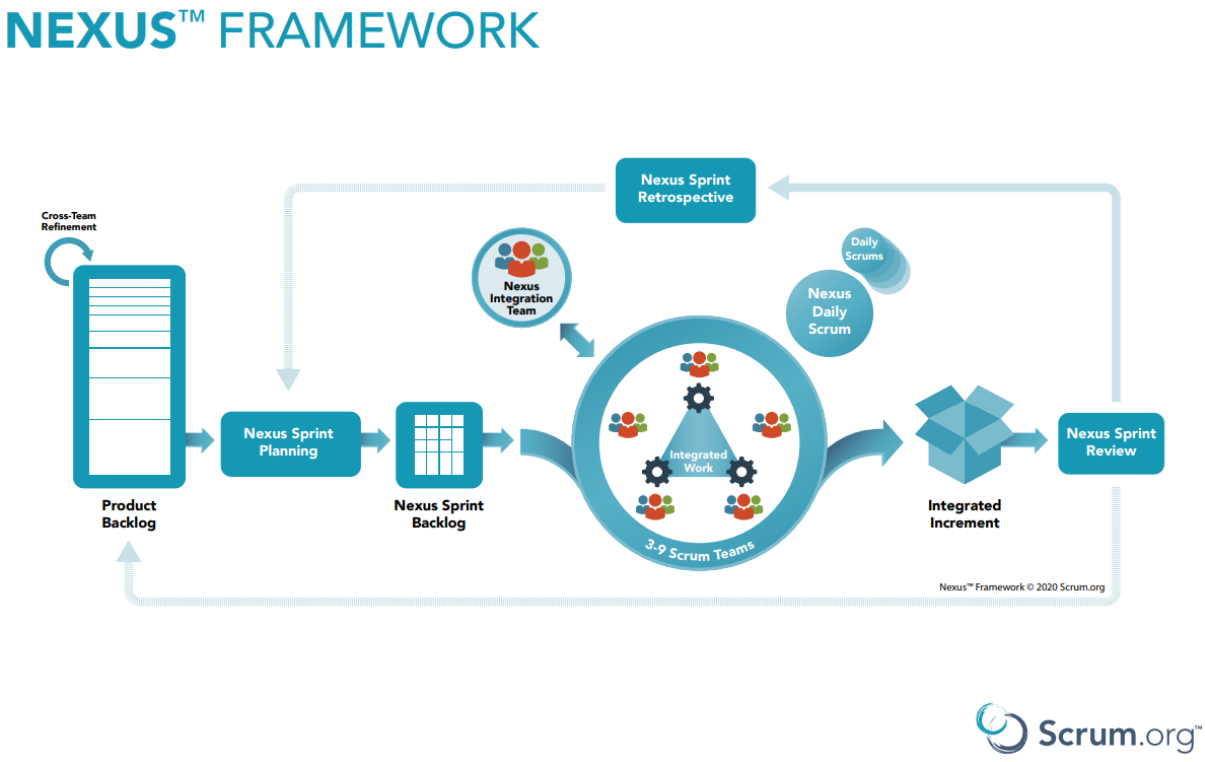


Figure 2.5: The Nexus framework [45]

2.3.3 Large Scale Scrum

Large Scale Scrum (LeSS) was developed by Vodde and Larman in 2005 [92]. This framework is very lightweight and works for up to a few thousand people in the LeSS Huge version. According to the authors, it works like Scrum with one team, and LeSS defines only one Product Backlog, one Product Owner, and one common Product Increment for all the teams. However, there are

some differences compared to Scrum. As one can see in Figure 2.6, there are two Sprint Plannings and two Sprint Retrospectives. One of each is assigned at the team level and one for all teams together [92].

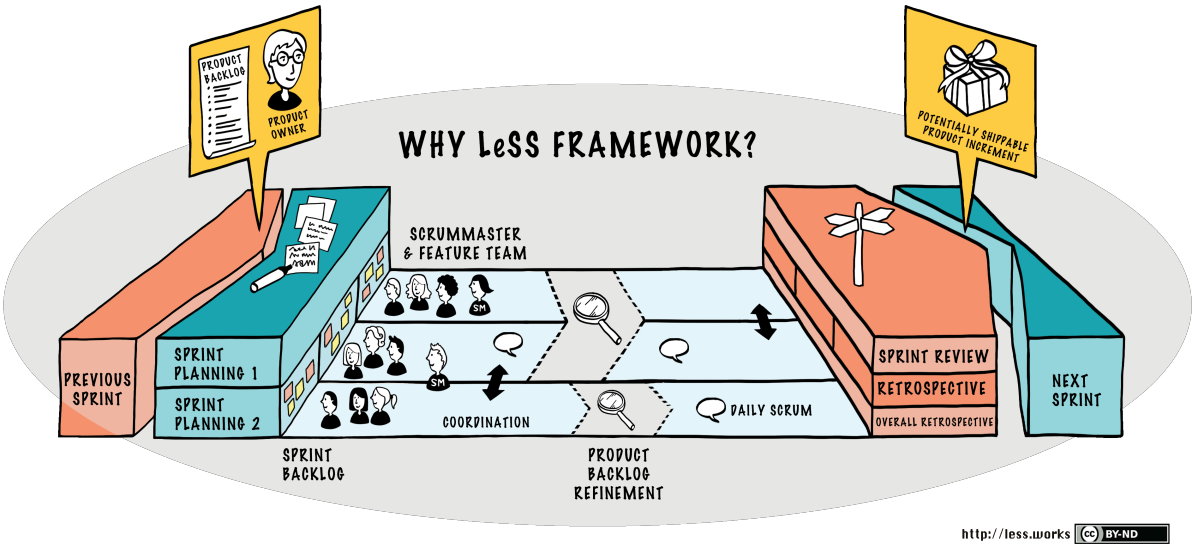


Figure 2.6: The LeSS framework [92]

2.3.4 Scaled Agile Framework

In comparison to the other agile scaling frameworks, Scaled Agile Framework (SAFe) is very detailed and extensively documented. There are four different configurations from which adopters can choose [44]:

- *Essential SAFe* only contains a minimum set of elements needed to employ the SAFe framework. Only the team and the program level are defined.
- *Large Solution SAFe* contains some more elements compared to the Essential configuration. The large solution level with its elements is added.
- *Portfolio SAFe* is the configuration that adds agile portfolio management to the framework. However, the large solution level is excluded.
- *Full SAFe* contains all levels whereby the portfolio level is on top of the Large Solution level. This is the most comprehensive configuration.

The newest version is SAFe 5.1 [44]. However, as the case organization uses the Portfolio configuration of SAFe 4.6[1], this version will be explained in detail in the following sections. A visualization of the framework can be found in Figure 2.7. Furthermore, a version of this

[1] © Scaled Agile, Inc.

framework with the adjustments made by the case organization can be found in Figure 4.1. The information for the following paragraphs about the SAFe 4.6 Portfolio configuration is taken from their website[2] [45].

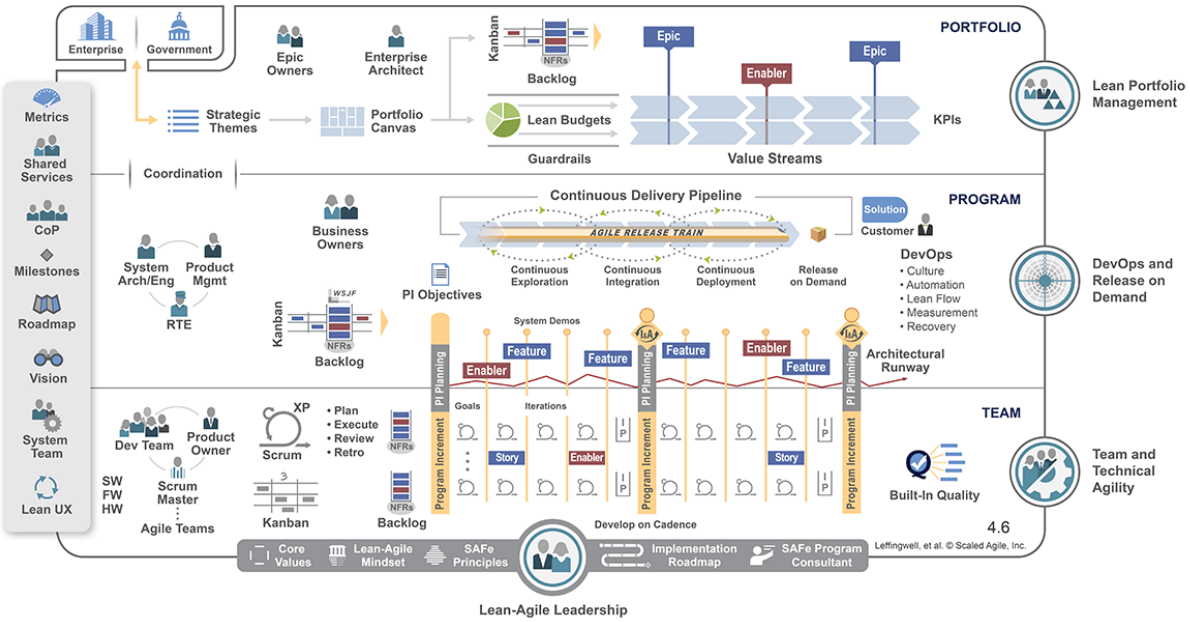


Figure 2.7: The SAFe 4.6 Portfolio Configuration [45]

Team Level

The team level is the bottom level of the SAFe Framework. Hereby, the single teams are set up as Scrum teams with Developers, a Scrum Master, and a Product Owner (see Section 2.1.2 about Scrum). The teams are all cross-functional and self-organized. They are altogether connected via the Agile Release Train (ART) on the portfolio level (see Section 2.3.4). Furthermore, all members of the program can participate in one or more Communities of Practice (CoPs) to exchange ideas with peers in a specific technical or business domain. The teams are not forced to work with Scrum. They could also use Kanban or Extreme Programming (XP), and a mix of several methods is possible too. For example, ScrumXP, which according to the SAFe 4.6 documentation, is the most used method.

Each team has its own *Team Backlog*, which contains stories derived from the Program Backlog (see Section 2.3.4). There are two types of stories defined in SAFe for the team level: Firstly, the *User Stories*, which express the needed functionality. These should be formulated user-centric so that the Developers can understand the users' needs. The second type of stories are the *Enabler Stories*, which can be formulated more technically or informally. These kinds of stories can contain, for example, re-factoring or architectural tasks.

[2] `https://v46.scaledagileframework.com`

Like in Scrum, the teams work in iterations in which they build incremental value. Each of these iterations should have a business and technological goals - the iteration goals. During each Product Increment (see Section 2.3.4), the team has one *Innovation and Planning Iteration*. All of the iterations should include a planning, a review, and a retrospective.
In addition to the agile teams, there is one *System Team*. Even though it is one of the agile teams, it is specialized in setting up and supporting the agile development environment. Its tasks may include setting up a continuous delivery pipeline, a test environment, or helping the agile teams integrate their work.

Program Level

The program level defines the *Agile Release Train (ART)*. This represents all the agile teams and develops and delivers the solution. The ART has its own heartbeat, called *Product Increment (PI)*, like the Iterations at the team level but encompasses more than one iteration of the agile teams. Each PI starts with the *PI Planning*, which is a face-to-face event where all agile teams participate. Together they sharpen their vision of the solution and plan their next Iterations and their PI objectives. At the end of a PI, the whole ART has a *PI System Demo* where the new increment of all the teams is presented to the stakeholders.
The *Program Backlog* contains all the features which deliver value to the customers. Similar to the enabler stories on the team level, enabler features necessary on the program level may be included in the program backlog. Also, the program level defines the *Continuous Delivery Pipeline*. It consists of *Continuous Exploration*, *Continuous Integration*, *Continuous Deployment*, and *Release on Demand*.

The program level adds four more roles:

- The *Release Train Engineer (RTE)* is the Scrum Master equivalent at the portfolio level. He is the agile coach and servant leader for the whole ART.
- The *System Architect* ensures that the ART has a common understanding of the technical and architectural vision. The system architect should bring all the teams to a shared direction and should act as a servant leader (see Section 2.2.1).
- The *Product Management* is the owner of the program backlog and can be a group of people as well. They prioritize the features according to the customer's needs.
- The *Business Owners* are the key stakeholders of the ART. They might have the business and technological responsibility for the solution of the ART regarding governance, compliance, or the return of investment.

Portfolio Level

Ar the portfolio level of SAFe, the enterprise executives define the *Strategic Themes* to enable the alignment between the enterprise strategy and the SAFe program's portfolio. These Strategic Themes represent the business objectives and connect the strategies of business and portfolio. This is facilitated by the *Portfolio Canvas* which is an adoption of the Business Model Canvas [57] to conceptualize the purpose and structure of the portfolio.
The portfolio level enables the agile management of several *Value Streams* which can build one or more solutions. The value streams represent the steps needed to build this solution. SAFe defines two types of value streams: The *Operational Value Streams*, which represent the delivery of value towards the customer, and the *Development Value Streams* which represent the development of new products or services.
The *Lean Budgets* is a set of practices to reduce the funding overhead. These practices are (1) the funding of values streams instead of projects, (2) the balancing of long time and short time investments, and (3) the participatory budgeting where participants from different value streams pool the budgets together. These practices are complemented by the *Lean Budget Guardrails* which are policies for the governance and the spending of budgets.
Furthermore, the portfolio level defines the *Epics*, which represent a larger solution development initiative that requires analysis and financial approval before the implementation starts. The implementation typically takes several PIs. The epics are held in the *Portfolio Backlog* which can be visualized in the *Portfolio Kanban*.

The Portfolio level defines two new roles:

- The *Epic Owners* is responsible for the definition of the epics and facilitates their implementation through the teams in the ART. To define the features and capabilities of the epic, they work closely together with the stakeholders.
- The *Enterprise Architect* is responsible for architectural initiatives for the portfolio and facilitates the reuse of artifacts like ideas or components. He also promotes an adaptive design to ensure the overall architecture is adaptive to change.

Core Competencies

Distributed over the different levels, SAFe defines five core competencies of a lean enterprise [45] that are necessary for a successful implementation of SAFe:

- The first one is *Lean Portfolio Management* and is on the portfolio level (see Section 2.3.4) located. This is necessary to align the strategy and the execution.
- The *Business Solutions and Lean Systems Engineering* competency is part of the large solution level, which is not part of the portfolio configuration is SAFe. Nevertheless, this

competency is a set of practices and principles to enable a lean-agile development of large solutions.

- The *DevOps and Release on Demand* competency is part of the program level (see Section 2.3.4). It describes the use of DevOps (see Section 2.2.3) as well as a continuous delivery pipeline, which enables the delivery of the solution or parts of it whenever needed.
- *Team and Technical Agility* is located at the team level (see Section 2.3.4). Team agility describes self-organized high-performing agile teams which maximize value delivery. As the second part of this competency, technical agility defines a set of practices and principles to deliver reliable solutions quickly.
- *Lean-Agile Leadership* is located at the bottom of the SAFe visualization (see Figure 2.7) and goes through all levels. This competency describes the kind of leadership which all leaders should apply. They should empower all teams and individuals to reach their highest potential. Therefore, the leaders can make use of the principles and practices as well as values and the mindset of SAFe (see the following paragraph).

Principles, Lean-Agile Mindset, and Values

SAFe defines nine immutable principles. These should serve as guardrails for the entire organization to be able to work in an agile manner:

1. Take an economic view
2. Apply systems thinking
3. Assume variability; preserve options
4. Build incrementally with fast, integrated learning cycles
5. Base milestones on objective evaluation of working systems
6. Visualize and limit work in progress (WIP), reduce batch sizes, and manage queue lengths
7. Apply cadence, synchronize with cross-domain planning
8. Unlock the intrinsic motivation of knowledge workers
9. Decentralize decision-making

The goal of SAFe is to deliver value to the customer and is the purpose of the SAFe lean-agile mindset. It is founded on lean-agile leadership and contains the four pillars *Respect for People and Culture*, *Flow*, *Innovation*, and *Relentless Improvement*. The mindset of SAFe includes the following four core values:

- *Alignment* is necessary to deal with fast change, distributed teams, and disruptive forces of the environment. It ensures that everyone is working in the same direction without the need for top-down command and control.
- To ensure that quality standards are always met, *Built-in Quality* is the second core value of SAFe. The teams have to fulfill the quality standards in each increment and are not supposed to add the quality later.
- *Transparency* and openness enable trust. Furthermore, transparency is needed to enable fast and good decision-making.
- *Program Execution* is the last of the SAFe core values. It enforces the continuous execution of the program and hence the delivery of value to customers.

2.4 Teamwork Quality Model

Högl and Gemünden researched the connection between teamwork and the success of innovative projects. They conceptualize this relationship with the Teamwork Quality model (TWQ model) [36]. Therefore, they followed Homans [37], Denison et al. [16], and Hackman [33] to define the scope of the TWQ model and conducted a literature review including several case studies to define the construct teamwork quality. Högl and Gemünden conceptualize teamwork quality as a multifaceted higher order (latent) construct. The six facets of teamwork quality are defined as follows: [36]

- Communication:
  "Is there sufficiently frequent, informal, direct, and open communication?"
  This indicator is elementary for teamwork, as communication is necessary to exchange information among the team members. Especially, informal, direct, personal, open, and spontaneous communication is crucial for exchanging ideas and contributions among the team members. [61, 42, 30]
- Coordination:
  "Are individual efforts well structured and synchronized within the team?"
  For the quality of teamwork, the individual contributions of subtasks performed in parallel must be well synchronized and coordinated. Therefore, teams must agree on a common workflow. [82, 43, 9]
- Balance of member contributions:
  "Are all team members able to bring in their expertise to their full potential?"
  Especially in cross-functional teams, it is important that everyone can contribute their ideas and views. For the quality of teamwork, the contributions must be balanced in terms of the knowledge and experience of the individual team members. [33, 74]

- Mutual support:
  "Do team members help and support each other in carrying out their tasks?"
  Högl and Gemünden build on the work of Tjosvold [85], who states that mutual support is more productive than the competition in interdependent tasks. It is crucial for the quality of teamwork that team members work together toward their common goal and support each other when necessary.

- Effort:
  "Do team members exert all efforts to the team's tasks?"
  It is essential for the quality of teamwork that all team members demonstrate a uniformly high level of commitment and place the team's tasks above others. In addition, everyone must have a shared understanding of sufficient effort. [33, 61, 12]

- Cohesion:
  "Are team members motivated to maintain the team? Is there team spirit?"
  According to Cartwright [13], team cohesion refers to the degree to which team members desire to remain on the team. A sense of belonging and togetherness enables good collaboration, which builds the basis for good teamwork.

Högl and Gemünden define project success as a multivariable construct consisting of team performance and the personal success of team members. Once again, they build on the work of Gladstein [30], Hackman [33], and Denison [16], among others.
The project success is measured with the indicators effectiveness and efficiency. Since both depend on the evaluator's perspective, multiple views must be included to evaluate team performance. Högl and Gemünden [36] defined efficiency as the degree to which schedules and project budgets are met. Effectiveness is defined as the degree to which quality expectations are met.
Following Hackman [33], Sundstrom et al. [78], and Denison et al. [16], teams need to work together in ways that increase motivation and ensure future teamwork. Therefore, Högl and Gemünden define the latent variable personal success of team members via the two indicators work satisfaction and learning.

Figure 2.8 visualizes this conceptual model of the TWQ model. How Högl and Gemünden as well as other researchers applied this model to different cases is described in the next chapter.

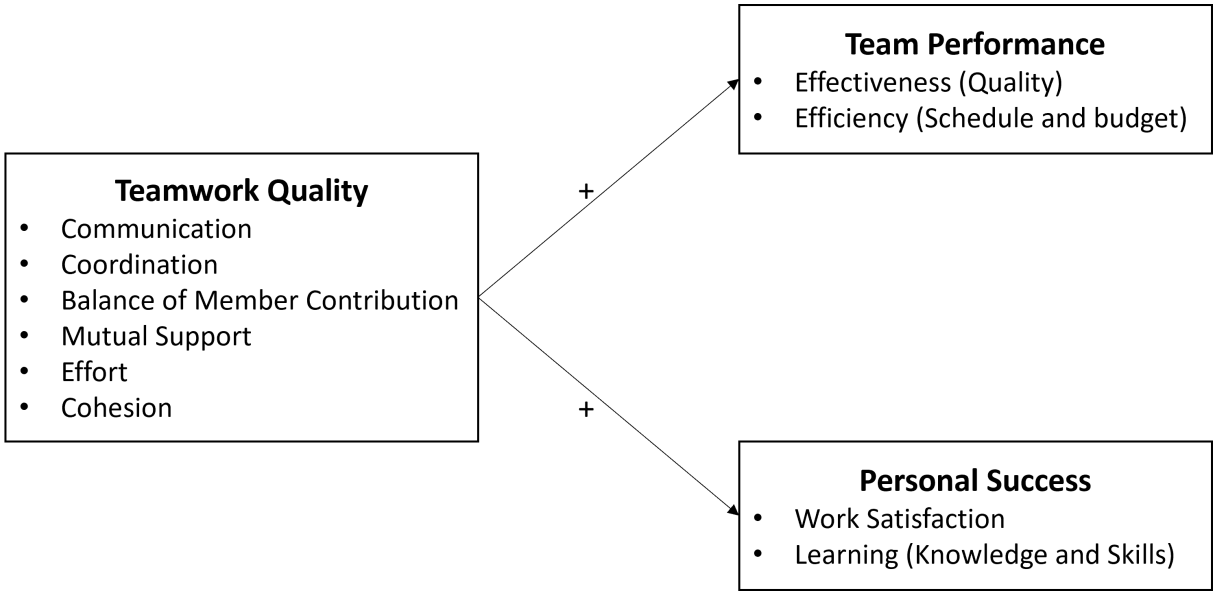


Figure 2.8: The conceptual TWQ model of Högl and Gemünden [36]

Chapter 3

Related Work

This chapter addresses what other researchers have already found related to the TWQ model and in the fields of team and program performance, as well as their influencing factors. Furthermore, the last section discusses the research on large-scale agile software development adoption in order to later relate this case study (See Chapter 4) to the literature and other cases. In each section, related literature is considered in chronological order.

3.1 Related Research on the Teamwork Quality Model

As mentioned in the previous chapter, Högl and Gemünden [36] introduced the TWQ model. To validate their conceptual model, they conducted a survey with 575 persons of 145 German software teams. TWQ and the team members' personal success are rated only by team members, while team leaders and managers also evaluate performance in addition to the team members. In three different models, they test the impact of TWQ on (1) team performance and personal success rated by team members, (2) team performance rated by team leaders, and (3) team performance rated by managers. Their results show a very strong relationship between TWQ and team members' personal success (standard coefficient 0.93, 87% variance explained) and between TWQ and team performance as rated by team members (standard coefficient 0.64, 41% variance explained). In contrast, the influence of TWQ on the team performance assessed by the team leaders is significantly lower (standard coefficient 0.34, 11% variance explained). The influence of TWQ on the team performance evaluated by the managers is even lower (standard coefficient 0.26, 7% variance explained). Regardless, their results show that the quality of collaboration is well represented by the six indicators (71% variance explained).

Lindsjørn et al. [49] adopted the TWQ model from traditional projects to agile projects. To do this, they used data from 71 agile software teams from 26 different companies. Instead of managers, they used responses from Product Owners as a second set of external evaluators of

team performance. They also developed a new questionnaire to meet the agile patterns of the indicators of the TWQ model.
By considering root mean squared error of approximation (RMSEA), the TWQ model has an almost close model fit for their data of the agile teams. Furthermore, some indicators were highly correlated, resulting in a non-positive definite matrix and negative error variance during estimation.
However, the similarity of the factor loadings of agile and traditional teams is very high. Hence, Lindsjørn et al. compared the results of their study with that of Högl and Gemünden [36] and came to the following conclusions: Firstly, the agreement between the different roles in rating team performance is lower in agile teams than in traditional ones. Secondly, the factor loadings between TWQ and the success measures are somewhat higher for the agile teams than for the traditional ones, except for the Product Owner performance ratings. There, the influence of TWQ on the team performance rated by the Product Owners is significantly lower than in the traditional teams (standard coefficient of 0.07 compared to 0.26).
The TWQ model with the factor loadings, path coefficients, and error variances of the agile teams is shown in Figure 3.1.

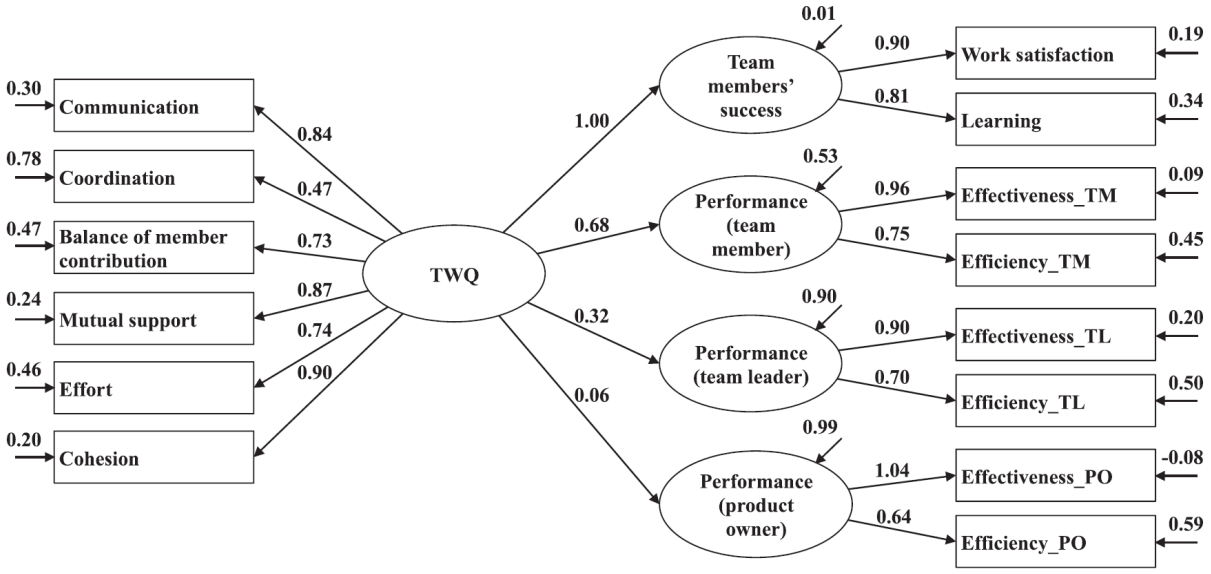


Figure 3.1: Standardized factor loadings, (structural) path coefficients, and error variances for the model under investigation in the study by Lindsjørn et al. [49]

Weimar et al. [93] extended the model of Högl and Gemünden by replacing the TWQ indicators coordination, balance of member contributions, and effort by coordination of expertise, value sharing, and trust. They tested their new model with 29 teams of 18 Dutch companies. In this study, TWQ explains 66% of the variance in team performance rated by team members and 40% of the variance in team performance rated by stakeholders.

In a further study, Lindsjørn et al. [48] investigated the differences between small and large projects by using the TWQ model. Therefore, they surveyed 31 teams in small agile projects and 33 teams in large agile projects, with a total of 320 team members and team leaders. Lindsjørn et al. defined projects consisting of one or two teams as small and projects consisting of more than ten teams as large.
Their study shows that the correlation between teamwork and team performance behaves the same in large and small projects as long as team members measure it. However, in large projects, the product quality (cf. effectiveness in Högl [36]) measured by the team leaders correlates negatively with the constructs of TWQ. In addition, in large projects, there is a disagreement in the evaluation of team performance between team members and team leaders.

Doepp [24] transferred the TWQ model from the team to the program level. As a baseline, he uses the questionnaire of Lindsjørn et al. [49]. The only change to the questionnaire is that the word team has been replaced by program in most of the questions in order to measure collaboration, success, and performance at the program level. In his study, Doepp investigated 14 teams of a German software company in the financial sector. His results show that the TWQ model is applicable to the program level. In his context, the quality of collaboration at the program level explains 80% of the performance measured by the team members, 28% of the performance measured by the Scrum Masters, 5% of the performance measured by the Product Owners, and 10% of the performance measured by the stakeholders.

Overall, all studies on the TWQ model show that it is applicable to different contexts. Thereby, the tendency of the factor loadings is always identical. The further away the respondents are from the teams, the less the quality of the collaboration explains the variance in the evaluation of team performance. Thus, TWQ explains the variance in performance measured by team members better than that measured by team leaders, Product Owners, or other outside groups of people. This tendency is evident in all studies, although all of them point out that this effect may in part be also due to rater bias. However, there is no straightforward comparability because not all studies used the same questionnaire.
With the adaptations of Weimar et al. [93], the TWQ model seems to describe the influence of teamwork on team performance better than the original one of Högl and Gemünden [36]. However, there is only this one study on it so far. The adaptation of the TWQ model from the team level to the program level by Doepp [24] provides promising results. However, the data from a single company is not sufficient to make reliable statements about the program level.

3.2 Related Research on Influence Factors on Team and Program Performance

Lencioni [46] describes in his book "The Five Dysfunctions of a Team" five pitfalls he has perceived when teams try to grow together. These are (1) absence of trust, (2) fear of conflict, (3) lack of commitment, (4) avoidance of accountability, and (5) inattention to results. Lencioni also concludes that if these five dysfunctions are solved and turned into the opposite, high-performing teams can emerge. Lencioni also describes that the five dysfunctions build on each other and should be resolved in their order.

Espinosa et al. [28] investigated the effect of team and task familiarity on team performance. For this, they investigated several geographically dispersed software teams. Their results show a positive influence of task familiarity on team performance, which decreases as tasks become more complex. However, task size has no influence on this effect. Furthermore, team familiarity also has a positive influence on team performance. This effect is even stronger in larger or distributed teams. In addition, Espinosa et al. found that team familiarity and task familiarity were substitutive. Thus, team performance benefits more from high task familiarity when team familiarity is low and vice versa.

In further research, Esponisa et al. [27] investigated the direct influence of temporal distance on team performance. For this purpose, they conducted a laboratory experiment. Their results show a direct association between temporal distance and team performance, which decreases significantly when the intervening team communication variables (communication frequency and turn-taking) are included in the analysis model. Furthermore, information transfer is found to be positively associated with production speed, while convergence increases product quality.

The internet company Google has also asked itself the question of what makes a team successful. For this purpose, they conducted more than 200 interviews and analyzed more than 250 attributes of about 180 teams [65]. Rozovsky [65] summarizes the results on re:work[1]. She concludes that the following five dynamics are particularly crucial for successful teams:

1. Psychological safety: Can we take risks on this team without feeling insecure or embarrassed?
2. Dependability: Can we count on each other to do high quality work on time?
3. Structure & clarity: Are goals, roles, and execution plans on our team clear?
4. Meaning of work: Are we working on something that is personally important for each of us?

[1] `https://rework.withgoogle.com/`

5. Impact of work: Do we fundamentally believe that the work we're doing matters?

[65]

The dynamics are ordered according to their importance for successful teams. Psychological safety is by far the most important characteristic of successful Google teams.

Other researchers have found that diversity of team members has a positive impact on performance [35], and that task complexity [35] and virtual teams [50] have a negative impact on team performance. In summary, there are a wide variety of factors that influence team performance, and science does not provide a clear picture.

3.3 Related Research on Large-Scale Agile Adoption

Dingsøyr et al. [21] have developed a taxonomy for classifying different scaling sizes. They also provide recommendations on how coordination can be managed in the appropriate settings. In total, their taxonomy consists of three types:

- Small-scale agile: This type consists of only one team. If this is the case, coordination can follow typical agile approaches such as daily meetings, joint planning, and retrospectives.
- Large-scale agile: This type consists of two to nine teams. For coordination, a new approach such as Scrum of Scrums is necessary.
- Very large-scale agile: This type consists of ten or more teams. In such very large-scale agile projects, multiple forums are necessary for overall coordination, like multiple Scrum of Scrums.

Bick et al. [7] conducted a multiple case study in five different large-scale agile software development programs at SAP SE[2]. They conducted 68 semi-structured interviews with Product Owners, Scrum Masters, and other key roles. Each of the five different programs used different approaches for scaling agile software development: [7]

- Scaling via central team directives:
  This approach was taken by one program with 13 teams at four different locations. The teams were aligned based on the business process that the software represented and were managed by a central team. The members' satisfaction with this approach was rather moderate because, among other things, dependencies could not be foreseen by the central team.
- Scaling via iterative proxy collaboration:
  This approach, which is quite similar to Scrum of Scrums (see Section 2.3.1), was chosen

[2] `https://www.sap.com/`

by a program with ten teams at two different locations. The satisfaction with this approach was quite high.

- Scaling via central team planning based on team inputs:
  A program used this approach with seven teams in six different locations. The modular software system allows a good balance of centralized and decentralized planning. The satisfaction with the approach was high.
- Scaling via full collaboration:
  The 85 members in the six teams of the project using this approach were co-located. This allowed program members to meet in person for release planning and other activities. The satisfaction with this approach was very high.
- Scaling via ad hoc communication:
  The program using this approach consisted of four teams in three different locations. Since this software was also modular, each module was managed by one team, so the need for coordination between teams was quite low. The Product Owners had weekly conferences with the Chief Product Owner. Overall, the satisfaction level was also quite high with this approach, as the culture was also based on very open communication.

Bick et al. [7] conclude that the approaches to inter-team coordination differ strongly in their types and methods. The management of dependencies also differs substantially. On the other hand, the different environmental factors do not seem to have an influence on the success of coordination.

Dikert et al. [20] conducted a systematic literature review to explore what challenges and success factors exist for large-scale agile transformations. For this purpose, they used a two-step process to filter out 52 from nearly 2000 papers, which they analyzed then in more detail. This analysis identified 35 challenges, which Dikert et al. sorted into nine different categories. These nine categories are: (1) Change resistance appeared in 38% of the papers, (2) lack of investment appeared in 31% of the papers, (3) agile is difficult to implement appeared in 48% of the papers, (4) coordination challenges in multi-team environment appeared in 31% of the papers, (5) different approaches emerge in a multi-team environment appeared in 21% of the papers, (6) hierarchical management and organizational boundaries appeared in 33% of the papers, (7) requirements engineering challenges appeared in 38% of the papers, (8) quality assurance challenges appeared in 14% of the papers, (9) integrating non-development functions appeared in 43% of the papers.
Furthermore, the literature analysis resulted in 29 success factors in eleven categories. These are (1) management support, which was mentioned in 38% of the cases, (2) commitment to change with a frequency of 17%, (3) leadership with a frequency of 17%, (4) choosing and customizing the agile approach with a frequency of 48%, (5) piloting with a frequency of 33%,

(6) training and coaching with a frequency of 36%, (7) engaging people with a frequency of 12%, (8) communication and transparency with a frequency of 17%, (9) mindset and alignment with a frequency of 40%, (10) team autonomy with a frequency of 24%, and (11) requirements management with a frequency of 24%.

Ebert and Paasivaara [25] conducted two case studies and compared their adoption of SAFe. They identified seven success factors: (1) Train personnel well in advance, (2) inform and engage people, (3) involve change agents, (4) hire an experienced external consultant, (5) prepare well for the first PI planning, (6) have a full-time RTE, and (7) take recognized improvement items seriously.

Dingsøyr et al. [23] conducted two case studies to investigate the use of group mode coordination in large-scale agile software development. The two programs studied, consisting of twelve and five teams respectively, are based on PRINCE2[3], and use Scrum at the team level. The results of the case studies show that group mode coordination in the form of scheduled and unscheduled meetings is used a lot. In addition, meetings seem to be changing. For example, unscheduled meetings have been formalized, and scheduled meetings have become unscheduled.

Paasivaara et al. [58] conducted a case study at Ericsson[4] to investigate their agile transformation. The motivation for the agile transformation is on the one hand that agility is part of the corporate strategy, but on the other that there is dissatisfaction with the current way of working and the need to be able to deliver software faster. The agile transformation was carried out in four phases: [58]

- Knowledge transfer and component-based teams
- Introducing agile
- Finding common ground through value workshops
- Towards continuous integration and deployment

During these phases, Ericsson faced several challenges, such as change resistance and technical debt.

Dingsøyr et al. [22] conducted a case study to find out how agile methods can be implemented at a very large-scale in terms of program organization, customer involvement, software architecture, and coordination between teams. Dingsøyr et al. found that the case study partner creates the solution description through teamwork in an iterative process. Hereby, alignment between the individual teams is one of the key success factors. The role of the architect in agile programs seems to be very challenging, as the interests of many stakeholders have to be weighed, and a lot of coordination is required. In addition, up-front and emergent architecture must be kept in

[3] https://www.prince2.com
[4] https://www.ericsson.com

balance. To enable exchange and coordination between the teams, numerous arenas have been created, such as Scrum of Scrums, lunch seminars, or open work areas.

In summary, there is no uniformly recommended approach or clear guidance for large-scale agile software development in the literature. Instead, it can be deduced that the procedure for scaling agile software development must be adapted to the respective environment. This is also evident in the challenges and success factors. The literature analysis by Dikert et al. [20] provides a good overview of these. Although the success factors do not promise successful agile scaling, companies that still have an agile scaling intention can benefit from the documented experiences of other companies.

Chapter 4

Case Study

This chapter describes the first part of the research - the investigation of the agile program at the case study partner. The chapter describes the case study design, the case study partner with the observed program, and the results.

4.1 Case Study design

This section gives relevant background information about the objective, setup, and methodology of the case study. Following Robson [63], the following questions must be answered in advance:

- Objective - what to achieve?
  The Goal of the case study is exploratory and descriptive. Firstly, it should provide insights into how a utility company performs a large-scale agile software development program. Secondly, the answer should help to place the survey results, and the application of the TWQ model to the survey data in the context of the investigated large-scale agile software development program. Lastly, this case study should give interested third parties enough information about the case study partner to compare the results with further studies.
  Therefore, the program setup and the adoption of the SAFe framework should be understood as well as the challenges and success factors of the program at the case study partner.

- The case - what is studied?
  The case is a large-scale agile software development program of a utility company. Section 4.2 describes the case study partner in more detail. The case study partner was observed between February and August 2021. The interviews took place in April 2021.

- Theory - frame of reference
  To understand the observations, findings, and the discussion of the results of the case study,

the basics about Scrum, large-scale agile software development, and SAFe are necessary. These are described in Chapter 2.

- Research questions - what to know?
  The main focus is to answer Research Question 1: "*How is a large-scale agile software development program performed at a utility company?*" In addition, the case study should provide relevant background information to place the survey results (see Chapter 5) in the context of the case.

- Methods - how to collect data?
  To generate insights, seven semi-structured interviews are conducted. The interview partners are two Agile Coaches [Itv1, Itv4], two Product Owners [Itv2, Itv5], two members of the Leadership Team [Itv3, Itv7], and the System Architect [Itv6]. Table 4.1 summarizes them. Each of the groups has different open-ended questions, some of which overlap between the groups. The respective questions are also included in the Appendix (see Section 8.2). All the interviews are time-boxed to 60 minutes, making it necessary to prioritize or skip some of the questions.

- Selection strategy - where to seek data?
  The case study partner is selected mainly based on their availability, which is typical in practice [5]. However, it fits the purpose of this work perfectly and complements the case study conducted by Doepp [24] (see Table 4.2).

Interview Partner's Role	Category	Interviewer	Date	Ref. Code
Agile Coach & RTE	Agile Coach (AC)	Manuel Styrsky	06.04.2021	Itv1
Product Owner	Product Owner (PO)	Manuel Styrsky	06.04.2021	Itv2
Program Manager & RTE	Leadership Team (LT)	Manuel Styrsky	07.04.2021	Itv3
Agile Coach	Agile Coach (AC)	Manuel Styrsky	07.04.2021	Itv4
Chief Product Owner	Product Owner (PO)	Manuel Styrsky	12.04.2021	Itv5
System Architect	System Architect (SA)	Manuel Styrsky	19.04.2021	Itv6
Head of IT department	Leadership Team (LT)	Manuel Styrsky	21.04.2021	Itv7

Table 4.1: Interviews partners of the case study

4.2 Case description

The case study partner is a German utility provider in the area of e-mobility with about 35.000 employees. The observed large-scale agile software development program develops and operates a platform for charging stations all across Europe and uses the portfolio configuration of SAFe 4.6 (see Section 2.3.4). The case study partner staffed its members of the observed program mainly by several external service providers. The program members are located in Germany, Slovakia, Vietnam, and Spain.

The characteristics of the case partner are mainly different compared to the case partner of Doepp [24], which makes the case partner interesting because it improves the external validity of the research, especially on the program level. Table 4.2 compares the characteristics of the two case study partners.

Characteristic	**This Case Study**	**Case Study of Doepp**
Industry sector	E-Mobility	Finance
Program size	7 teams 67 people	10 teams 150 people
Timing of observation	13th-16th PI: Agile working is already established	1st-3rd PI: At the begin of the agile transition
Staffing	Internal and external, via different external service providers	Internal
Geographical dispersion	International (Germany, Vietnam, Slovakia, Spain)	National (Several locations in Germany)
Working style	100% remote, due to the corona pandemic	Various: Co-located, partly co-located, and remote teams
Framework	SAFe 4.6 Portfolio Configuration	SAFe 4.6 Essential Configuration

Table 4.2: Characteristics of the case study partners

4.3 Adoption of Large-Scale agile Software Development

This section describes how the case study partner performs the large-scale agile software development and builds the basis for the answer of Research Question 1: "*How is a large-scale agile software development program performed at a utility company?*" The responses of the interview partners are clustered into questions regarding the historical background, the adoption of the SAFe framework, architectural aspects, as well as challenges, success factors, and lessons learned. Some of the interview partners were asked about their assessments regarding inter-team coordination as well as the TWQ-Indicators.

4.3.1 Historical Background and Agile Transformation

In 2017, a major migration project was on the agenda of the case study partner. The old monolith and the newly developed micro-service architecture had to be operated simultaneously. Furthermore, the different teams had many dependencies. This context made the scaling necessary. Therefore, the case study partner hired several external service providers, leading to strong personnel growth. [Itv1 (AC), Itv3 (LT), Itv5 (PO), Itv7 (LT)]
To deal with these challenges, one of the project leads recommended SAFe as scaling agile

framework. After some deliberation about the supported number of teams and the framework's scope, they decided to go with it. There was no day X where they said that SAFe would be used from now on. Instead, the introduction was more of a smooth transition and a bottom-up movement driven by the Agile Coaches. [Itv5 (PO), Itv1 (AC), Itv7 (LT)]
The single teams were already working in an agile manner. Gradually, they came into the program, and later, even more projects joined the SAFe framework. How the case study partner implemented SAFe in detail is described in the Section 4.3.2. In the beginning, there was no support from the top management. After a change of personnel, the management support increased significantly. Then, the program and portfolio level were added, and the training in agile methods got even more attention. Initially, Scrum Masters and Agile Coaches attended SAFe trainings and shared the knowledge with all teams. [Itv1 (AC), Itv3 (LT), Itv4 (AC), Itv7 (LT)]
Just as there was no precise start date, there is no precise date when one would say the implementation of SAFe is complete. The Agile Coaches still adopt and improve processes and working patterns. [Itv1 (AC), Itv4 (AC)]
During the whole time, the case study partner observed many changes in its environment. The change of the company owner, offshoring, spin-offs of parts of the company, changing business models, budget cuts as well as the resulting fluctuation, and the required the switch to remote work from one day to the next due to the corona pandemic [98] were only some of the external influence factors the program faced. The subjective degree of adoption was rated as "partially to completely" by all the interview partners. In addition, they rated the framework as supportive, necessary, inspiring, and quite the right thing for the moment. [Itv1 (AC), Itv2 (PO), Itv3 (LT), Itv4 (AC), Itv5 (PO), Itv7 (LT)]
Against expectations, the remote working did not lead to a performance decrease [Itv2 (PO), Itv4 (AC)]. After the go-live of the new platform, the new company owner cut the budget for the program. As a result, the program size was roughly halved from 150 to just under 70 people by now.

4.3.2 SAFe Adoption

In order to use the framework, changes had to be made to both the framework and the organization. The following paragraphs address these two sides of the adjustments.

Adaptations to the Framework

As stated before, the case study partner uses the portfolio configuration of SAFe 4.6. Figure 4.1 shows a visualization of the adjustments made by the case study partner. The information about the framework adoption comes from interviews with the two Agile Coaches, one of the Product Owners and a member of the Leadership Team [Itv1 (AC), Itv3 (LT), Itv4 (AC), Itv5 (PO)].

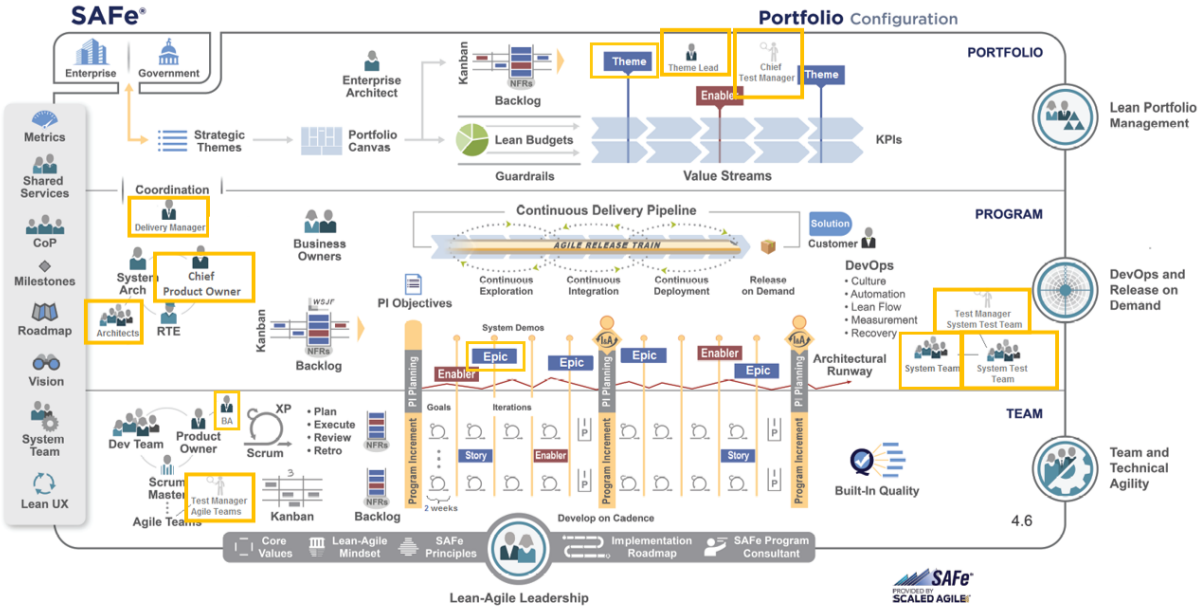


Figure 4.1: The Portfolio Configuration of the SAFe 4.6 Framework [45] with the adjustment made by the case organization in yellow rectangles
(Figure was provided by the case study partner)

The first thing that catches the eye is changed names for existing components. The reason for these changes is that the names defined by SAFe have already been used for other entities in the case organization. The change of names concerns the following elements:

- Theme (instead of Epic)
- Chief Product Owner (instead of Product Management)
- Epic (instead of Feature)

Over time, some more elements were added:

- The *Theme Lead* comes close to the Epic Owners in the original SAFe. This role accompanies a Theme from the beginning to the end and keeps the threads together. For smaller Themes where only one team is involved, this can also be the Product Owner. However, usually, this is a dedicated person.
- The *Chief Test Manager*, the *Test Manager - System Test Team*, and the *Test Manager - Agile Teams* were a group of roles that were responsible for the tests on the different organizational levels. Due to budget cuts, only one person is currently responsible for test management at all levels, which is not considered optimal by the interview partners.
- The *Delivery Manager* was responsible for holding the threads together during the migration project but has since faded into the background.

- The *Architects* support the System Architect and the agile teams, as the application is now too large and too complex for a single person to handle. Together with the teams, they establish security and architectural standards for the agile teams. More details about the architecting can be found in Section 4.3.3.
- The *System Test Team* is responsible for the end-to-end tests across the system borders.
- The System Team is already part of the original SAFe. It was added to the graphic to make its dependency on the System Test Team more explicit. Among others, it operates continuous testing and deployment pipelines.
- The *Business Analysts* supported the Product Owners at the early phases with the requirements analysis.

In addition to the changes visible in Figure 4.1, there are four other adjustments. (1) The System Demo occurs only once during a PI instead of after each iteration. (2) There are no dedicated Scrum Masters for every team. Instead, four Agile Coaches work together to support the whole train and the teams where they need support. The reason for this is due in part to the budget cuts as described in Section 4.3.1. (3) The Leadership Team is a virtual team containing all members on the program level, the agile coaches, and the Test Manager. (4) The PI Plannings are scheduled with almost one week significantly longer than recommended by SAFe.

Adaptations to the Organization

Besides the adjustments to the framework, several things changed in the organization. During the time of the framework implementation, much change happened independently from the changes that the SAFe implementation brought [Itv5 (PO)]. Therefore, some of the changes are not directly attributable to the introduction of SAFe.
The major change was the abolition of hierarchies, authority to issue directives, and disciplinary management. Instead, they focused on the SAFe roles and accountabilities. [Itv1 (AC), Itv3 (LT)]
Furthermore, a new mindset was necessary: Self-organization had to be introduced and enhanced, including a sense of responsibility within the teams. On the other side, the management had to learn to think iteratively and incrementally. Additionally, some new roles of the framework had to be established. Furthermore, the collaboration between the teams and the architects had to be realigned. [Itv1 (AC), Itv7 (LT)]
The Figure 4.2 was made with Gephi[1] and shows the perceived dependencies between the single teams. The strong interconnectedness between the teams suggests that there are no subgroups within the ART and that the program is well-tailored. The data comes from the survey, which is described in more detail in Chapter 5.

[1] `https://gephi.org/`

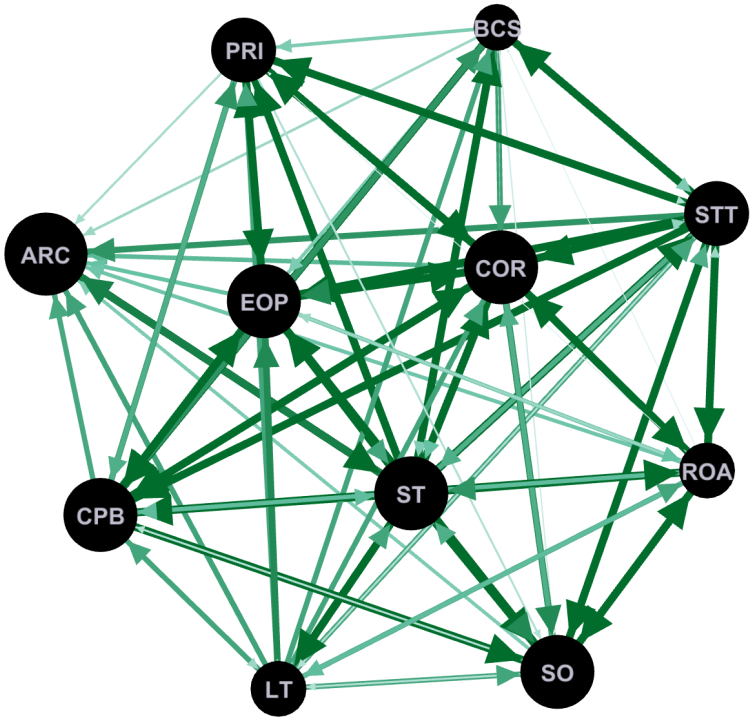


Figure 4.2: The team dependency graph of the observed ART (more details can be found in Section 5.3)
Legend:
- Node size: Degree of ingoing edges
- Edge thickness: Relative proportion of team members who identified the respective dependency

4.3.3 Agile Architecting

The agile manifesto only states: "The best architectures, requirements, and designs emerge from self-organizing teams" [54]. Since this is not a clear instruction on how architecture should be handled in agile projects, a more in-depth perspective should be taken on the architecture processes in the program under study.
The insights in this section come from the interview with the System Architect of the case study partner [Itv6 (SA)].
The case study partner pursues five goals with its architecture:

- Scalability
- Flexibility
- Reliability
- Security
- Cloud-Driven

To reach these goals, the case study partner applies a domain-driven design and uses loose

coupling, security by design, and a mix of synchronous and asynchronous communication patterns on the technical side. Whether these goals are achieved is checked by load tests, failure documentation, and code analysis. Responsible for the architecture is the System Architect together with the Architecture Team. These make the architecture decisions at the macro level, and the individual teams primarily make the architecture decisions at the micro-level themselves. The teams are encouraged to find a good balance of innovation and stability, and also to perform proof of concepts (PoCs). In principle, architectural decisions should be made so that the entire program supports them. In order to communicate and document architecture decisions, a decision log and architecture documents in the wiki are used in addition to regular exchanges.
The architectural decisions are mainly made in advance during the PI planning and in the so-called Use Case Process. During the iterations, only fine-tuning is done. For big projects, event storming workshops are held. Hence, architectural decisions are made intentionally as well as emergent. The most significant architectural decisions were (1) the breakup of the monolith to a microservice architecture, (2) the focus on the cloud, and (3) the use of kubernetes[2].
As described in Section 4.3.2 and 4.3.4, the whole program faces strong dependencies between the different teams. Therefore, contracts are used to define the application programming interfaces (APIs) between services developed by different teams. After implementation, integration tests verify that the services work together as agreed in the contracts. These contracts allow the teams to develop their services more independently from the other teams. The biggest challenge the architects face is engaging teams while not losing sight of the big picture.

4.3.4 Challenges

Especially the implementation of the SAFe framework brought some challenges:
Since agile working methods were new to some of the teams and individuals, they first had to be convinced of the sense, and added value of the framework [Itv3 (LT), Itv5 (PO), Itv7 (LT)]. Everyone takes different amounts of time to understand and adopt the agile mindset [Itv3 (LT), Itv7 (LT)] and for some, the scope of the framework was quite overwhelming [Itv1 (AC)]. In addition, cooperation between the teams first had to be established [Itv7 (LT)].
Furthermore, there was much effort necessary to adopt the organization to the framework, and vice versa [Itv1 (AC)]. The lac of top management support [Itv3 (LT)], the rapid growth [Itv5 (PO)], and the many dependencies [Itv1 (AC)] made this even more difficult.
Currently, the dependencies between the teams are still a challenge. As they are of technical and functional nature, the case study partner introduced CoPs to bring people together, "Dependency Sessions" to identify and evaluate dependencies, "Contracts" for the APIs to be able to develop more independently, and dependency boards to make the dependencies visible. [Itv1 (AC), Itv2 (PO), Itv4 (AC)]

[2] `https://kubernetes.io/`

Other challenges mentioned are capacity planning, program-level estimations, and forecasts, and the implementation of agile processes at portfolio level [Itv7 (LT)].
A big pain point is customer involvement. The requirements come over the roadmap process from the product management department, a legally separated company. In the beginning, it was not easy to integrate product management into the agile approach. More than 900 business customers make it even harder to involve them all directly, especially small customers. This situation leads to the fact that customers and users are not involved in the development process and have no contact with the developers. Therefore, the developers do not get direct feedback, apart from the ratings of the app users for the app development team. [Itv2 (PO), Itv3 (LT), Itv5 (PO), Itv7 (LT)]
The fact that the people come from several different external service providers did not influence the program negatively. Instead, they share their knowledge and behave like one big team [Itv1 (AC), Itv4 (AC), Itv7 (LT)]. The management of the case study partner puts effort to mix the teams as much as possible, which they see as a success factor (see the next section and Figure 4.3) [Itv7 (LT)]. Only the scheduling of meetings is challenging, as the people are working in different time zones and with different mother-tongues [Itv2 (PO)].

4.3.5 Lessons Learned and Success Factors

The interview partners were also asked what they learned from their experience in this specific program. The responses were mainly related to the introduction of SAFe.
In their opinion, they did not follow the recommendations of the SAFe implementation roadmap enough. As described in the previous sections, the program was set up bottom-up instead of engaging the top management first. The interview partners stated, that they would consider it more next time [Itv1 (AC), Itv4 (AC), Itv7 (LT)]. In addition, the internal interview partners realized that they should have involved top management more in the decision-making process in order to get more support from them [Itv3 (LT), Itv7 (LT)].
Even though they put a lot of time and effort into educating employees on the framework, they would spend more time in there next time [Itv1 (AC), Itv4 (AC), Itv7 (LT)]. The same applies to external consulting [Itv7 (LT)]. One of the interview partners said that he would highly recommend to engage external consulting services to any company looking to implement SAFe. Making adjustments to the framework was felt to be a good idea, as well as taking more time for PI plannings [Itv1 (AC), Itv4 (AC)].
Regarding the architecting, it was recognized that the individual teams had to be more involved and encouraged to participate [Itv6 (SA)].

The interview partners perceived several success factors regarding the program, its setup, and its performance. These can be clustered into four groups. The first group is cultural aspects like openness, motivation, and the will to improve and change. The second group is processes

like periodical retrospectives, shorter and stable PIs among others. The third group is the setup with the agile coaches, management support, and tool support. The last is the skills including the trainings and the technical excellence. Since mainly the best members were retained after the budget cuts (see Section 4.3.1), the technical and professional expertise of the employees is correspondingly high.
All the named success factors with their occurrences are visualized in Figure 4.3. All interview partners emphasized how much they value the culture within the program and are convinced that this is the most important success factor.

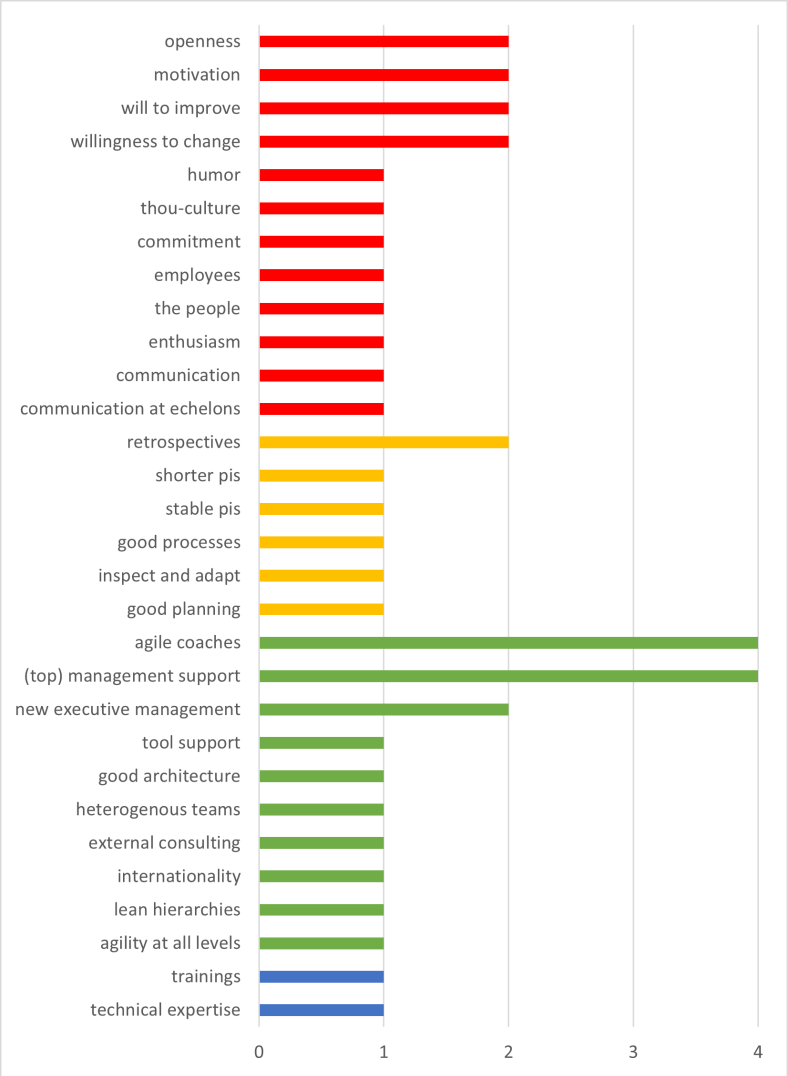


Figure 4.3: Success factors of the observed program as mentioned by the interview partners
x-axis: number of mentions
Coloring: red - cultural aspects, orange - processes, green - setup, blue - skills

4.3.6 Assessments Regarding the Indicators of the TWQ Model

Some of the interview partners were also asked about their assessments regarding the indicators of the TWQ model.

Although communication was difficult initially, especially between the individual teams, it is now considered quite good. It is also seen as one of the main reasons why the program is running so well nowadays. The interview partners are also delighted with the coordination and appreciate the transparency and the excellent tool support, especially in remote working. Even if there are differences between the individual employees, mutual support is considered to work very well. There is a very high willingness to help over the whole program. The effort is also perceived very favorably. Everyone knows what they are working for, is motivated, and gives their best. The same applies to cohesion which is perceived as very pronounced in every team, although fluctuation has reduced it slightly in some places. The balance of members' contributions is also positively assessed. Even if external staff are only engaged for specific technologies but are usually capable of much more, everyone contributes. [Itv1 (AC), Itv4 (AC)]

The situation is similar for the indicators of success. Work satisfaction is perceived to be very high, although it has fallen slightly in one team due to the change of a Product Owner. Learning is perceived even more positively. Right from the beginning, the team members learned a lot and learned quickly, no one was afraid of asking questions, and there was a lot of information exchange. [Itv2 (PO), Itv4 (AC)]

One of the Product Owners also rates the effectiveness and efficiency as good, although of course, this could be even better [Itv2 (PO)].

Chapter 5

Teamwork Quality Survey

This chapter describes how the survey was set up and conducted. It also describes how the data was processed and analyzed. Furthermore, the chapter describes the results.

5.1 Methodology

5.1.1 Questionnaire Design

The survey consists of two questionnaires, one for the team level and one for the program level. Both questionnaires consist of two parts. The first part asks for general information about the respondent, such as experience, education, and team. These questions are necessary to compare the results with other surveys and to exclude the influences of latent variables. The second part, with a total of 60 questions, examines the ten indicators of the TWQ model. The 60 questions of the second part are taken from Doepp [24] and are based on the questionnaire of Lindsjørn et al. [49]. Doepp adopted the questionnaire to the program level mainly by replacing the word "team" with the word "program" in the questions. All the 60 questions regarding the indicators of the TWQ model are 5 point Likert scaled [47] with the additional option "don't know". The survey was conducted online during the PI planning sessions, using the Questback[1] software. After the program members were briefly introduced to the topic without explicitly addressing the TWQ model, participants were invited to complete the surveys. Time was granted for both surveys on two different days during the PI Plannings. Since the participation rates were still not high enough, reminders were sent by e-mail.
For the program level data analysis and evaluation (see sections 5.3 and 6.1), the data from this survey are merged with those from Doepp's survey [24]. The resulting more extensive database allows for more robust conclusions regarding the applicability of TWQ model at the program

[1] `https://www.questback.com/`

level and the comparison of the TWQ model at both levels. However, this and the following section focus on this survey, as the information on Doepp's [24] survey is described in detail in his thesis. The questions used for the surveys of this study can be found in the Appendix in Section 8.1.

5.1.2 Setup and Approach

Following Lindsjørn et al. [49], the used method to analyze the data is SEM. The tool setup was also based on that of Lindsjørn et al. [49]. Data processing and analysis was done with RStudio[2] and SEM with the package lavaan[3] version 0.6-7.
The following steps were necessary to analyze the data: (1) The survey data was exported from Questback as CVS files and imported to RStudio. (2) To achieve data quality while not discarding too much data, data were checked for unengaged responses using the following four criteria:

- The duration it took the respondent to complete the survey:
  A threshold of 90 seconds was defined here. If respondents answered the questionnaire with about 70 questions in a shorter time, it could be assumed that the questions were not read carefully.
- The standard deviation over the whole questionnaire:
  Here, only answers are excluded if the standard deviation is exactly 0. In this case, it can be assumed that the respondent always made the same selection without reading the questions carefully.
- The reverse coded items were compared to the other ones:
  The threshold for this criterion was set to a delta of 3.0. This means that if the mean values of the normally coded questions and the reverse coded questions have a delta greater than this threshold, it is assumed that the reverse coded questions were mainly not recognized, and the questions were not read carefully.
- The number of questions answered with "don't know":
  Responses for which all questions for at least one indicator were answered with "don't know" were excluded, since in this case, parameters for the TWQ model would be missing. In addition, responses for which more than 20 questions (33.3%) of the entire questionnaire were answered with "don't know" were excluded, as the reliability is no longer guaranteed.

(3) In the third step, the indicator values are calculated by computing the arithmetic mean of the related questions. In this step, the reverse coded questions were inverted, and "don't know" responses were omitted. This distorts the result less than if these are replaced by a fixed number

[2] `https://www.rstudio.com/`
[3] `https://lavaan.ugent.be/`

such as the overall mean. (4) For the Scrum Masters, Product Owners, and stakeholders, the six indicators of teamwork quality were determined by the mean of the associated teams or programs since they are not direct team members. (5) Finally, the SEM analysis could be performed. (6) In the last step, the model fit was evaluated using the following criteria:

- P-value: The p-value essentially indicates the probability with which the null hypothesis holds in reality. In other words, that there is no correlation between TWQ and project success. This p-value should, at best, be less than 0.05 but at least less than 0.10.
- The standardized root mean squared residual (SRMR) is a very sensitive measure to detect misspecified factor covariances or latent structures and should be below 0.06 and below 0.11, respectively [39, 52].
- The root mean squared error of approximation (RMSEA) is a very sensitive measure to detect misspecified factor loadings and should be below 0.05. Furthermore, it takes the complexity of the model into account and compensates for it. [39, 52]

5.1.3 Structural Equation Modeling (SEM)

Hoyle [38] describes in his book the concepts of structural equation modeling (SEM). The information in this section is taken from his book unless otherwise noted.

In SEM, a distinction is made between two different types of variables: Latent variables are variables that cannot be measured directly but are of interest. Related to the TWQ model these are the latent exogenous variable TWQ, as well as latent endogenous variables success of team members and performance. The second type of variables are those that can be measured directly. These serve as estimates or indicators of the latent variables, allowing the latent variables to be determined. In case of the TWQ model, these are the six facets of teamwork quality as well as effectiveness, efficiency, work satisfaction, and learning.
The model also consists of two parts: The measurement model describes the form in which the measurable variables map the latent variables. In the TWQ model, for example, the six facets of teamwork quality are used to map the latent variable TWQ. The structural model describes the relationships between the latent variables. In case of the TWQ model, this is the influence of TWQ on team member success and performance.

5.2 Respondents

Both of the surveys were sent out to all 67 program members. On the team level, 48 (71.6%), and on the program level, 41 (61.2%) responded. Table 5.1 shows how they are distributed over the different teams. The Leadership Team (LT) encompasses all unique roles with responsibilities on the program level as described in Section 4.3.2. Table 5.2 shows the distribution over the different

Team	Respondents Team Level	Respondents Program Level	Team Size
ARC	2	1	3
BCS	4	2	7
COR	7	7	9
CPB	5	5	7
EOP	7	4	9
LT	8	8	9
PRI	3	1	5
ROA	5	5	7
SO	3	3	4
ST	1	2	4
STT	3	3	3
Total	48	41	67

Table 5.1: Respondents by team

Role	Type	Resp. Team Level	Resp. Program Level	Total
Agile Coach	Scrum Master	4	4	4
Architect	Dev. Team	2	1	3
Developer	Dev. Team	17	14	42
Tester	Dev. Team	14	10	
Member of LT	Stakeholder	5	4	5
PO / Teamlead	Product Owner	5	6	9
Member of ST	Dev. Team	1	2	4
Total		48	41	67

Table 5.2: Respondents by role

roles. Except for the Agile Coaches, the Members of the Leadership Team were not asked about their specific roles to ensure anonymity. The role types were introduced in order to be able to process the results and to achieve better comparability with other studies. Table 5.3 shows the distribution of the respondents according to education, experience, agile experience, and age. The questions about education and age were optional; hence, not all participants answered these questions.

In the survey of Doepp [24], 79 (53.4%) persons responded on the team level survey (57 of type "Dev. Team", 4 Product Owners, 8 Scrum Masters, and 10 Stakeholders), and 43 (29.1%) on the program level survey (21 of type "Dev. Team", 7 Product Owners, 6 Scrum Masters, and 9 Stakeholders). More details can be found in his thesis.

Characteristic	Category	Team level N	Team level %	Program level N	Program level %
Education*	A-Level	2	4.2%	1	2.4%
	Bachelor / Diplom	28	58.4%	24	58.6%
	Master	18	37.5%	12	29.3%
	PhD / Professor	0	0.0%	1	2.4%
	Other	0	0.0%	1	2.4%
	No answer	0	0.0%	2	4.9%
Experience	1-2 Years	0	0.0%	0	0.0%
	3-5 Years	8	16.7%	6	14.6%
	6-10 Years	22	45.8%	18	43.9%
	11-15 Years	14	29.2%	11	26.8%
	16-20 Years	3	6.3%	6	14.6%
	> 20 Years	1	2.1%	0	0.0%
Agile Experience	1-2 Years	3	6.3%	2	4.9%
	3-5 Years	29	60.0%	27	65.8%
	6-10 Years	15	31.3%	11	26.8%
	11-15 Years	1	2.1%	1	2.4%
	16-20 Years	0	0.0%	0	0.0%
	> 20 Years	0	0.0%	0	0.0%
Age*	20-25 Years	3	6.3%	2	4.9%
	26-30 Years	11	22.9%	6	14.6%
	31-35 Years	16	33.3%	12	29.3%
	36-40 Years	14	29.2%	14	34.1%
	41-45 Years	2	4.2%	4	9.7%
	46-50 Years	1	2.1%	0	0.0%
	51-60 Years	0	0.0%	0	0.0%
	> 60 Years	0	0.0%	0	0.0%
	no answer	1	2.1%	3	7.3%
Total		48	100%	41	100%

Table 5.3: Respondents by different characteristics
* these questions were optional

5.3 Data Analysis

In this section, the results of the data analysis are presented. For both the team and program level, the data from this case study and the merged data from both studies were analyzed. The separate analysis of the data of the study of Doepp [24], is also shown in his thesis, which is why it is not included here. However, since the TWQ model has already been validated several times at the team level, and the main goal is to validate whether the TWQ model is also applicable to the program of the utility company, the primary focus for the team level is on the data from

this program. In contrast, the focus for the program level is on the merged data since the data from one program is not sufficient to draw conclusions at the program level.

5.3.1 Team Level

From the total of 48 responses at the team level from the program of the utility company, only the response of one team member has to be removed because the standard deviation of the questions over the whole questionnaire is 0.0.
In the data from the study by Doepp [24] at the team level, the responses of 10 team members must be removed. Four of them apparently did not recognize reverse coded questions. Three others completed the questionnaire in less than 90 seconds, and the last three did not answer any of the corresponding questions for at least one indicator. In addition, the response of one Product Owner is excluded since the standard deviation of the individual questions over the entire questionnaire is 0.0.

Descriptive Statistics on Team Level

The results in Table 5.4 and Table 5.5 show the descriptive statistics at the team level with the data from the program of the utility company and the merged data, respectively. For mean, median, standard deviation, kurtosis, skewness, and variance influence factor (VIF), the mean values of the respondents' respective questions were used. For the alpha, the values of the single questions (items) were used.
The mean and median for all indicators are around 4.5 in the data from the program of the utility company, which is relatively high for a 5-point Likert scale. For the merged data, the values are around 4.0.
Kurtosis and skewness indicate the extent to which the distribution deviates from a normal distribution. Kurtosis indicates whether the actual distribution is flatter (negative values) or steeper (positive values) than the normal distribution. Skewness indicates whether the left (negative values) or right (positive values) tail is longer than in a normal distribution. Both kurtosis and skewness should be between -1 and 1 [34]. For the data from the program of the utility company, the kurtosis values of three indicators are outside the acceptable range, and for the merged data, only that of the effectiveness assessed by the Product Owners is outside the acceptable range. The remaining values are in a good or acceptable range. For skewness, all values are within an acceptable range. Only the values of the mutual support and work satisfaction indicators are slightly high. In general, the left tail of the individual indicators of both datasets is longer than in a normal distribution.

Indicator	Rater	Items	Mean	Median	SD	Kurtosis	Skewness	Alpha	Alpha[1]	VIF
Communication	TM(32)	10	4.28	4.30	0.29	-1.01	-0.02	0.34	0.69	2.03
Coordination	TM(32)	4	4.33	4.25	0.43	-0.40	-0.40	0.18	0.46	2.22
Mutual Support	TM(32)	7	4.72	4.86	0.33	0.04	-1.04	0.71		2.00
Effort	TM(32)	4	4.34	4.50	0.47	-0.17	-0.67	0.46	0.73	2.21
Cohesion	TM(32)	10	4.53	4.53	0.35	-1.09	-0.44	0.70	0.86	2.54
Balance of memb. contr.	TM(32)	3	4.38	4.42	0.49	-0.73	-0.51	0.38	0.57	2.17
Work satisfaction	TM(32)	4	4.62	4.75	0.44	0.77	-1.09	0.71		2.03
Learning	TM(32)	4	4.33	4.50	0.69	-0.95	-0.61	0.85		1.99
Effectiveness	TM(32)	10	4.39	4.37	0.44	-1.45	-0.06	0.85		1.81
Efficiency	TM(32)	5	4.34	4.25	0.50	-1.20	-0.04	0.60		1.81
Effectiveness	PO(6)	10	4.39	4.67	0.85	-0.80	-0.95	0.96		15.38
Efficiency	PO(6)	5	4.20	4.40	0.80	-1.21	-0.67	0.92		15.38
Effectiveness	SM(4)	10	4.48	4.55	0.39	-2.24	-0.18	0.92		1.10
Efficiency	SM(4)	5	4.45	4.40	0.53	-2.38	0.05	0.78		1.10
Effectiveness	SH(5)	10	4.26	4.29	0.52	-1.64	0.14	0.92		1.85
Efficiency	SH(5)	5	4.46	4.60	0.56	-1.15	-0.80	NA		1.85

Table 5.4: Descriptive statistics of the survey data from the program of the utility company at team level
Legend:
- Alpha[1]: calculated without taking the reverse coded questions into account
- TM = team member; SM = Scrum Master; PO = Product Owner; SH = stakeholder
- The numbers in brackets for the raters describe how many respondents rated the respective indicator.
- Kurtosis and skewness: green if $\in$ (-1, 1); yellow if $\in$ (-1.5, -1.0] or $\in$ [1.0, 1.5); red if $\leq$ -1.5 or $\geq$ 1.5
- Alpha: green if $\geq$ 0.7; yellow if $\in$ [0.5, 0.7); red if $<$ 0.5
- VIF: green if $<$ 4; yellow if $\in$ [4, 10); red if $\geq$ 10

Indicator	Rater	Items	Mean	Median	SD	Kurtosis	Skewness	Alpha	Alpha[1]	VIF
Communication	TM(79)	10	3.96	4.10	0.59	-0.87	-0.42	0.33	0.69	2.98
Coordination	TM(79)	4	3.94	4.00	0.76	-0.13	-0.65	0.19	0.47	3.91
Mutual Support	TM(79)	7	4.48	4.57	0.53	1.07	-1.21	0.71		3.81
Effort	TM(79)	4	3.91	4.00	0.78	-0.70	-0.46	0.45	0.71	3.68
Cohesion	TM(79)	10	4.03	4.10	0.73	-0.85	-0.48	0.71	0.87	5.38
Balance of memb. contr.	TM(79)	3	4.00	4.33	0.75	-1.03	-0.41	0.34	0.58	3.02
Work satisfaction	TM(79)	4	4.31	4.50	0.70	0.44	-1.03	0.87		3.97
Learning	TM(79)	4	4.16	4.25	0.75	-0.15	-0.74	0.85		2.96
Effectiveness	TM(79)	10	4.02	4.10	0.68	-0.29	-0.59	0.90		2.58
Efficiency	TM(79)	5	3.67	3.80	0.90	-0.56	-0.40	0.90		2.58
Effectiveness	PO(13)	10	3.79	3.50	0.86	-1.44	0.12	0.95		4.16
Efficiency	PO(13)	5	3.46	3.40	1.02	-1.57	0.01	0.92		4.16
Effectiveness	SM(8)	10	4.02	4.15	0.76	-0.75	-0.72	0.96		3.58
Efficiency	SM(8)	5	4.10	4.10	0.86	-1.35	-0.45	0.92		3.58
Effectiveness	SH(15)	10	4.17	4.29	0.49	-1.25	-0.21	0.88		1.76
Efficiency	SH(15)	5	4.21	4.40	0.69	-1.03	-0.59	0.88		1.76

Table 5.5: Descriptive statistics of the merged survey data on team level
Legend:
- Alpha[1]: calculated without taking the reverse coded questions into account
- TM = team member; SM = Scrum Master; PO = Product Owner; SH = stakeholder
- The numbers in brackets for the raters describe how many respondents rated the respective indicator.
- Kurtosis and skewness: green if $\in$ (-1, 1); yellow if $\in$ (-1.5, -1.0] or $\in$ [1.0, 1.5); red if $\leq$ -1.5 or $\geq$ 1.5
- Alpha: green if $\geq$ 0.7; yellow if $\in$ [0.5, 0.7); red if $<$ 0.5
- VIF: green if $<$ 4; yellow if $\in$ [4, 10); red if $\geq$ 10

Cronbach's alpha provides information about the internal consistency of a scale, i.e., the extent to which a set of questions (items) measure the same construct, and takes values between 0 and 1 [84]. Values above 0.7 are good and values below 0.5 are not acceptable [29]. The results in Table 5.4 and Table 5.5 show that for four of the six TWQ indicators, Cronbach's alpha is not in an acceptable range for both data sets. However, if one takes out the inverse-coded questions, only coordination is still in the non-acceptable range. For the indicators without reverse coded questions, no second alpha is reported. Since for the efficiency measured by the stakeholders at least one of the questions was not answered by any of the respondents from the program of the utility company, no alpha could be calculated for this indicator.
The VIF is often used as a measure of multicollinearity and is defined by the proportion of variance that the i-th independent variable shares with the other independent variables in the model. As a rule of thumb, acceptable limits are often specified below 4 or 10. However, higher values should not be directly considered a problem [56]. All the VIFs are in an acceptable range, except for the estimates of the Product Owners regarding effectiveness and efficiency in the data from the program of the utility company.

Structural Equation Modeling Results on Team Level

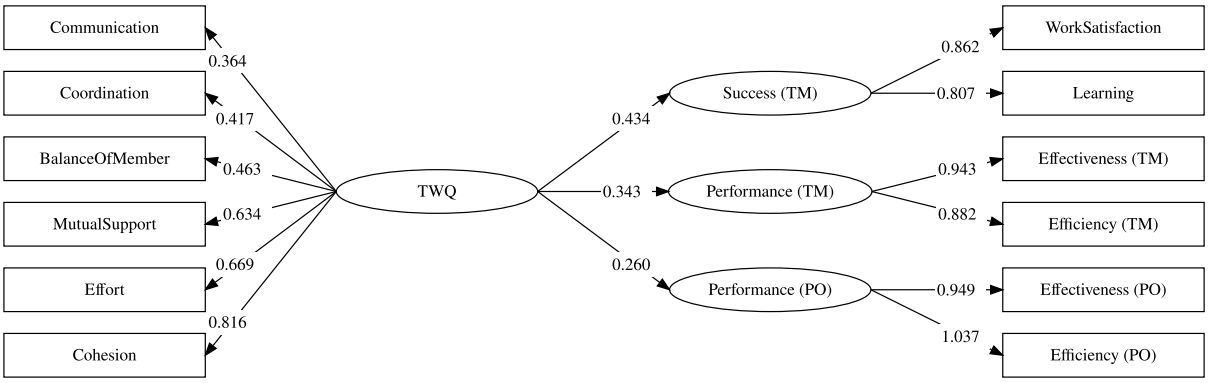


Figure 5.1: Visualization of the model with the survey data from the program of the utility company at team level (TM= team meber, PO = Product Owner)

For the TWQ model at the team level, only the responses of the team members and Product Owners were used. The reason for this is that in the program of the utility company, the Agile Coaches and stakeholders are not assigned to a direct team. Therefore no connection can be made between the TWQ of a single team and the team-specific performance assessments.
Table 5.6 and Figure 5.1 show the results of applying the TWQ model to the data of this case study. Here, the indicators of the latent variable TWQ can represent it with a 90% confidence interval. Except for coordination (p-value=0.069) and balance of member contribution (p-value=0.056), the reliability of all factor loadings is within a 95% confidence interval. It should be noted that the values of the factor loadings of the indicators of TWQ are in a broad range

	Rater	Factor Loading	p-value
Construct =∼ indikator (Measurement model)			
TWQ =∼ Communication		0.364	0.000
TWQ =∼ Coordination		0.417	0.069
TWQ =∼ Balance Of Member Contrib.	TM	0.463	0.056
TWQ =∼ Mutual Support	(32)	0.634	0.032
TWQ =∼ Effort		0.669	0.029
TWQ =∼ Cohesion		0.816	0.024
Success =∼ Work Satisfaction	TM	0.862	0.000
Success =∼ Learning	(32)	0.807	0.000
Performance =∼ Effectiveness	TM	0.943	0.000
Performance =∼ Efficiency	(32)	0.882	0.000
Performance =∼ Effectiveness	PO	0.949	0.000
Performance =∼ Efficiency	(6)	1.037	0.000
Latent endog. ∼ latent exog. (Structural model)			
Success (TM) ∼ TWQ		0.434	0.080
Performance (TM) ∼ TWQ		0.343	0.114
Performance (PO) ∼ TWQ		0.260	0.157

Metadata	**Value**
p-value	0.116
SRMR	0.092
RMSEA	0.073
Degrees of freedom	48
Total raters	47
Iterations in lavaan	101

Table 5.6: Result of the model with the survey data from the program of the utility company at team level
Legend:
- TM = team member; PO = Product Owner
- The numbers in brackets for the raters describe how many respondents rated the respective indicator.
- P-value, SRMR, and RMSEA are colored according to theirs thresholds described in Section 5.1.2.

between 0.35 and 0.82. The latent variables success and performance can be represented with factor loadings between 0.8 and some more than 1.0 by their indicators within a 99% confidence interval. Also, the data shows a factor loading of about 0.4 between TWQ and Success, which is within a 90% confidence interval. The factor loadings between TWQ and the performance assessments of the Product Owners and team members are slightly lower and outside the 90% confidence interval and thus not statistically significant.
This is also reflected in the p-value of the overall model (0.116). The fit measures SRMR and RMSEA are also quite high but still within their acceptable ranges, respectively.

Table 5.7 and Figure 5.2 show the results of applying the TWQ model to the merged data of both programs at the team level. Thereby, all latent variables can be represented with high

	Rater	Factor Loading	p-value
Construct =∼ indikator (Measurement model)			
TWQ =∼ Communication		0.828	0.000
TWQ =∼ Coordination		0.862	0.000
TWQ =∼ Balance Of Member Contrib.	TM	0.854	0.000
TWQ =∼ Mutual Support	(79)	0.815	0.000
TWQ =∼ Effort		0.920	0.000
TWQ =∼ Cohesion		0.915	0.000
Success =∼ Work Satisfaction	TM	0.952	0.000
Success =∼ Learning	(79)	0.888	0.000
Performance =∼ Effectiveness	TM	0.994	0.000
Performance =∼ Efficiency	(79)	0.891	0.000
Performance =∼ Effectiveness	PO	0.973	0.000
Performance =∼ Efficiency	(13)	0.998	0.000
Latent endog. ∼ latent exog. (Structural model)			
Success (TM) ∼ TWQ		0.560	0.000
Performance (TM) ∼ TWQ		0.474	0.000
Performance (PO) ∼ TWQ		0.124	0.197

Metadata	**Value**
p-value	0.006
SRMR	0.032
RMSEA	0.072
Degrees of freedom	48
Total Raters	115
Iterations in lavaan	136

Table 5.7: Result of the model on team level with the merged survey data
Legend:
- TM = team member; PO = Product Owner
- The numbers in brackets for the raters describe how many respondents rated the respective indicator.
- P-value, SRMR, and RMSEA are colored according to their thresholds described in Section 5.1.2.

factor loadings between 0.8 and 1.0 by their indicators within a 99% confidence interval. The data also shows factor loadings of around 0.5 between TWQ and the success and performance rated by the team members, which is also within a 99% confidence interval. The factor loading between TWQ and the performance assessments of the Product Owners of 0.12 is significantly lower and statistically not reliable (p-value=0.197). All the fit measures here are within their acceptable or even good range, respectively.

The data from both datasets, that from the program of the utility company and that of the merged data, was also applied to models, including the assessments of the Scrum Masters and stakeholders. The individual factor loadings of the different models are very similar to the initial model. However, the p-values of the whole models are significantly higher in the alternative

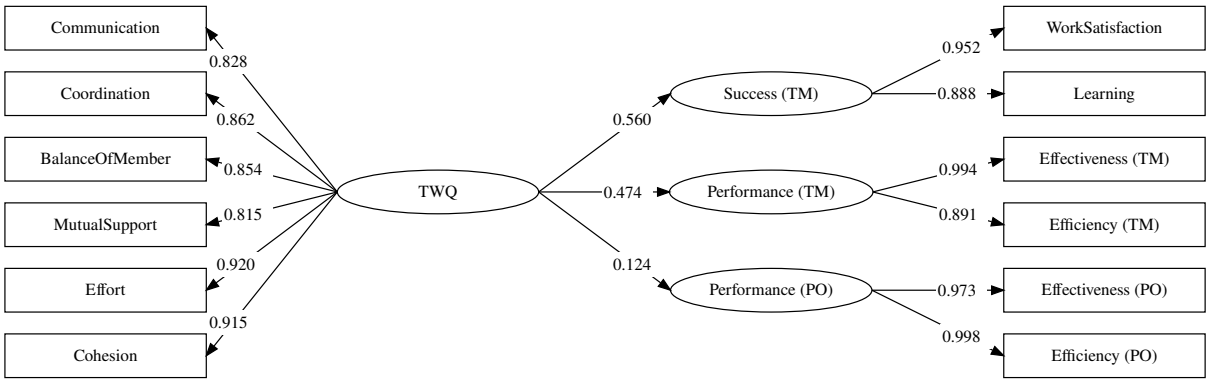


Figure 5.2: Visualization of the model on team level with the merged survey data (TM= team meber, PO = Product Owner)

models. Depending on the model specification, the p-value is between 0.30 and 0.98 for the models with the data from the program of the utility company and between 0.07 and 0.33 for the models with the merged data. SRMR and RMSEA take only marginally lower values compared to the initial model. In contrast, the RMSEA could not be calculated in the attempts to apply the different models to the data of this case study.

5.3.2 Program Level

At the program level, the TWQ model will only be applied to the merged data since the data from one program is not sufficient to draw conclusions at the program level.
Of the total of 41 responses from the survey at the program level from the program of the utility company at the program level, the responses of one Product Owner and two team members must be removed because they answered "don't know " to all questions for at least one indicator. The responses of two other team members must be removed because they answered more than 20 questions with "don't know". In addition, the answer of another Product Owner must be removed because the delta between the mean values of the reverse coded and the standard questions is greater than 3.0. This results in 35 responses remaining from the data at the program level from the program of the utility company.
Of the 43 answers from the data of the study by Doepp [24], the answers of one Scrum Master and one team member must be removed, since they have a standard deviation of 0.0 over the entire questionnaire, respectively have answered more than 20 questions with "don't know". In addition, the answers of four other team members have to be removed because they answered the questionnaire in less than 90 seconds, which indicates that they could not read the questions carefully. This last filter was only applied to the team members, as everyone else in Doepp's study only had to answer the last 15 questions of the questionnaire. This leaves a total of 37 responses from Doepp's study. Thus, 12 responses had to be removed, leaving a total of 72 responses for further processing and analysis.

Descriptive Statistics on Program Level

For the program level, the same statistics were used as for the team level. The descriptive statistics of the data from the program of the utility company is presented in Table 5.8 and the descriptive statistics of merged data is presented in Table 5.9.
The mean and median values at the program level are predominantly between 3.5 and 4.0 for the merged data and between 3.5 and 4.5 for those of this study.
The kurtosis values of the data from the program of the utility company are almost exclusively in the negative range, which indicates that the distributions are predominantly flatter than those of the normal distribution. The values for efficiency and effectiveness of all evaluators are outside the acceptable range, except the efficiency rated by the team members. For the merged data, except for one exception, all kurtosis values are within a good or acceptable range and indicate a flatter distribution than the normal distribution predominantly.
The values for skewness are in an acceptable range for both data sets. Only the skewness of the effectiveness rated by the team members from the program of the utility company is slightly high. In the data from the program of the utility company, the values of the indicators evaluated by the team members are predominantly slightly left-tailed, and the indicators evaluated by the other roles are slightly right-tailed. However, in the merged data, almost all indicators are left-tailed.
The alpha values of both data sets are predominantly in the acceptable range, with no value in the unacceptable range. The alphas calculated without the reverse coded questions are each slightly higher than those calculated with the reverse coded questions.
None of the VIF values of the merged data set are in the unacceptable range, but some are in the range between 4 and 10 and thus slightly high. In the data from the program of the utility company, however, the VIFs of the indicators effectiveness and efficiency measured by Scrum Masters and Product Owners are pretty high at just over 36 and 17, respectively.

Indicator	Rater	Items	Mean	Median	SD	Kurtosis	Skewness	Alpha	Alpha[1]	VIF
Communication	TM(23)	10	3.58	3.60	0.55	-0.69	0.06	0.75	0.79	2.07
Coordination	TM(23)	4	3.62	3.75	0.67	-0.08	-0.34	0.80	0.85	3.34
Mutual Support	TM(23)	7	4.09	4.00	0.57	-0.76	-0.09	0.87		3.67
Effort	TM(23)	4	3.85	3.75	0.60	-0.57	-0.41	0.75	0.82	2.08
Cohesion	TM(23)	10	4.06	4.20	0.46	0.23	-0.93	0.83	0.85	4.10
Balance of memb. contr.	TM(23)	3	3.91	4.00	0.62	-0.91	-0.28	0.59	0.83	1.77
Work satisfaction	TM(23)	4	4.04	4.00	0.68	0.27	-0.76	0.90		5.64
Learning	TM(23)	4	4.06	4.00	0.61	-0.48	-0.37	0.88		5.53
Effectiveness	TM(23)	10	3.74	4.00	0.73	1.59	-1.40	0.95		3.23
Efficiency	TM(23)	5	3.68	4.00	0.95	0.55	-0.71	0.96		3.23
Effectiveness	PO(4)	10	3.75	3.80	1.06	-1.87	-0.11	0.95		36.36
Efficiency	PO(4)	5	3.45	3.40	1.24	-1.91	0.09	0.96		36.36
Effectiveness	SM(4)	10	4.45	4.35	0.33	-1.99	0.41	0.77		17.82
Efficiency	SM(4)	5	4.65	4.60	0.30	-2.28	0.14	0.59		17.82
Effectiveness	SH(4)	10	4.40	4.35	0.36	-2.32	0.10	0.86		7.13
Efficiency	SH(4)	5	4.20	4.10	0.28	-1.88	0.53	0.67		7.13

Table 5.8: Descriptive statistics of the survey data from the program of the utility company at program level
Legend:
- Alpha[1]: calculated without taking the reverse coded questions into account
- TM = team member; SM = Scrum Master; PO = Product Owner; SH = stakeholder
- The numbers in brackets for the raters describe how many respondents rated the respective indicator.
- Kurtosis and skewness: green if $\in$ (-1, 1); yellow if $\in$ (-1.5, -1.0] or $\in$ [1.0, 1.5); red if $\leq$ -1.5 or $\geq$ 1.5
- Alpha: green if $\geq$ 0.7; yellow if $\in$ [0.5, 0.7); red if $<$ 0.5
- VIF: green if $<$ 4; yellow if $\in$ [4, 10); red if $\geq$ 10

Indicator	Rater	Items	Mean	Median	SD	Kurtosis	Skewness	Alpha	Alpha[1]	VIF
Communication	TM(39)	10	3.47	3.40	0.59	-0.47	-0.14	0.75	0.79	2.67
Coordination	TM(39)	4	3.52	3.67	0.74	-0.64	-0.41	0.79	0.85	2.77
Mutual Support	TM(39)	7	3.83	3.86	0.69	-0.99	-0.12	0.87		4.12
Effort	TM(39)	4	3.63	3.75	0.79	-1.02	-0.39	0.74	0.80	3.31
Cohesion	TM(39)	10	3.83	4.00	0.61	0.29	-0.97	0.83	0.85	6.05
Balance of memb. contr.	TM(39)	3	3.67	3.67	0.72	-0.75	-0.35	0.60	0.84	2.75
Work satisfaction	TM(39)	4	3.86	4.00	0.76	-0.48	-0.52	0.90		5.57
Learning	TM(39)	4	3.80	4.00	0.92	0.28	-0.93	0.92		6.08
Effectiveness	TM(39)	10	3.56	3.90	0.84	-0.34	-0.86	0.94		3.76
Efficiency	TM(39)	5	3.39	3.60	1.03	0.10	-0.68	0.94		3.76
Effectiveness	PO(11)	10	3.16	3.10	0.92	-0.45	0.17	0.95		8.13
Efficiency	PO(11)	5	3.04	3.20	0.95	0.07	0.04	0.93		8.13
Effectiveness	SM(9)	10	4.04	4.20	0.68	-1.37	-0.35	0.93		2.47
Efficiency	SM(9)	5	3.91	4.00	0.81	-1.66	-0.14	0.91		2.47
Effectiveness	SH(13)	10	3.81	4.10	0.85	-1.09	-0.65	0.94		7.63
Efficiency	SH(13)	5	3.51	4.00	1.11	-0.88	-0.60	0.94		7.63

Table 5.9: Descriptive statistics of the merged survey data on program level
Legend:
- Alpha[1]: calculated without taking the reverse coded questions into account
- TM = team member; SM = Scrum Master; PO = Product Owner; SH = stakeholder
- The numbers in brackets for the raters describe how many respondents rated the respective indicator.
- Kurtosis and skewness: green if $\in$ (-1, 1); yellow if $\in$ (-1.5, -1.0] or $\in$ [1.0, 1.5); red if $\leq$ -1.5 or $\geq$ 1.5
- Alpha: green if $\geq$ 0.7; yellow if $\in$ [0.5, 0.7); red if $<$ 0.5
- VIF: green if $<$ 4; yellow if $\in$ [4, 10); red if $\geq$ 10

Structural Equation Modeling Results on Program Level

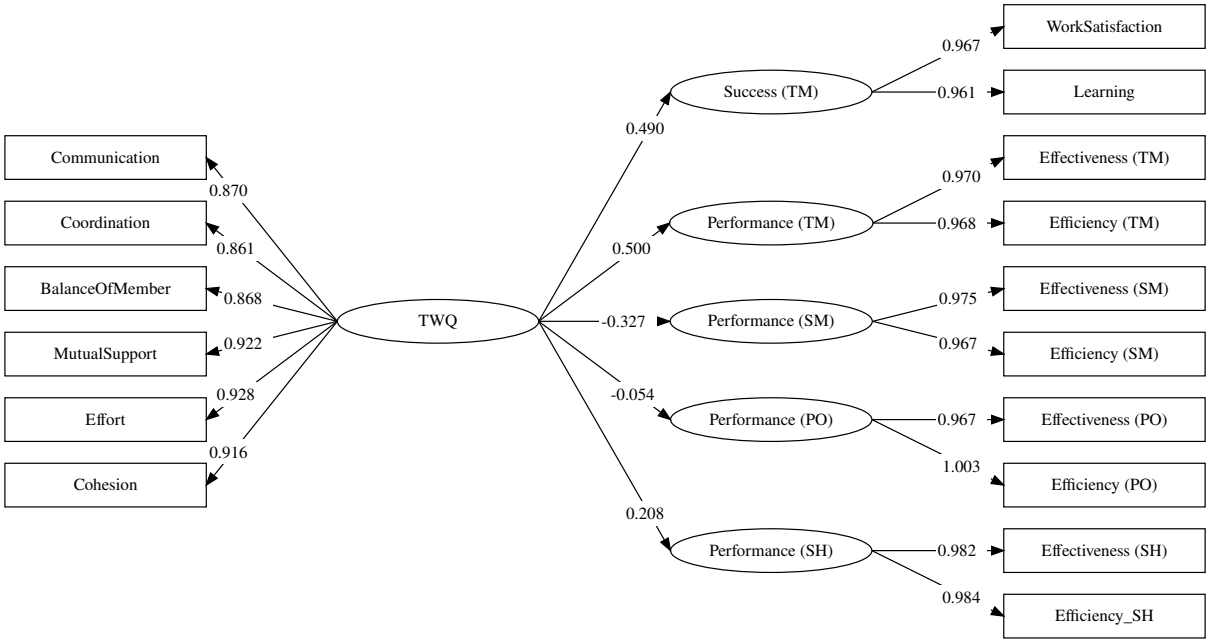


Figure 5.3: Visualization of the model on program level with the merged survey data (TM = team member, SM = Scrum Master, PO = Product Owner, SH = stakeholder)

At the program level, the TWQ model was applied only to the merged data. This is because at the program level the quality of teamwork at the program level was assessed, and one program alone is not sufficient to make reliable statements about the impact of teamwork at the program level on project success.

Table 5.10 and Figure 5.10 show the results of applying the TWQ model to the merged data of the program level surveys. Hereby, all latent variables can be represented by their indicators with factor loadings between 0.8 and 1.0 within a 99% confidence interval. The data shows factor loadings of around 0.5 between TWQ and the team members' rated success and performance, which are also within a 99% confidence interval. The factor loading of TWQ on the performance assessments of the Scrum masters is negative with about -0.3 and also lies in a 99% confidence interval. The data shows no significant influence of TWQ on the performance estimates of Product Owners. However, the data shows a slightly positive factor loading of TWQ on the performance assessments of the stakeholders, which lies within a 90% confidence interval.

The fit measures of the overall model are all in an acceptable or even good range. If the model is calculated without the assessments of Scrum Masters and stakeholders, the p-value is even lower, but the RMSEA is in an unacceptable range.

	Rater	Factor Loading	p-value
Construct =∼ indikator (Measurement model)			
TWQ =∼ Communication		0.870	0.000
TWQ =∼ Coordination		0.861	0.000
TWQ =∼ Balance Of Member Contrib.	TM	0.868	0.000
TWQ =∼ Mutual Support	(39)	0.922	0.000
TWQ =∼ Effort		0.928	0.000
TWQ =∼ Cohesion		0.916	0.000
Success =∼ Work Satisfaction	TM	0.967	0.000
Success =∼ Learning	(39)	0.961	0.000
Performance =∼ Effectiveness	TM	0.970	0.000
Performance =∼ Efficiency	(39)	0.968	0.000
Performance =∼ Effectiveness	SM	0.975	0.000
Performance =∼ Efficiency	(9)	0.967	0.000
Performance =∼ Effectiveness	PO	0.967	0.000
Performance =∼ Efficiency	(11)	1.003	0.000
Performance =∼ Effectiveness	SH	0.982	0.000
Performance =∼ Efficiency	(13)	0.984	0.000
Latent endog. ∼ latent exog. (Structural model)			
Success (TM) ∼ TWQ		0.490	0.000
Performance (TM) ∼ TWQ		0.500	0.000
Performance (SM) ∼ TWQ		-0.327	0.006
Performance (PO) ∼ TWQ		-0.054	0.656
Performance (SH) ∼ TWQ		0.208	0.084

Metadata	**Value**
p-value	0.052
SRMR	0.031
RMSEA	0.060
Degrees of freedom	89
Total Raters	72
Iterations in lavaan	176

Table 5.10: Result of the model on program level with the merged survey data
Legend:
- TM = team member; SM = Scrum Master; PO = Product Owner; SH = stakeholder
- The numbers in brackets for the raters describe how many respondents rated the respective indicator.
- P-value, SRMR, and RMSEA are colored according to their thresholds described in Section 5.1.2.

Chapter 6

Discussion

This chapter discusses the interviews and survey results and evaluates them in the context of the research questions and related literature. It also discusses the limitations of the research findings.

6.1 Findings

Similar to the research approach, the research findings are also divided into two parts: The first part is about the findings according to the large-scale agile software development program at the utility company, and the second part is about the adoption of the TWQ model to this program and the program level of large-scale agile software development programs. The following sections will discuss the findings of these two parts of the research.

6.1.1 Large-Scale Agile Software Development - Case Study

Research Question 1:
How is a large-scale agile software development program performed at a utility company?

The utility company uses SAFe for their large-scale agile development program. The trigger for setting up this agile program was a large migration project. The framework was introduced bottom-up and piece by piece, whereby the framework was adapted to the organization and vice versa. After 16 PIs by now, the whole program has settled in well and, despite many changes, the program works with a stable, high performance. Simultaneously, there are always new challenges, such as the more vital involvement of customers and users during observation.
Following the taxonomy of Dingsøyr et al. [21], the program with seven teams currently falls into the ”large-scale agile” type. In contrast, previously, the program would have fallen into the

"very large-scale agile" type, as it consisted of more than ten teams with a peak of 17 teams.
In reference to the five agile scaling approaches investigated by Bick et al. [7], the approach of the case study partner's program would be a mixture of "scaling via central team planning based on team inputs", as the central roadmap process governs the overarching planning, and "scaling via full collaboration". In contrast to the observations of Bick et al., direct communication in the case program is done remotely due to the corona pandemic.
Of the challenges that Dikert et al. [20] found in their literature review, "agile is difficult to implement", and initially, "hierarchical management and organizational boundaries" apply most to the case study partner's program. This also aligns quite well with the challenges identified by the respondents of the 14th Annual State of Agile Report [18].
Among the success factors identified by Dikert et al. [20], "management support", "commitment to change", "leadership", "choosing and customizing the agile approach", "training and coaching", and "mindset and alignment" apply to the program studied.
The findings of this case study and the success factors of this SAFe program also match the success factors found by Ebert and Paasivaara [25] in their case studies on the adoption of SAFe. It should be highlighted, however, that the interview partners strongly emphasized the cultural aspects, gave them great importance, and also see their culture as a decisive factor for their success.
Similar to the case study by Dingsøyr et al. [22], the architect's role is also a challenge in this program, while alignment between the teams is a success factor. This could be since architecture specifications are generally challenging to combine with self-organizing teams, but a strong alignment between the teams makes an excellent overarching architecture possible.
Contrary to findings from other research, this program did not find a negative impact of remote work on team performance. Among other things, this could be due to the fact that the program was already well established and had already worked remotely and internationally in some parts. Furthermore, the well-functioning culture, which is based on openness, intrinsic motivation, and the will to improve and change, could also be a reason why remote working has performed so well in this program.
In summary, the studied program has many similarities with other case studies on agile programs, although there are also a few differences as every implementation of SAFe is unique.

6.1.2 Adoption of the TWQ Model - Survey

The survey data and its analysis was used to answer the research questions according to the adoption of the TWQ model. This subsection addresses them in detail.

Research Question 2:
Can the TWQ model be applied to the team level of a large-scale agile software development program of a utility organization?

The answer to this question is based on the application of the TWQ model to the data from the survey at the team level from the program of the utility company. Section 5.3.1 describes the corresponding data analysis. Looking at the descriptive statistics in Table 5.4, it is already noticeable that many of the statistical metrics are outside their acceptable limits. Especially as some of the alpha values are very low, the results have to be considered critical because these low alphas challenge the internal consistency of the questionnaire. One reason for this could be that some respondents do not speak German or English as their mother tongue, but the questionnaire was only asked in these two languages. In addition, it could also be due to the fact that some of the reverse-coded questions are misleading. For example, question Q11, "There are mediators in the team communication through whom much communication is conducted" is reverse coded because mediators usually affect direct communication and are often used for conflict resolution. However, one could also think that mediators stand for the Agile Coaches or Scrum Masters. They facilitate the communication, leading to the question not being identified as reverse coded by every respondent. In addition, some of the indicators are only covered with three or four questions, which could lead to a strong influence on the alpha value if one of the questions was not understood correctly or was assessed slightly differently. However, since other studies, such as that of Lindsjørn et al. [49], already validated the questionnaire at the team level, it can be assumed that the questionnaire covers the individual constructs or indicators quite well.
The mean values of the single indicators are about 4.5, which is relatively high for a 5-point Likert scale. However, the results from the interviews confirm these positive assessments (see Section 4.3.6).
The factor loadings of the measurement model are all statistically significant, indicating that the individual indicators can well represent the latent variables. In particular, the latent endogenous variables can be represented quite well by their respective indicators. Thus, the data of this survey fits the measurement model of the TWQ model.
The situation is somewhat worse for the structural model. The factor loadings of the correlation between TWQ and the performance measured by team members (factor loading = 0.34) and Product Owners (factor loading = 0.26) are pretty low and slightly outside the 90% confidence interval (p-value = 0.11 resp. 0.16). SRMR (0.09) and RMSEA (0.07), while quite high, are still within the acceptable range, suggesting that the latent structures are not necessarily misspecified. The high values here could also be due to the high p-values of the latent structures or the small amount of data.
Compared to the studies of Högl [36], Lindsjørn et al. [49], and that of Doepp [24], the factor

loadings of the indicators of TWQ are also significantly lower in the application of the TWQ model to the team level data of the program of the utility company. The factor loadings of the correlations between TWQ and the performance and success assessments of the team members are also significantly lower here than in the other studies.
The indicators mutual support, effort, and cohesion have a particularly strong influence on the quality of teamwork. Nevertheless, the indicators communication, coordination, and balance of member contribution also significantly influence the quality of teamwork. However, these three indicators also have very low alpha values. The fact that the indicators were not covered well enough by their respective questions could also be an explanation for the weaker representation of TWQ by these three indicators than by the other ones.
Overall, the structural model can be represented quite well. However, the factor loadings differ significantly from other studies, and the correlations of the structural model are not statistically significant to a sufficient degree. Nevertheless, two of the three fit measures of the overall model are in their acceptable range. To answer the research question, it can be said that the TWQ model can be conditionally applied to the program of the utility company.

Research Question 3:
Can the TWQ model be applied to the program level of large-scale agile software development programs?

The answer to this research question is based on the application of the TWQ model to the merged survey data of both studies at the program level. As described in the previous chapter, reliable conclusions at the program level can only be made if more than one program is investigated. Section 5.3.2 describes the corresponding data analysis. From the descriptive statistics in Table 5.9, it can be seen that the statistical parameters are all in their acceptable range, except for a single exception with the kurtosis of efficiency as rated by the Scrum Masters. Therefore, it can be assumed that the questionnaire adapted to the program level covers the single indicators adequately. The somewhat lower alpha value for the indicator balance of member contribution has similarly appeared in the studies by Doepp [24], and Lindsjørn et al. [49]. This low alpha value could be because the questions do not represent the indicator well enough. Another reason could be that due to the small number of questions, the non-recognition of the reverse-coded question of an individual respondent has a strong influence on the alpha value. This is also reflected in the significantly higher alpha value when the reverse-coded question Q45 is removed.
Since all latent variables can be determined very well by their respective indicators with factor loadings between 0.8 and 1.0 within a confidence interval of 99%, it can be deduced that the measurement model of TWQ model can be applied very well to the program level. The results of the SEM analysis also show a positive influence of TWQ at the program level on the performance and success ratings of the team members with factor loadings of about 0.5 within a 99% confidence interval. The influence of TWQ at the program level on performance rated

by the Scrum Masters and Agile Coaches is also within a 99% confidence interval but is in a negative direction (-0.33). While there is no influence of TWQ at the program level on the performance measured by the Product Owners, there is a slightly positive influence (0.21) on the performance measured by the stakeholders within a 90% confidence interval.
The statistical fit measures of the overall model are all within the acceptable range. Therefore, it can be concluded that both the structural model and the overall model also fit the data quite well.
Lindsjørn et al. [48] discovered the negative correlation between the Scrum Master's performance evaluation and the TWQ constructs in a somewhat similar way in their study. In the study of Doepp [24], the same correlation occurred in a positive direction. These differences at the program level between this study and Doepp's study could also be due to the fact that Doepp's study only examined one program, which affects external validity. However, it is difficult to explain the negative influence of TWQ on the Scrum Masters' performance ratings. This negative influence could be because Scrum Masters, by the nature of their job, look for problems and potential for improvement.
The non-existent influence of TWQ on the performance assessments of the Product Owners could be since although they are pretty close to the teams, they always strive for a higher performance due to their job responsibility.
Caution should be taken when interpreting the positive influence of TWQ on the performance ratings of the stakeholders. Since only stakeholders of the whole program level were surveyed and only two programs were analyzed, several side effects may have led to this dependency. Firstly, during the survey, the program in this study was in its 13th PI and, therefore, was already settled in while the program in Doepp's [24] study was at the very beginning. In addition, the stakeholders of this study were very involved in the construction of the program and very satisfied with their outcome, while the outcome of the program of the study of Doepp [24] could not be visible yet.
The answer to this question is that the TWQ model can be applied well to agile programs. However, the correlations of the latent structures behave differently, and caution must be taken when interpreting the results. Nevertheless, all six indicators of TWQ are highly significant and have very high factor loadings. Hence, they are all significant indicators of the teamwork quality of agile programs.

Research Question 4:
What are commonalities and differences between the TWQ models at team and program level?

To answer the last research question, the merged data from both studies, this one, and Doepp's [24], were used to compare the TWQ model at team and program level.
When comparing the descriptive statistics in Table 5.5 and Table 5.9, it is noticeable that the

mean values at the program level are somewhat lower than at the team level. This could be due to the "better than average" effect [2, 1], which shows that people often rate themselves better than the average. In terms of program and team, the respondents would rate their team better than the average of the other teams. Another explanation for the difference in mean values between the team and program level could be that cohesion and cooperation are more strongly established within a team than between different teams. The statistical distributions for the individual indicators are both flatter than the normal distribution and slightly left-tailed at the team and program level, and thus very similar at both levels. The VIFs are also similar at both levels, although they are slightly higher at the program level.

Comparing the results of the SEM analysis, it is noticeable that the factor loadings between the latent variables and their indicators are very similar at both levels. This suggests that the indicators can be very well represented by their respective questions at both levels for the merged data. Therefore, the questionnaire seems to be very well adapted to the program level.

However, there are some differences in the relationship between TWQ and the performance and success ratings. The factor loading between TWQ and the success ratings at the program level is slightly lower than at the team level (0.49 vs. 0.56). This could be because respondents naturally have closer relationships with their team members than with the rest of the program, which leads them to learn more from each other and feel more comfortable within their team than within the program as a whole. Here again, however, the "better than average" effect [2, 1] could be another reason for this difference.

On the contrary, the factor loading between TWQ and the performance assessments of the team members at the program level is slightly higher than at the team level (0.50 vs. 0.47). However, the differences are only marginal, so this could also be a coincidence.

It is also interesting to note that the factor loading of TWQ on the performance ratings of Product Owners is somewhat lower at the program level than at the team level (-0.05 vs. 0.12). However, for both, the p-values are very high and outside the 90% confidence interval, which makes conclusions unreliable.

The statistical fit measures for the overall model are in an acceptable range for both models; they are also roughly identical, especially SRMR and the p-value when the model at the program level is calculated without the assessments of Scrum Masters and stakeholders.

With a few exceptions, the TWQ model behaves quite similarly overall at the team and program level as long as it is applied to data from the same context. However, the factor loadings of the structural model at both the team and program level differ significantly from those in other studies, as discussed in the previous research questions.

6.2 Limitations

The approach and context of the interviews and survey both have several limitations. Firstly, the program was studied for only five months, so changes over time can only be recorded to a minimal extent. Although the interviews also discussed the past, the data are not representative of the total duration of the program. In future studies, the program under study could be observed over a more extended period of time, and interviews could be repeated to investigate changes over time.
In addition, the entire study took place during the corona pandemic, which caused all observations and the interviews to be conducted remotely. This made it impossible, for example, to observe how the people interacted with each other on a daily basis. In studies done after the corona pandemic, this could be possible again.
Additionally only seven interviews were conducted, which is not a large number considering the size of the program. However, since there are no contradictions between the individual interviews, it is assumed that they can give a clear picture of the program of the utility company. Additionally, no stakeholders outside the program were interviewed, which resulted in a lack of comparison of perceptions of success and performance with external views. In future studies, the percentage of people interviewed could be increased, and more people who are not part of the program could be interviewed.

In the respect of the TWQ model surveys, there are also some limitations.
All program members worked remotely, which could influence their perception of teamwork. In future studies, remote and co-located programs could be investigated separately to ensure that remote working does not bias the results.
The survey results may also be affected because not all respondents speak German or English as their mother tongue. The possible resulting misunderstandings could have biased the results, as some of the survey participants may not have understood the questions correctly. In future studies, the questionnaire should be available to all participants in their mother tongue.
Also, the high alpha values in the survey at the team level in this study limit the robustness of the statements regarding Research Question 2. However, since the questionnaire has already been validated in other studies, it can be assumed that it covers the respective constructs well. However, it should be checked whether individual questions could be formulated more concretely.
Additionally, the survey was conducted during the PI planning. In the initial interviews, members of the case study partner have told that these days are often a very emotional phase for the team members, affecting the survey results. If possible, future studies should choose a timing when participants are not emotionally aroused. However, the chosen timing has made the very high participation rate in the surveys possible.
As described in Section 4.3.2, there are no Scrum Masters assigned to individual teams in the program of the utility company, which does not make the applicability of the TWQ model

impossible but does make it difficult to compare the results with other studies.
Even if for the conclusions regarding the TWQ model on program level the data of another study is added, two programs are still a very small number to make reliable statements. Moreover, both programs use SAFe, which can also produce internal side effects which are not investigated.

Chapter 7

Conclusion and Future Work

This chapter summarizes the results and reveals where further research is needed.

7.1 Conclusion

Research Question 1:
How is a large-scale agile software development program performed at a utility company?

Concerning the first research question, it can be seen that the agile program of the case study partner is strongly oriented towards SAFe and has followed a bottom-up approach against the recommendations of the SAFe implementation roadmap. Nevertheless, the case study partner is delighted with its implementation of the framework.
In contrast to the findings of other researchers and the expectations of the program members, the remote working required due to the corona pandemic surprisingly had no observable negative effects on the overall performance. Among other things, this could be due to the fact that the program was already well established and select parts had worked remotely as well as internationally.
The challenges and success factors identified in this program are mainly consistent with the ones identified in the literature. It should be emphasized that the interview partners attach great importance to culture and consider this to be a decisive factor for the success of the whole program.

Research Question 2:
Can the TWQ model be applied to the team level of a large-scale agile software development program of a utility organization?

When applying the TWQ model to the team level of the case study partner's program, there were several limitations. Some of the values of the descriptive statistics are out of their acceptable range. Especially, many of the alpha values are very low. These low values could be because not all questions were understood correctly, as not all participants could answer the questionnaire in their mother tongue.
In addition, the factor loadings of the latent structures of about 0.5 are pretty low compared to other studies. Furthermore, the p-values of these factor loadings are mainly outside the 90% confidence interval, which means that the relationship between TWQ and performance cannot be confirmed with certainty.
This is also reflected in the fit-measures of the overall model. Nevertheless, the individual latent variables can be reasonably well represented by the respective indicators, suggesting that the measurement model fits the data quite well.

Research Question 3:
Can the TWQ model be applied to the program level of large-scale agile software development programs?

The main goal of this thesis was to verify to what extent the TWQ model can be applied to the program level of large-scale agile software development programs. The data analysis of two different large-scale agile software development programs shows that the descriptive statistics are mainly within their acceptance limits. Furthermore, the individual latent variables can be determined adequately by their indicators, suggesting that the measurement model fits reasonably well at the program level.
However, the latent structures at the program level differ strongly from the findings of other studies at the team level. All the respective factor loadings are significantly lower than in other studies, and the influence between TWQ and the performance ratings of the Scrum Masters is actually negative. This negative correlation between TWQ and team leaders on large projects was also discovered by Lindsjørn et al. [48] in a somewhat similar way. Furthermore, TWQ seems to have no effect on the performance rating of Product Owners at the program level. Nevertheless, TWQ seems to be positively correlated with the performance ratings of stakeholders. Although this could also be due to other external influencing factors, as described in Section 6.1.2.
The fit-measures of the overall model are all within their acceptance limits. Hence, there is a strong indication that the TWQ model can be applied well to the program level of agile programs.

Research Question 4:
What are commonalities and differences between the TWQ models at team and program level?

Surprisingly, the average ratings at the team level are higher than at the program level when comparing the descriptive statistics at the team and program level. Possible explanations for this observation are described in Section 6.1.2. Furthermore, the latent structures at the program show slightly different correlations than at the team level. In more detail, the factor loading between TWQ and the success ratings is slightly lower. Furthermore, the factor loading between TWQ and the performance assessments of the team members is slightly higher, and the factor loading of TWQ on the performance rating of the Product Owners is somewhat higher at the team level than at the program level. However, overall, the TWQ model behaves very similarly in the SEM analysis at the team and program level.

Implications for Practice

For companies that practice large-scale agile software development, the results of this case study can help identify areas requiring attention in large-scale agile software development programs. Although, of course, every agile project is unique, the success factors and lessons learned (see Section 4.3.5) are particularly interesting, especially for companies that also rely extensively on external service providers. For these companies, the cultural aspects are probably even more important determiners of success.
The TWQ model explains more the elementary correlations. Hence, rather limited conclusions can be drawn for the application of the results in practice. However, it is recommended to pay attention to the fact that a high quality of teamwork is essential in agile projects. Even if the influence of TWQ on the performance ratings of the Scrum Masters appears to be negative, a positive influence on the success as well as the performance ratings of team members and stakeholders can be seen. The six indicators of TWQ could be an inspiration for a starting point of improving the quality of teamwork.

7.2 Future Work

The results of this thesis indicate that more research is needed in the area of large-scale agile software development as well as the context of the TWQ model. Although the thesis shows new research insights, it simultaneously reveals new questions for future research. There are also limitations in this study that need to be researched. How these limitations and the new questions can be addressed will be described below.
The limitations from Section 6.2 can be addressed in future studies through the following four considerations. Firstly, only two agile programs have been studied, both of which use SAFe to

scale agile practices. A broader range of agile programs, which in the best case use different frameworks than those analyzed so far, should be investigated to make more reliable statements and exclude any possible influence of the framework decision. Secondly, a long-term study should be conducted in which an agile software development program is observed and studied from the very beginning to the very end. Thirdly, regarding the survey, it would be beneficial if a mixed group of external stakeholders from various fields also participate in the survey in future studies. Finally, multiple interviews should then be conducted with several program members and external stakeholders to gain deeper insights into the program and better place the survey data in the context of the program.

Despite the limitations, the results of this thesis indicate four key areas where further research is needed. Firstly, new studies should investigate whether the TWQ model with the adaptations of Weimar et al. [93] explains the relationships between the quality of teamwork and success as well as performance better than the original model of Högl and Gemünden [36]. Secondly, the reasons for the negative effect of TWQ on the performance assessments of Scrum Masters and Product Owners at the program level should be investigated. Thirdly, the two programs examined showed different mean scores on single indicators, with those of the program examined in this study being slightly higher in each case than those of the program in Doepp's study [24]. This may suggest that other factors in the underlying context influence the performance of agile programs. Since Doepp's study was conducted at the very beginning of the agile program and this study after the program was already well established, one of these factors could be the timing of the survey. The TWQ model survey could be carried out several times to explore how TWQ and its influence on the success and performance of projects change over time. Finally, the reasons for the different mean values of the individual indicators at the team and program level should also be investigated.

This thesis advances the research in the area of large-scale agile software development and the application of the TWQ model to the program level of large-scale agile software development programs while also raising many important questions that still need to be answered. As large-scale agile software development becomes increasingly important, these insights can help many companies and researchers.

Chapter 8

Appendix

8.1 Survey Questionnaire

Table 8.1 shows the questions and their possible answers that were ask to determine the context of the respondents. Q07 was only part of the program level survey.
Table 8.2 shows the questions necessary for the TWQ model. These are all 5 point Likert scaled [47] with the additional option "don't know". They are took from Doepp [24] and based on the questionnaire of Lindsjørn et al. [49]. The notes in *italic* in the first two columns indicate if a question is reverse coded, and to which variable name it is mapped in the survey data.

Q01	**Which role description applies to you?** Welche Rollenbezeichnung trifft auf Sie zu? ◯ Developer ◯ Tester ◯ System Team Member ◯ Architect ◯ Agile Coach ◯ Product Owner / Team-Lead ◯ Member of Leadership Team ◯ Anderer / Other
Q02	**How many years have you been working in the field of software development?** Seit wie vielen Jahren sind Sie im Bereich der Softwareentwicklung tätig? ◯ 1 – 2 Jahre / Years ◯ 3 – 5 Jahre / Years ◯ 6 – 10 Jahre / Years

	◯ 11 – 15 Jahre / Years ◯ 16 – 20 Jahre / Years ◯ > 20 Jahre / Years
Q03	**How many years have you been working in the field of agile software development?** Seit wie vielen Jahren sind Sie im Bereich der agilen Softwareentwicklung tätig? ◯ 1 – 2 Jahre / Years ◯ 3 – 5 Jahre / Years ◯ 6 – 10 Jahre / Years ◯ 11 – 15 Jahre / Years ◯ 16 – 20 Jahre / Years ◯ > 20 Jahre / Years
Q04	**How old are you? (Optional)** Wie alt sind Sie? (Optional) ◯ 20-25 Jahre / Years ◯ 26-30 Jahre / Years ◯ 31-35 Jahre / Years ◯ 36-40 Jahre / Years ◯ 41-45 Jahre / Years ◯ 46-50 Jahre / Years ◯ 51-60 Jahre / Years ◯ > 60 Jahre / Years
Q07	**Education? (Optional)** ◯ Ausbildung? (Optional) ◯ Abitur / A-level ◯ Bachelor / Diplom / Fachausbildung ◯ Master / Magister ◯ Doktor / Professor / PhDs ◯ Anderer / Other
Q07	**Team member in? (currently)** Teammitgleid in? (zum aktuellen Zeitpunkt) ◯ ST ◯ STT ◯ COR ◯ PRI ◯ ROA ◯ SO

	◯ EOP ◯ CPB ◯ ARC ◯ LT ◯ Stakeholder / No Team
Q07*	**To which teams does your team have dependencies?** Zu welchen Teams hat Ihr Team Abhängigkeiten? ☐ ST ☐ STT ☐ COR ☐ PRI ☐ ROA ☐ SO ☐ EOP ☐ CPB ☐ ARC ☐ LT

Table 8.1: Questions to determine the context of the respondent (* only part of the program level survey)

Communication

Q08	Team *v_31*	**There is frequent communication within the team** Es findet eine regelmäßige Kommunikation innerhalb des Teams statt
	Program *v_32*	**There is frequent communication within the program** Es findet eine regelmäßige Kommunikation innerhalb des Programms
Q09	Team *v_33*	**The team members communicate often in spontaneous meetings, phone conversations, etc.** Die Teammitglieder kommunizieren oft in spontanen Meetings, Telefonaten, etc.
	Program *v_34*	**The program members communicate often in spontaneous meetings, phone conversations, etc.** Die Programmmitglieder kommunizieren oft in spontanen Meetings, Telefonaten, etc.
Q10	Team *v_35*	**The team members communicate mostly directly and personally with each other** Die Teammitglieder kommunizieren meist direkt und persönlich miteinander

	Program *v_36*	**The program members communicate mostly directly and personally with each other** Die Programmmitglieder kommunizieren meist direkt und persönlich miteinander
Q11 *reverse coded*	Team *v_37*	**There are mediators in the team communication through whom much communication is conducted** Es gibt Mediatoren in der Team Kommunikation, durch diese die Kommunikation begleitet wird
	Program *v_38*	**There are mediators in the program communication through whom much communication is conducted** Es gibt Mediatoren in der Programm Kommunikation, durch diese die Kommunikation begleitet wird
Q12	Team *v_39*	**Relevant ideas and information relating to the teamwork is shared openly by all team members** Relevante Ideen und Informationen zur Teamarbeit werden von allen Teammitgliedern offen ausgetauscht
	Program *v_40*	**Relevant ideas and information relating to the teamwork in the programm is shared openly by all program members** Relevante Ideen und Informationen zur Teamarbeit im Programm werden von allen Programmmitgliedern offen ausgetauscht
Q13 *reverse coded*	Team *v_41*	**Important information is kept away from other team members in certain situations** Wichtige Informationen werden in bestimmten Situationen von anderen Teammitgliedern ferngehalten
	Program *v_42*	**Important information is kept away from other program members in certain situations** Wichtige Informationen werden in bestimmten Situationen von anderen Programmmitgliedern ferngehalten
Q14 *reverse coded*	Team *v_43*	**In the team there are conflicts regarding the openness of the information flow** Im Team gibt es Konflikte um die Offenheit des Informationsaustausches
	Program *v_44*	**In the program there are conflicts regarding the openness of the information flow** Im Programm gibt es Konflikte um die Offenheit des Informationsaustausches

Q15	Team *v_45*	**The team members are happy with the timeliness in which they receive information from other team members** Die Teammitglieder sind zufrieden mit der Aktualität, mit der sie Informationen von anderen Teammitgliedern erhalten
	Program *v_46*	**The program members are happy with the timeliness in which they receive information from other program members** Die Programmmitglieder sind zufrieden mit der Aktualität, mit der sie Informationen von anderen Programmmitgliedern erhalten
Q16	Team *v_47*	**The team members are happy with the precision of the information they receive from other team members** Die Teammitglieder sind zufrieden mit der Qualität der Informationen, die sie von anderen Teammitgliedern erhalten
	Program *v_48*	**The program members are happy with the precision of the information they receive from other program members** Die Programmmitglieder sind zufrieden mit der Qualität der Informationen, die sie von anderen Programmmitgliedern erhalten
Q17	Team *v_49*	**The team members are happy with the usefulness of the information they receive from other team members** Die Teammitglieder sind zufrieden mit dem Mehrwert der Informationen, die sie von anderen Teammitgliedern erhalten
	Program *v_50*	**The program members are happy with the usefulness of the information they receive from other program members** Die Programmmitglieder sind zufrieden mit dem Mehrwert der Informationen, die sie von anderen Programmmitgliedern erhalten

Coordination

Q18	Team *v_105*	**The work done on subtasks within the team is closely harmonized** Die Arbeit an Teilaufgaben im Team ist eng aufeinander abgestimmt
	Program *v_106*	**The work done on subtasks within the program is closely harmonized** Die Arbeit an Teilaufgaben im Programm ist eng aufeinander abgestimmt
Q19	Team *v_107*	**There are clear and fully comprehended goals for subtasks within our team** Es gibt klare und vollständig verstandene Ziele für Teilaufgaben in unserem Team

	Program *v_108*	**There are clear and fully comprehended goals for subtasks within our program** Es gibt klare und vollständig verstandene Ziele für Teilaufgaben in unserem Programm
Q20	Team *v_109*	**The goals for subtasks are accepted by all team members** Die Ziele für die Teilaufgaben werden von allen Teammitgliedern akzeptiert
	Program *v_110*	**The goals for subtasks are accepted by all program members** Die Ziele für die Teilaufgaben werden von allen Programmmitgliedern akzeptiert
Q21 *reverse coded*	Team *v_111*	**There are conflicting interests in our team regarding subtasks/subgoals** In unserem Team gibt es Interessenkonflikte bei Teilaufgaben/Unterzielen
	Program *v_112*	**There are conflicting interests in our program regarding subtasks/subgoals** In unserem Programm gibt es Interessenkonflikte bei Teilaufgaben/Unterzielen

Mutual Support

Q22	Team *v_71*	**The team members help and support each other as best they can** Die Teammitglieder helfen und unterstützen sich gegenseitig so gut sie können
	Program *v_72*	**The program members help and support each other as best they can** Die Programmmitglieder helfen und unterstützen sich gegenseitig so gut sie können
Q23	Team *v_73*	**If team conflicts come up, they are easily and quickly resolved** Treten Konflikte im Team auf, lassen sie sich leicht und schnell lösen
	Program *v_74*	**If program conflicts come up, they are easily and quickly resolved** Treten Konflikte im Programm auf, lassen sie sich leicht und schnell lösen
Q24	Team *v_75*	**Discussions and controversies in the team are conducted constructively** Diskussionen und Kontroversen im Team werden konstruktiv geführt
	Program *v_76*	**Discussions and controversies in the program are conducted constructively** Diskussionen und Kontroversen im Programm werden konstruktiv geführt

Q25	Team *v_77*	**Suggestions and contributions of team members are respected** Vorschläge und Beiträge von Teammitgliedern werden berücksichtigt
	Program *v_78*	**Suggestions and contributions of program members are respected** Vorschläge und Beiträge von Programmmitgliedern werden berücksichtigt
Q26	Team *v_79*	**Suggestions and contributions of team members are discussed and further developed** Vorschläge und Beiträge von Teammitgliedern werden diskutiert und weiterentwickelt
	Program *v_80*	**Suggestions and contributions of program members are discussed and further developed** Vorschläge und Beiträge von Programmmitgliedern werden diskutiert und weiterentwickelt
Q27	Team *v_81*	**The team is able to reach consensus regarding important issues** Das Team ist in der Lage, einen Konsens über wichtige Themen zu erzielen
	Program *v_82*	**The program is able to reach consensus regarding important issues** Die Programmmitglieder sind in der Lage, einen Konsens über wichtige Themen zu erzielen
Q28	Team *v_83*	**The team cooperate well** Das Team arbeitet gut zusammen
	Program *v_84*	**The program cooperates well** Die Programmmitglieder arbeiteten gut zusammen Effort

Effort

Q29	Team *v_199*	**Every team member fully pushes the teamwork** Jedes Teammitglied treibt die Teamarbeit voran
	Program *v_200*	**Every program member fully pushes the teamwork in the program** Jedes Programmmitglied treibt die Programmarbeit voran
Q30	Team *v_201*	**Every team member makes the teamwork their highest priority** Jedes Teammitglied macht die Teamarbeit zu seiner obersten Priorität
	Program *v_202*	**Every program member makes the teamwork in the program their highest priority** Jedes Programmmitglied macht die Teamarbeit im Programm zu seiner obersten Priorität
Q31	Team *v_203*	**The team put(s) much effort into the teamwork** Das Team hat viel Mühe in die Teamarbeit gesteckt

	Program *v_204*	**The program put(s) much effort into the teamwork in the program** Die Programmmitglieder haben viel Mühe in die Teamarbeit im Programm gesteckt
Q32 *reverse coded*	Team *v_205*	**There are conflicts regarding the effort that team members put into the teamwork** Es gibt Konflikte bezüglich der Leistung, welche die Teammitglieder in die Teamarbeit einbringen
	Program *v_206*	**There are conflicts regarding the effort that program members put into the teamwork in the program** Es gibt Konflikte bezüglich der Leistung, welche die Programmmitglieder in die Teamarbeit im Programm einbringen

Cohesion

Q33	Team *v_153*	**The teamwork is important to the team** Die Teamarbeit ist wichtig für das Team
	Program *v_154*	**The teamwork in the program is important to the program members** Die Teamarbeit im Programm ist wichtig für das Programm
Q34	Team *v_155*	**It is important to team members to be part of the team** Es ist wichtig, für die Teammitglieder, Teil des Teams zu sein
	Program *v_156*	**It is important to program members to be part of the program** Es ist wichtig, für die Programmmitglieder, Teil des Programms zu sein
Q35 *reverse coded*	Team *v_157*	**The team does not see anything special in this teamwork** Das Team sieht in der Teamarbeit nichts Besonderes
	Program *v_158*	**The program members do not see anything special in this teamwork in the program** Die Teammitglieder sehen in der Teamarbeit im Programm nichts Besonderes
Q36	Team *v_159*	**The team members are strongly attached to the team** Die Teammitglieder sind stark mit dem Team verbunden
	Program *v_160*	**The program members are strongly attached to the program** Die Programmmitglieder sind stark mit dem Programm verbunden
Q37	Team *v_161*	**All team members are fully integrated in the team** Alle Teammitglieder sind vollständig in das Team integriert

	Program *v_162*	**All program members are fully integrated in the program** Alle Programmmitglieder sind vollständig in das Programm integriert
Q38 *reverse coded*	Team *v_163*	**There were many personal conflicts in the team** Es gab viele persönliche Konflikte im Team
	Program *v_164*	**There were many personal conflicts in the program** Es gab viele persönliche Konflikte im Programm
Q39	Team *v_165*	**There is mutual sympathy between the members of the team** Es besteht gegenseitige Sympathie zwischen den Mitgliedern des Teams
	Program *v_166*	**There is mutual sympathy between the members of the program** Es besteht gegenseitige Sympathie zwischen den Mitgliedern des Programms
Q40	Team *v_167*	**The team sticks together** Das Team hält zusammen
	Program *v_168*	**The program members stick together** Das Programm hält zusammen
Q41	Team *v_169*	**The members of the team feel proud to be part of the team** Die Mitglieder des Teams sind stolz darauf, Teil des Teams zu sein
	Program *v_170*	**The members of the program feel proud to be part of the program** Die Mitglieder des Programms sind stolz darauf, Teil des Programms zu sein
Q42	Team *v_171*	**Every team member feels responsible for maintaining and protecting the team** Jedes Teammitglied fühlt sich verantwortlich für die Aufrechterhaltung und den Fortbestand des Teams
	Program *v_172*	**Every program member feels responsible for maintaining and protecting the program** Jedes Programmmitglied fühlt sich verantwortlich für die Aufrechterhaltung und den Fortbestand des Programms

Balance of member - contribution

Q43	Team *v_193*	**The team recognizes the specific characteristics (strengths and weaknesses) of the individual team members** Das Team erkennt die spezifischen Eigenschaften (Stärken und Schwächen) der einzelnen Teammitglieder

	Program *v_194*	**The program members recognize the specific characteristics (strengths and weaknesses) of the individual program members** Die Programmmitglieder erkennen die spezifischen Eigenschaften (Stärken und Schwächen) der einzelnen Programmmitglieder
Q44	Team *v_195*	**The team members contribute to the achievement of the team's goals in accordance with their specific potential** Die Teammitglieder tragen entsprechend ihrer spezifischen Fähigkeiten zur Erreichung der Teamziele bei
	Program *v_196*	**The program members contribute to the achievement of the program's goals in accordance with their specific potential** Die Programmmitglieder tragen entsprechend ihrer spezifischen Fähigkeiten zur Erreichung der Programmziele bei
Q45 *reverse coded*	Team *v_197*	**Imbalance of member contributions cause conflicts in our team** Das Ungleichgewicht der Beiträge der Teammitglieder führt zu Konflikten in unserem Team
	Program *v_198*	**Imbalance of member contributions cause conflicts in our program** Das Ungleichgewicht der Beiträge der Programmmitglieder führt zu Konflikten in unserem Programm

Team members' success - Work Satisfaction

Q46	Team *v_227*	**So far, the team can be pleased with its work** Bisher kann das Team mit seiner Arbeit zufrieden sein
	Program *v_228*	**So far, the program members can be pleased with its work** Bisher können die Programmmitglieder mit der Arbeit zufrieden sein
Q47	Team *v_229*	**The team members gain from the collaborative teamwork** Die Teammitglieder profitieren von der gemeinsamen Teamarbeit
	Program *v_230*	**The program members gain from the collaborative teamwork in the program** Die Programmmitglieder profitieren von der gemeinsamen Teamarbeit im Programm
Q48	Team *v_231*	**The team members will like to do this type of collaborative work again** Die Teammitglieder werden diese Art der Zusammenarbeit gerne wiederholen

	Program *v_232*	**The program members will like to do this type of collaborative work again** Die Programmmitglieder werden diese Art der Zusammenarbeit gerne wiederholen
Q49	Team *v_233*	**We are able to acquire important know-how through this teamwork** Durch diese Teamarbeit sind wir in der Lage, wichtiges Know-how zu erwerben
	Program *v_234*	**We are able to acquire important know-how through this teamwork in the program** Durch diese Teamarbeit im Programm sind wir in der Lage, wichtiges Know-how zu erwerben

Team members' success - Learning

Q50	Team *v_235*	**We consider this teamwork as a technical success** Wir betrachten diese Teamarbeit als echten technischen Erfolg
	Program *v_236*	**We consider this teamwork in the program as a technical success** Wir betrachten diese Teamarbeit im Programm als echten technischen Erfolg
Q51	Team *v_237*	**The team learn important lessons from this teamwork** Das Team kann aus dieser Teamarbeit wichtige Erkenntnisse ziehen
	Program *v_238*	**The program members learn important lessons from this teamwork in the program** Die Programmmitglieder können aus dieser Teamarbeit im Programm wichtige Erkenntnisse ziehen
Q52	Team *v_239*	**Teamwork promotes one personally** Teamarbeit fördert jeden persönlich
	Program *v_240*	**Teamwork in the program promotes one personally** Teamarbeit im Programm fördert jeden persönlich
Q53	Team *v_241*	**Teamwork promotes one professionally** Teamarbeit fördert einen professionell
	Program *v_242*	**Teamwork in the program promotes one professionally** Teamarbeit im Programm fördert einen professionell Team Performance

Team Performance - Effectiveness

Q54	Team *v_263*	**Going by the results, this teamwork can be regarded as successful** Ausgehend von den Ergebnissen kann das Teamwork als erfolgreich bewertet werden
	Program *v_264*	**Going by the results, this teamwork in the program can be regarded as successful** Ausgehend von den Ergebnissen kann das Teamwork im Programm als erfolgreich bewertet werden
Q55	Team *v_265*	**All demands of the customers are satisfied in the team** Alle Anforderungen der Kunden an das Team sind erfüllt
	Program *v_266*	**All demands of the customers are satisfied in the program** Alle Anforderungen der Kunden an das Programm sind erfüllt
Q56	Team *v_267*	**From the company's perspective, all team goals are achieved** Aus Unternehmenssicht werden alle Teamziele erreicht
	Program *v_268*	**From the company's perspective, all program goals are achieved** Aus Unternehmenssicht werden alle Programmziele erreicht
Q57	Team *v_269*	**The performance of the team advances our image to the customer** Die Leistung des Teams fördert unser Image beim Kunden
	Program *v_270*	**The performance of the program advances our image to the customer** Die Leistung des Programms fördert unser Image beim Kunden
Q58	Team *v_271*	**The teamwork result is of high quality** Das Ergebnis der Teamarbeit ist von hoher Qualität
	Program *v_272*	**The teamwork in the program result is of high quality** Das Ergebnis der Teamwork im Programm ist von hoher Qualität
Q59	Team *v_273*	**The customer is satisfied with the quality of the teamwork result** Der Kunde ist mit der Qualität der Arbeitsergebnisse im Team zufrieden
	Program *v_274*	**The customer is satisfied with the quality of the teamwork results in the program** Der Kunde ist mit der Qualität der Arbeitsergebnisse im Programm zufrieden
Q60	Team *v_275*	**The team is satisfied with the teamwork result** Das Team ist mit dem Ergebnis der Teamarbeit zufrieden

	Program *v_276*	**The program members are satisfied with the teamwork results in the program** Die Programmitglieder sind mit dem Ergebnis des Teamworks im Programm zufrieden
Q61	Team *v_277*	**The product produced in the team, requires little rework** Das im Team produzierte Produkt erfordert wenig überarbeitung
	Program *v_278*	**The product produced in the program, requires little rework** Das im Programm produzierte Produkt erfordert wenig überarbeitung
Q62	Team *v_279*	**The team product proves to be stable in operation** Das Produkt des Teams erweist sich als stabil im Betrieb
	Program *v_280*	**The program product proves to be stable in operation** Das Produkt des Programms erweist sich als stabil im Betrieb
Q63	Team *v_281*	**The team product proves to be robust in operation** Das Produkt des Teams erweist sich als robust im Betrieb
	Program *v_282*	**The program product proves to be robust in operation** Das Produkt des Programms erweist sich als robust im Betrieb

Team Performance - Efficiency

Q64	Team *v_303*	**The company is satisfied with how the teamwork progresses** Das Unternehmen ist mit dem Fortschritt der Teamarbeit zufrieden
	Program *v_304*	**The company is satisfied with how the teamwork in the program progresses** Das Unternehmen ist mit dem Fortschritt der Teamarbeit im Programm zufrieden
Q65	Team *v_305*	**Overall, the team works in a cost-efficient way** Insgesamt arbeitet das Team kosteneffizient
	Program *v_306*	**Overall, the program works in a cost-efficient way** Insgesamt arbeitet das Programm kosteneffizient
Q66	Team *v_307*	**Overall, the team works in a time-efficient way** Insgesamt arbeitet das Team zeiteffizient
	Program *v_308*	**Overall, the program works in a time-efficient way** Insgesamt arbeitet das Programm zeiteffizient
Q67	Team *v_309*	**The team is within schedule** Das Team ist im Zeitplan

	Program *v_310*	**The program is within schedule** Das Programm ist im Zeitplan
Q68	Team *v_311*	**The team is within budget** Das Team liegt im Rahmen des Budgets
	Program *v_312*	**The program is within budget** Das Programm liegt im Rahmen des Budgets

Table 8.2: Questions for the TWQ model indicators (all of them are 5 point Likert scaled with the additional option "don't know" [47])

8.2 Questionnaires of the Interviews

These are the questions that were asked to the respective interview partners. Since the interviews were conducted in German, the questions here are also in German.

Interview 1 - Agile Coach (06.04.2021)

1 Allgemeine Fragen zu Ihrer Rolle
1.1 Welche Rollenbezeichnung besitzt Ihre Tätigkeit im Unternehmen?
1.2 Seit wie vielen Jahren sind Sie im Bereich der agilen Softwareentwicklung tätig?
1.3 Welche Aufgaben und Verantwortungen haben Sie?
1.4 Wer sind Ihre Stakeholder?

2 Wahl des Scaling Agile Frameworks
2.1 Welche/s Scaling Agile Framework/s nutzen Sie aktuell in Ihrem agilen Programm?
2.2 Wie sind Sie auf das eingesetzte Scaling Agile Framework gestoßen?
2.3 Nach welchen Kriterien erfolgte die Wahl des Scaling Agile Frameworks?
2.4 Welche weiteren Kriterien sollten Ihrer Meinung nach in Betracht gezogen werden?
2.5 Welche Herausforderungen gab es bei der Wahl des eingesetzten Scaling Agile Frameworks?

3 Einführung des Scaling Agile Frameworks 3.1 Wer ist Wahl und Einführung des eingesetzten Scaling Agile Frameworks verantwortlich?
3.2 Wie lange dauerte die Einführung?
3.3 Wie sind Sie bei der Skalierung vorgegangen (z. B. Start mit Pilotprojekt)?
3.4 Wie wurden die Mitarbeiter bzgl. des eingesetzten Scaling Agile Frameworks geschult?
3.5 Welche Herausforderungen und Probleme hatten Sie bei der Einführung des Scaling Agile Frameworks?
3.6 Wie viel Anpassung war notwendig, um das Scaling Agile Framework in Ihr agiles Programm zu integrieren?
3.7 Was sind Ihre Lessons Learned in Bezug auf die Einführung des Frameworks?

4 Umsetzung und Rolle des Scaling Agile Frameworks

4.1 Wie hoch ist der Grad der Umsetzung?

4.2 Welche Rolle nimmt das Scaling Agile Framework im Programm ein?

4.3 Wie viel Umstellung in Ihrem Programm war bzw. ist notwendig, um das Framework zu nutzen?

4.4 Welche Einflussfaktoren konnten Sie während der Entwicklung identifizieren?

4.5 Wie laufen die PI-Planings ab?

5 Inter Team Koordination

5.1 Welche Arten von Abhängigkeiten gibt es zwischen den einzelnen Teams Ihres Programms?

5.2 Und wie werden diese gemanagt? Wer ist mit involviert?

5.3 Welche Herausforderungen gab es beim Managen der Abhängigkeiten? Und wie wurden sie gelöst?

5.4 Welche Arte(n) von Wissen wurde zwischen den Teams ausgetauscht?

5.5 Gab es Wissen, bei dem sich der Austausch zwischen den Teams als besonders schwierig herausgestellt hat?

5.6 Welche Methoden und Praktiken wurden verwendet, um Wissen zwischen den Teams auszutauschen?

5.7 Welche davon schätzen Sie persönlich davon als besonders wichtig ein?

5.8 Wie hat die Tatsache, dass mehrere Unternehmen gemeinsam an der Entwicklung beteiligt sind, den Wissensaustausch beeinflusst?

6 Teamwork Quality Model

6.1 Was ist Ihre Einschätzung zu den einzelnen Faktoren des Teamwork Quality Modells? (Fokus: linke Seite)

7 Retrospektive des Scaling Agile Frameworks

7.1 Ist das Framework Ihrer Meinung nach für das aktuelle Produkt geeignet?

7.2 Wird das eingesetzte Framework Ihren Vorstellungen gerecht?

7.3 Wo sehen Sie Verbesserungsmöglichkeiten von Scaling Agile Frameworks?

7.4 Welcher Vorteil ergibt sich bei der Nutzung eines Scaling Agile Frameworks in der Entwicklung?

7.5 Was sind aus Ihrer Sicht wichtige Faktoren für den Erfolg des aktuellen Programms?

Interview 2 - Product Owner (06.04.2021)

1 Allgemeine Fragen zu Ihrer Rolle

1.1 Welche Rollenbezeichnung besitzt Ihre Tätigkeit im Unternehmen?

1.2 Seit wie vielen Jahren sind Sie im Bereich der agilen Softwareentwicklung tätig?

1.3 Welche Aufgaben und Verantwortungen haben Sie?

1.4 Wer sind Ihre Stakeholder?

2 Wahl des Scaling Agile Frameworks
2.1 Welche/s Scaling Agile Framework/s nutzen Sie aktuell in Ihrem agilen Programm?
2.2 Wie sind Sie auf das eingesetzte Scaling Agile Framework gestoßen?
2.3 Nach welchen Kriterien erfolgte die Wahl des Scaling Agile Frameworks?
2.4 Welche weiteren Kriterien sollten Ihrer Meinung nach in Betracht gezogen werden?
2.5 Welche Herausforderungen gab es bei der Wahl des eingesetzten Scaling Agile Frameworks?

3 Einführung des Scaling Agile Frameworks
3.1 Wer ist Wahl und Einführung des eingesetzten Scaling Agile Frameworks verantwortlich?
3.2 Wie lange dauerte die Einführung?
3.3 Wie sind Sie bei der Skalierung vorgegangen (z. B. Start mit Pilotprojekt)?
3.4 Wie wurden die Mitarbeiter bzgl. des eingesetzten Scaling Agile Frameworks geschult?
3.5 Welche Herausforderungen und Probleme hatten Sie bei der Einführung des Scaling Agile Frameworks?
3.6 Wie viel Anpassung war notwendig, um das Scaling Agile Framework in Ihr agiles Programm zu integrieren?
3.7 Was sind Ihre Lessons Learned in Bezug auf die Einführung des Frameworks?

4 Umsetzung und Rolle des Scaling Agile Frameworks
4.1 Wie hoch ist der Grad der Umsetzung?
4.2 Welche Rolle nimmt das Scaling Agile Framework im Programm ein?
4.3 Wie viel Umstellung in Ihrem Programm war bzw. ist notwendig, um das Framework zu nutzen?
4.4 Welche Einflussfaktoren konnten Sie während der Entwicklung identifizieren?

Question set 5 and 6 were skipped due to lack of time

7 Customer Involvement
7.1 Wie werden Kunden und Nutzer in den Entwicklungsprozess einbezogen? (Z.b. bei PI-Plannings, Kontinuierlich, . . .)
7.2 Wie werden Kunden und Nutzer über den Fortschritt informiert?
7.3 Wie werden die Anforderungen der Kunden und Nutzer erhoben?
7.4 Welche Herausforderungen sind beim Einbeziehen der von Kunden und Nutzer die Skalierung der agilen Praktiken in Ihrem Programm aufgekommen?
7.5 Wie haben Sie die zuvor genannten Herausforderungen adressiert? Wurden bspw. bestimmte Methoden oder Praktiken angewandt oder Visualisierungen erstellt?
7.6 Welche Vorteile sind beim Einbeziehen der von Kunden und Nutzer die Skalierung der agilen Praktiken in Ihrem Programm entstanden?
7.7 Zu welchen Zeitpunkten wird Software released?

8 Retrospektive des Scaling Agile Frameworks

8.1 Ist das Framework Ihrer Meinung nach für das aktuelle Produkt geeignet?

8.2 Wird das eingesetzte Framework Ihren Vorstellungen gerecht?

8.3 Wo sehen Sie Verbesserungsmöglichkeiten von Scaling Agile Frameworks?

8.4 Welcher Vorteil ergibt sich bei der Nutzung eines Scaling Agile Frameworks in der Entwicklung?

8.5 Was sind aus Ihrer Sicht wichtige Faktoren für den Erfolg des aktuellen Programms?

Interview 3 - Leadership Team (07.04.2021)

1 Allgemeine Fragen zu Ihrer Rolle

1.1 Welche Rollenbezeichnung besitzt Ihre Tätigkeit im Unternehmen?

1.2 Seit wie vielen Jahren sind Sie im Bereich der agilen Softwareentwicklung tätig?

1.3 Welche Aufgaben und Verantwortungen haben Sie?

1.4 Wer sind Ihre Stakeholder?

2 Wahl des Scaling Agile Frameworks

2.1 Welche/s Scaling Agile Framework/s nutzen Sie aktuell in Ihrem agilen Programm?

2.2 Wie sind Sie auf das eingesetzte Scaling Agile Framework gestoßen?

2.3 Nach welchen Kriterien erfolgte die Wahl des Scaling Agile Frameworks?

2.4 Welche weiteren Kriterien sollten Ihrer Meinung nach in Betracht gezogen werden?

2.5 Welche Herausforderungen gab es bei der Wahl des eingesetzten Scaling Agile Frameworks?

3 Einführung des Scaling Agile Frameworks

3.1 Wie hoch ist der Grad der Skalierung des aktuellen Entwicklungsvorhabens?

3.2 Warum wurde eine Skalierung für notwendig gehalten?

3.3 Wer ist Wahl und Einführung des eingesetzten Scaling Agile Frameworks verantwortlich?

3.4 Wie lange dauerte die Einführung?

3.5 Wie sind Sie bei der Skalierung vorgegangen (z. B. Start mit Pilotprojekt)?

3.6 Wie wurden die Mitarbeiter bzgl. des eingesetzten Scaling Agile Frameworks geschult?

3.7 Welche Herausforderungen und Probleme hatten Sie bei der Einführung des Scaling Agile Frameworks?

3.8 Wie viel Anpassung war notwendig, um das Scaling Agile Framework in Ihr agiles Programm zu integrieren?

3.9 Was sind Ihre Lessons Learned in Bezug auf die Einführung des Frameworks?

4.1 Wie hoch ist der Grad der Umsetzung?

4.2 Welche Rolle nimmt das Scaling Agile Framework im Programm ein?

4.3 Wie viel Umstellung in Ihrem Programm war bzw. ist notwendig, um das Framework zu nutzen?

4.4 Welche Einflussfaktoren konnten Sie während der Entwicklung identifizieren?

5 Customer Involvement
5.1 Wie werden Kunden und Nutzer in den Entwicklungsprozess einbezogen? (Z.b. bei PI-Plannings, Kontinuierlich, ...)
5.2 Welche Herausforderungen sind beim Einbeziehen der von Kunden und Nutzer die Skalierung der agilen Praktiken in Ihrem Programm aufgekommen?
5.3 Wie haben Sie die zuvor genannten Herausforderungen adressiert? Wurden bspw. bestimmte Methoden oder Praktiken angewandt oder Visualisierungen erstellt?
5.4 Welche Vorteile sind beim Einbeziehen der von Kunden und Nutzer die Skalierung der agilen Praktiken in Ihrem Programm entstanden?

6 Herausforderungen
6.1 Welche Herausforderungen sind durch die Skalierung der agilen Praktiken in Ihrem Programm entstanden?
6.2 Wie haben Sie die zuvor genannten Herausforderungen adressiert? Wurden bspw. bestimmte Methoden oder Praktiken angewandt oder Visualisierungen erstellt?

7 Erfolgsgeschichten
7.1 Wie wird bei Ihnen der Erfolg agiler Initiativen gemessen?
7.2 Was sind aus Ihrer Sicht wichtige Faktoren für den Erfolg des aktuellen Programms?
7.3 Was sollte nach Ihrer Meinung bei der Skalierung agiler Praktiken nicht tun?

8 Retrospektive des Scaling Agile Frameworks
8.1 Ist das Framework Ihrer Meinung nach für das aktuelle Produkt geeignet?
8.2 Wird das eingesetzte Framework Ihren Vorstellungen gerecht?
8.3 Wo sehen Sie Verbesserungsmöglichkeiten von Scaling Agile Frameworks?

Interview 4 - Agile Coach (07.04.2021)

identical to interview 1

Interview 5 - Product Owner (12.04.2021)

1 Allgemeine Fragen zu Ihrer Rolle
1.1 Welche Rollenbezeichnung besitzt Ihre Tätigkeit im Unternehmen?
1.2 Seit wie vielen Jahren sind Sie im Bereich der agilen Softwareentwicklung tätig?
1.3 Welche Aufgaben und Verantwortungen haben Sie?
1.4 Wer sind Ihre Stakeholder?

2 Wahl des Scaling Agile Frameworks

2.1 Welche/s Scaling Agile Framework/s nutzen Sie aktuell in Ihrem agilen Programm?
2.2 Wie sind Sie auf das eingesetzte Scaling Agile Framework gestoßen?
2.3 Nach welchen Kriterien erfolgte die Wahl des Scaling Agile Frameworks?
2.4 Welche weiteren Kriterien sollten Ihrer Meinung nach in Betracht gezogen werden?
2.5 Welche Herausforderungen gab es bei der Wahl des eingesetzten Scaling Agile Frameworks?

3 Einführung des Scaling Agile Frameworks
3.1 Wer ist Wahl und Einführung des eingesetzten Scaling Agile Frameworks verantwortlich?
3.2 Wie lange dauerte die Einführung?
3.3 Wie sind Sie bei der Skalierung vorgegangen (z. B. Start mit Pilotprojekt)?
3.4 Wie wurden die Mitarbeiter bzgl. des eingesetzten Scaling Agile Frameworks geschult?
3.5 Welche Herausforderungen und Probleme hatten Sie bei der Einführung des Scaling Agile Frameworks?
3.6 Wie viel Anpassung war notwendig, um das Scaling Agile Framework in Ihr agiles Programm zu integrieren?
3.7 Was sind Ihre Lessons Learned in Bezug auf die Einführung des Frameworks?

4 Umsetzung und Rolle des Scaling Agile Frameworks
4.1 Wie hoch ist der Grad der Umsetzung?
4.2 Welche Rolle nimmt das Scaling Agile Framework im Programm ein?
4.3 Wie viel Umstellung in Ihrem Programm war bzw. ist notwendig, um das Framework zu nutzen?
4.4 Welche Einflussfaktoren konnten Sie während der Entwicklung identifizieren?

5 Inter Team Koordination
5.1 Welche Arten von Abhängigkeiten gibt es zwischen den einzelnen Teams Ihres Programms?
5.2 Und wie werden diese gemanagt? Wer ist mit involviert?
5.3 Welche Herausforderungen gab es beim Managen der Abhängigkeiten? Und wie wurden sie gelöst?
5.4 Welche Arte(n) von Wissen wurde zwischen den Teams ausgetauscht?
5.5 Wissen die Teams stets, an was die anderen Teams gerade arbeiten?
5.6 Waren Sie in der Lage, den Workload zwischen den Teams zu verteilen/verschieben? Warum war das einfach/schwierig?
5.7 Welche Tools haben Sie zur Unterstützung verwendet?
5.8 Wie hat die Tatsache, dass mehrere Unternehmen gemeinsam an der Entwicklung beteiligt sind, die Verteilung der Aufgaben beeinflusst?

6 Teamwork Quality Model
6.1 Was ist Ihre Einschätzung zu den einzelnen Faktoren des Teamwork Quality Modells? (Fokus: linke Seite)

7 Customer Involvement

7.1 Wie werden Kunden und Nutzer in den Entwicklungsprozess einbezogen? (Z.b. bei PI-Plannings, Kontinuierlich, . . .)

7.2 Wie werden Kunden und Nutzer über den Fortschritt informiert?

7.3 Wie werden die Anforderungen der Kunden und Nutzer erhoben?

7.4 Welche Herausforderungen sind beim Einbeziehen der von Kunden und Nutzer die Skalierung der agilen Praktiken in Ihrem Programm aufgekommen?

7.5 Wie haben Sie die zuvor genannten Herausforderungen adressiert? Wurden bspw. bestimmte Methoden oder Praktiken angewandt oder Visualisierungen erstellt?

7.6 Welche Vorteile sind beim Einbeziehen der von Kunden und Nutzer die Skalierung der agilen Praktiken in Ihrem Programm entstanden?

7.7 Zu welchen Zeitpunkten wird Software released?

8 Retrospektive des Scaling Agile Frameworks

8.1 Ist das Framework Ihrer Meinung nach für das aktuelle Produkt geeignet?

8.2 Wird das eingesetzte Framework Ihren Vorstellungen gerecht?

8.3 Wo sehen Sie Verbesserungsmöglichkeiten von Scaling Agile Frameworks?

8.4 Welcher Vorteil ergibt sich bei der Nutzung eines Scaling Agile Frameworks in der Entwicklung?

8.5 Was sind aus Ihrer Sicht wichtige Faktoren für den Erfolg des aktuellen Programms?

Interview 6 - System Architect (19.04.2021)

1 Allgemeine Fragen zu Ihrer Rolle

1.1 Welche Rollenbezeichnung besitzt Ihre Tätigkeit im Unternehmen?

1.2 Seit wie vielen Jahren sind Sie im Bereich der agilen Softwareentwicklung tätig?

1.3 Welche Aufgaben und Verantwortungen haben Sie?

1.4 Wer sind Ihre Stakeholder?

2 Rolle von Architekten

2.1 Wer ist bei Ihnen für die Architekturaktivitäten verantwortlich?

2.2 Wer trifft bei Ihnen Architekturentscheidungen?

2.3 Wie werden Architekturentscheidungen getroffen, z. B. nach welchem Prinzip bzw. Modell?

2.4 Wer ist bei Ihnen in Architekturaktivitäten involviert und in welcher Art und Weise?

2.5 Wie stehen die Entwickler zur Architektur?

2.6 Ist die Rolle des Architekten in agilen Projekten Ihrer Meinung nach notwendig?

3 Agile Architecting

3.1 Wie erfolgt bei Ihnen Architecting?

3.2 Zu welchem Zeitpunkt wird bei Ihnen Architecting durchgeführt?

3.3 In welchen Prozessen bzw. Events wird Architecting durchgeführt (z. B. Event Storming Workshop)?
3.4 Wie werden Architekturentscheidungen kommuniziert?
3.5 Wie erfolgt die Dokumentation der Architektur?
3.6 Welche Architekturmodelle werden in Ihrem Programm eingesetzt (z. B. Domänenmodell, Business Capability Maps)?
3.7 Welche Architekturprinzipien verwenden Sie?
3.8 Welche Ziele verfolgen Sie mit Ihrer Architektur?
3.9 Wie messen Sie den Erfolg der Architektur?
3.10 Welche großen Architekturentscheidungen gab es?
3.11 Welche Auswirkungen hatten diese?
3.12 Welchen Einfluss hat die Architektur auf die Produkt-Eigenschaften?

4 Herausforderungen bei der agilen Architektur
4.1 Mit welchen Herausforderungen müssen die Architekten umgehen?
4.2 Wie wurden die zuvor genannten Herausforderungen adressiert? Wurden bspw. bestimmte Methoden oder Praktiken angewandt oder Visualisierungen erstellt?

5. Tooling
5.1 Welche Architekturtools nutzen Sie und zu welchem Zweck?

6. Lessons Learned
6.1 Was sind Ihre Lessons Learned bezüglich Architekturaktivitäten, -verantwortlichkeiten und -dokumentationen?
6.2 Was sind aus Ihrer Sicht wichtige Faktoren für den Erfolg des aktuellen Programms?

Interview 7 - Leadership Team (21.04.2021)

1 Allgemeine Fragen zu Ihrer Rolle
1.1 Welche Rollenbezeichnung besitzt Ihre Tätigkeit im Unternehmen?
1.2 Seit wie vielen Jahren sind Sie im Bereich der agilen Softwareentwicklung tätig?
1.3 Welche Aufgaben und Verantwortungen haben Sie?
1.4 Wer sind Ihre Stakeholder?

2 Wahl des Scaling Agile Frameworks
2.1 Welche/s Scaling Agile Framework/s nutzen Sie aktuell in Ihrem agilen Programm?
2.2 Wie sind Sie auf das eingesetzte Scaling Agile Framework gestoßen?
2.3 Nach welchen Kriterien erfolgte die Wahl des Scaling Agile Frameworks?
2.4 Welche weiteren Kriterien sollten Ihrer Meinung nach in Betracht gezogen werden?
2.5 Welche Herausforderungen gab es bei der Wahl des eingesetzten Scaling Agile Frameworks?

3 Einführung des Scaling Agile Frameworks

3.1 Wie hoch ist der Grad der Skalierung des aktuellen Entwicklungsvorhabens?
3.2 Warum wurde eine Skalierung für notwendig gehalten?
3.3 Wer ist Wahl und Einführung des eingesetzten Scaling Agile Frameworks verantwortlich?
3.4 Wie lange dauerte die Einführung?
3.5 Wie sind Sie bei der Skalierung vorgegangen (z. B. Start mit Pilotprojekt)?
3.6 Welche Herausforderungen und Probleme hatten Sie bei der Einführung des Scaling Agile Frameworks?
3.7 Was sind Ihre Lessons Learned in Bezug auf die Einführung des Frameworks?

4.1 Wie hoch ist der Grad der Umsetzung?
4.2 Welche Rolle nimmt das Scaling Agile Framework im Programm ein?
4.3 Wie viel Umstellung in Ihrem Programm war bzw. ist notwendig, um das Framework zu nutzen?

5 Customer Involvement
5.1 Wie werden Kunden und Nutzer in den Entwicklungsprozess einbezogen? (Z.b. bei PI-Plannings, Kontinuierlich, . . .)
5.2 Welche Herausforderungen sind beim Einbeziehen der von Kunden und Nutzer die Skalierung der agilen Praktiken in Ihrem Programm aufgekommen?
5.3 Wie haben Sie die zuvor genannten Herausforderungen adressiert? Wurden bspw. bestimmte Methoden oder Praktiken angewandt oder Visualisierungen erstellt?
5.4 Welche Vorteile sind beim Einbeziehen der von Kunden und Nutzer die Skalierung der agilen Praktiken in Ihrem Programm entstanden?

6 Herausforderungen
6.1 Welche Herausforderungen sind durch die Skalierung der agilen Praktiken in Ihrem Programm entstanden?
6.2 Wie haben Sie die zuvor genannten Herausforderungen adressiert? Wurden bspw. bestimmte Methoden oder Praktiken angewandt oder Visualisierungen erstellt?

7 Erfolgsgeschichten
7.1 Wie wird bei Ihnen der Erfolg agiler Initiativen gemessen?
7.2 Was sind aus Ihrer Sicht wichtige Faktoren für den Erfolg des aktuellen Programms?
7.3 Was sollte nach Ihrer Meinung bei der Skalierung agiler Praktiken nicht tun?

8 Retrospektive des Scaling Agile Frameworks
8.1 Ist das Framework Ihrer Meinung nach für das aktuelle Produkt geeignet?
8.2 Wird das eingesetzte Framework Ihren Vorstellungen gerecht?

List of Figures

List of Tables

Acronyms

API application programming interface. 39

ART Agile Release Train. 17–19, 37, 38, 95

CoP Community of Practice. 17, 39

DAD Disciplined Agile Delivery. 14

DSDM dynamic system development method. 6, 7

FDD feature-driven development. 7

LeSS Large Scale Scrum. 2, 14–16, 95

PI Product Increment. 18, 19, 34, 41, 60, 64, 66

PoC proof of concept. 39

RMSEA root mean squared error of approximation. 25, 45, 52–54, 58, 59, 62

RTE Release Train Engineer. 18

SAFe Scaled Agile Framework. 2, 8, 11, 14, 16, 17, 19–21, 30, 32–37, 39, 40, 60, 61, 67, 68, 70, 95

SEM structural equation modeling. 3, 4, 44, 45, 65, 70

SRMR standardized root mean squared residual. 45, 52–54, 59, 62, 65

TWQ model Teamwork Quality model. v, 3–6, 21, 22, 24–26, 32, 42–45, 47, 51, 52, 54, 58, 60–67, 69–71, 73, 86, 96

VIF variance influence factor. 48, 51, 55, 65

WIP work in progress. 20

Bibliography

[1] Mark D. Alicke, David A. Dunning, and Joachim Krueger. *The Self in Social Judgment*. Psychology Press, 2005.

[2] Mark D. Alicke, Mary. L. Klotz, David L. Breitenbecher, Tricia J. Yurak, and et al. Personal contact, individuation, and the better-than-average effect. *Journal of Personality and Social Psychology*, 68(5), 1995.

[3] Len Bass, Ingo Weber, and Liming Zhu. *DevOps: A Software Architect's Perspective*. Addison-Wesley Professional, 2015.

[4] Kent Beck and Cynthia Andres. *Extreme programming explained: Embrace change*. The XP series. Addison-Wesley, Boston, 2. ed., 6. printing edition, 2007.

[5] Izak Benbasat, David K. Goldstein, and Melissa Mead. The case research strategy in studies of information systems. *MIS Quarterly*, 11(3), 1987.

[6] Oliver Bendel. Definition: Digitalisierung. *Springer Fachmedien Wiesbaden GmbH*, 19.02.2018.

[7] Saskia Bick, Alexander Scheerer, and Kai Spohrer. Inter-team coordination in large agile software development settings: Five ways of practicing agile at scale. In *Proceedings of the Scientific Workshop Proceedings of XP2016*, pages 1–5, New York, NY, USA, 2016. ACM.

[8] Barry W. Boehm. A spiral model of software development and enhancement. *Computer*, 21(5), 1988.

[9] Michael T. Brannick, Ashley Prince, Carolyn Prince, and Eduardo Salas. The measurement of team process. *Human factors*, 37(3), 1995.

[10] Bundesrepublik Deutschland. V-modell xt.

[11] Brian Burke. Top strategic technology trends for 2021. *Gartner*, 2020, 2020.

[12] Michael A. Campion, Gina J. Medsker, and Catherine A. Higgs. Relations between work group characteristics and effectiveness: Implications for designing effective work groups. *Personnel Psychology*, 46(4), 1993.

[13] Dorwin Cartwright and Alvin Zander, editors. *Group Dynamics: Research and Theory, Third Edition Edited by Dorwin Cartwright and Alvin Zander (Book Review)*, volume 19. Routledge, 1969.

[14] Luisanna Cocco, Katiuscia Mannaro, Giulio Concas, and Michele Marchesi. Simulating kanban and scrum vs. waterfall with system dynamics. In 12th International Conference, editor, *Agile Processes in Software Engineering and Extreme Programming*. Springer, Berlin, Heidelberg, 2011.

[15] John W. Creswell and Vicki L. P. Clark. *Designing and conducting mixed methods research*. SAGE, Los Angeles and London and New Delhi and Singapore and Washington DC and Melbourne, third edition edition, 2018.

[16] Daniel R. Denison, Stuart L. Hart, and Joel A. Kahn. From chimneys to cross-functional teams: Developing and validating a diagnostic model. *Academy of Management Journal*, 39(4), 1996.

[17] digital.ai. 1st annual state of agile report, 2007.

[18] digital.ai. 14th annual state of agile report, 2020.

[19] digital.ai. 15th annual state of agile report, 2021.

[20] Kim Dikert, Maria Paasivaara, and Casper Lassenius. Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, 119, 2016.

[21] Torgeir Dingsøyr, Tor E. Fægri, and Juha Itkonen. What is large in large-scale? a taxonomy of scale for agile software development. In *Product-Focused Software Process Improvement*. Springer, Cham, 2014.

[22] Torgeir Dingsøyr, Nils B. Moe, Tor E. Fægri, and Eva A. Seim. Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation. *Empirical Software Engineering*, 23(1), 2018.

[23] Torgeir Dingsøyr, Knut Rolland, Nils B. Moe, and Eva A. Seim. Coordination in multi-team programmes: An investigation of the group mode in large-scale agile software development. *Procedia Computer Science*, 121, 2017.

[24] Maximilian Doepp. Using multiteam systems theory and team work quality to identify influence factors for measuring the performance of agile teams in large-scale agile development.

[25] Christof Ebert and Maria Paasivaara. Scaling agile. *IEEE Software*, 34(6), 2017.

[26] Floris M. A. Erich, Chintan Amrit, and Maya Daneva. A qualitative study of devops usage in practice. *Journal of Software: Evolution and Process*, 29(6), 2017.

[27] Alberto J. Espinosa, Ning Nan, and Erran Carmel. Temporal distance, communication patterns, and task performance in teams. *Journal of Management Information Systems*, 32(1), 2015.

[28] Alberto J. Espinosa, Sandra A. Slaughter, Robert E. Kraut, and James D. Herbsleb. Familiarity, complexity, and team performance in geographically distributed software development. *Organization Science*, 18(4), 2007.

[29] Darren George and Paul Mallery. *SPSS for Windows step by step: A simple guide and reference, 11.0 update*. A & B, Boston, 4th ed. edition, 2003.

[30] Deborah L. Gladstein. Groups in context: A model of task group effectiveness. *Administrative Science Quarterly*, 29(4), 1984.

[31] Robert K. Greenleaf, Larry C. Spears, and Stephen R. Covey, editors. *Servant leadership: A journey into the nature of legitimate power and greatness*. Paulist Press, New York and Mahwah, N.J., 25th anniversary edition edition, 2002.

[32] Richard J. Hackman. *Leading teams: Setting the stage for great performances*. Harvard Business School Press, Boston, Mass., 2002.

[33] Richard J. Hackman and J. Lorsch. The design of work teams. *Handbook of organizational behavior*, 1987.

[34] Joseph F. Hair, Tomas G. M. Hult, Christian M. Ringle, and Marko Sarstedt. *A primer on partial least squares structural equation modeling (PLS-SEM)*. SAGE, Los Angeles and London and New Delhi and Singapore and Washington DC and Melbourne, second edition edition, 2017.

[35] Malcolm Higgs, Ulrich Plewnia, and Jorg Ploch. Influence of team composition and task complexity on team performance. *Team Performance Management: An International Journal*, 11(7/8), 2005.

[36] Martin Hoegl and Hans G. Gemünden. Teamwork quality and the success of innovative projects: A theoretical concept and empirical evidence. *Organization Science*, 12(4), 2001.

[37] George Homans. *Social behavior; its elementary forms*. Harcourt Brace Jovanovich, New York, rev. ed. edition, 1974.

[38] Rick H. Hoyle. *Structural equation modeling: Concepts, issues, and applications*. Sage Publ, Thousand Oaks, 1995.

[39] Li-tze Hu and Peter M. Bentler. Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. *Structural Equation Modeling: A Multidisciplinary Journal*, 6(1), 1999.

[40] ILX Group. Prince2: Informationen & prince2 kurse für projektmanager — de, 2021.

[41] Matthias Janson. Deutschland fehlen it-experten. *Statista*, 13.01.2021.

[42] Ralph Katz. The effects of group longevity on project communication and performance. *Administrative Science Quarterly*, 27(1), 1982.

[43] James R. Larson Jr. and Linda J. Schaumann. Group goals, group coordination, and group member motivation. *Human Performance*, 6(1), 1993.

[44] Dean Leffingwell. Safe 5.1 framework, 2021.

[45] Dean Leffingwell and Drew Jemilo. Five-core-competencies-for-the-lean-enterprise. 2018.

[46] Patrick Lencioni. *The five dysfunctions of a team: A leadership fable*, volume v.13 of *J-B Lencioni Series*. Jossey-Bass, San Francisco, 1st ed. edition, 2002.

[47] Rensis Likert. A technique for the measurement of attitudes. *Archives of Psychology, 22 140, 55*, 1932.

[48] Yngve Lindsjørn, Gunnar R. Bergersen, Torgeir Dingsøyr, and Dag I. K. Sjøberg. Teamwork quality and team performance: Exploring differences between small and large agile projects. In Juan Garbajosa, Xiaofeng Wang, and Ademar Aguiar, editors, *Agile Processes in Software Engineering and Extreme Programming*, Lecture Notes in Business Information Processing, Cham, 2018. Springer International Publishing.

[49] Yngve Lindsjørn, Dag I. K. Sjøberg, Torgeir Dingsøyr, Gunnar R. Bergersen, and Tore Dybå. Teamwork quality and project success in software development: A survey of agile development teams. *Journal of Systems and Software*, 122, 2016.

[50] Mei Lu, Mary B. Watson-Manheim, Katherine M. Chudoba, and Eleanor Wynn. Virtuality and team performance: Understanding the impact of variety of practices. *Journal of Global Information Technology Management*, 9(1), 2006.

[51] Lucy E. Lwakatare, Pasi Kuvaja, and Markku Oivo. Dimensions of devops. In *Agile Processes in Software Engineering and Extreme Programming*. Springer, Cham, 2015.

[52] George A. Marcoulides and Randall E. Schumacker. *Advanced Structural Equation Modeling: Issues and Techniques*. Taylor and Francis, Hoboken, 2013.

[53] Anthony Mersino. Agile project success rates are 2x higher than traditional projects, 2018.

[54] Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ken Schwaber, and Jeff Sutherland. Manifesto for agile software development, 2001.

[55] Eric Naiburg. The nexus™ guide: The definitive guide to scaling scrum with nexus, 2021.

[56] Robert M. O'brien. A caution regarding rules of thumb for variance inflation factors. *Quality & Quantity*, 41(5), 2007.

[57] Alexander Osterwalder and Yves Pigneur. *Business model generation: A handbook for visionaries, game changers, and challengers*. Wiley&Sons, New York, 2013.

[58] Maria Paasivaara, Benjamin Behm, Casper Lassenius, and Minna Hallikainen. Large-scale agile transformation at ericsson: a case study. *Empirical Software Engineering*, 23(5), 2018.

[59] Bastian Pauly and Adél Holdampf-Wendel. 86.000 offene stellen für it-fachkräfte. *Bitkom e.V.*, 2020, 16.12.2020.

[60] Kai Peterson. Is lean agile and agile lean? a comparison between two software development paradigms. In *Modern Software Engineering Concepts and practices: Advanced Approaches*. IGI Global, 2011.

[61] Mary B. Pinto and Jeffrey K. Pinto. Project team communication and cross-functional cooperation in new program development. *Journal of Product Innovation Management*, 7(3), 1990.

[62] Mary Poppendieck and Tom Poppendieck. *Lean software development: An agile Toolkit for Software Development Manager*. The agile software development series. Addison-Wesley, Boston, 2003.

[63] Colin Robson. *Real world research: A resource for social scientists and practitioner-researchers*. Blackwell, Malden, MA, 2 edition, 2002.

[64] Walker W. Royce. Managing the development of large software systems: concepts and techniques. *Proc. IEEE WESTCON, Los Angeles*, 1970.

[65] Julia Rozovsky. The five keys to a successful google team, 2015.

[66] Robert F. Russell and Gregory A. Stone. A review of servant leadership attributes: developing a practical model. *Leadership & Organization Development Journal*, 23(3), 2002.

[67] Ken Schwaber. Scrum development process, 1994.

[68] Ken Schwaber and Mike Beedle. *Agile software development with Scrum*. Series in agile software development. Prentice Hall, Upper Saddle River, NJ, 2002.

[69] Ken Schwaber and Jeff Sutherland. The scrum guide, 2017.

[70] Ken Schwaber and Jeff Sutherland. The scrum guide: The definitive guide to scrum: The rules of the game, 2020.

[71] ScrumGuides.org. Scrum guide revisions — scrum guides, 2020.

[72] Scrum.org. The scrum framework poster, 20.03.2021.

[73] Scrum.org. The nexus™ guide, 29.03.2021.

[74] Anson Seers. Team-member exchange quality: A new construct for role-making research. *Organizational Behavior and Human Decision Processes*, 43(1), 1989.

[75] Larry C. Spears and Robert K. Greenleaf. *Reflections on leadership: How Robert K. Greenleaf's theory of servant-leadership influenced today's top management thinkers*. Wiley, New York NY u.a., 1995.

[76] Ralph D. Stacey. *Strategic management and organisational dynamics: The challenge of complexity*. Financial Times, London, 3. ed. edition, 2000.

[77] Jennifer Stapleton. *DSDM, dynamic systems development method: The method in practice*. Addison-Wesley, Harlow, reprint edition, 1998.

[78] Eric Sundstrom, Kenneth P. de Meuse, and David Futrell. Work teams: Applications and effectiveness. *American Psychologist*, 45(2), 1990.

[79] Jeff Sutherland. Agile can scale: Inventing and reinventing scrum in five companies. *IT Cutter IT journal*, 2001(Vol. 14, No. 12), 2001.

[80] Jeff Sutherland. The scrum@scale guide: The definitive guide to scrum@scale: Scaling that works, 2006.

[81] Hirotaka Takeuchi and Ikujiro Nonaka. The new new product development game. *HARVARD BUSINESS REVIEW*, 1986.

[82] Scott I. Tannenbaum, Rebecca L. Beard, and Eduardo Salas. Team building and its influence on team effectiveness: an examination of conceptual and empirical developments. In Louise Kelley, editor, *Issues, Theory, and Research in Industrial/Organizational Psychology*, volume 82 of *Advances in Psychology*. Elsevier textbooks, s.l., 1992.

[83] Abbas Tashakkori and Charles Teddlie, editors. *Sage handbook of mixed methods in social & behavioral research*. SAGE, Los Angeles and London and New Dehli and Singapore and Wshington D.C., second edition edition, 2010.

[84] Mohsen Tavakol and Reg Dennick. Making sense of cronbach's alpha. *International Journal of Medical Education*, 2, 2011.

[85] Dean Tjosvold. Cooperation theory and organizations. *Human Relations*, 37(9), 1984.

[86] Ömer Uludağ, Nina-Mareike Harders, and Florian Matthes. Documenting recurring concerns and patterns in large-scale agile development. In Tiago Boldt, editor, *Proceedings of the 24th European Conference on Pattern Languages of Programs - EuroPLop '19*, New York, New York, USA, 2019. ACM Press.

[87] Ömer Uludağ, Martin Kleehaus, Xian Xu, and Florian Matthes. Investigating the role of architects in scaling agile frameworks. In *2017 IEEE 21st International Enterprise Distributed Object Computing Conference (EDOC)*, pages 123–132, 2017.

[88] Nils Urbach and Frederik Ahlemann. Die digitale revolution – wie technologische trends die business-welt verändern. In Nils Urbach and Frederik Ahlemann, editors, *IT-Management im Zeitalter der Digitalisierung*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[89] Nils Urbach and Frederik Ahlemann. *IT-Management im Zeitalter der Digitalisierung: Auf dem Weg zur IT-Organisation der Zukunft*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[90] Nils Urbach and Frederik Ahlemann. Kein business ohne it – it ist der zentrale und unverzichtbare treiber unternehmerischer wertschöpfung. In Nils Urbach and Frederik Ahlemann, editors, *IT-Management im Zeitalter der Digitalisierung*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[91] Dirk van Dierendonck. Servant leadership: A review and synthesis. *Journal of Management*, 37(4), 2011.

[92] Bas Vodde and Craig Larman. Less framework - large scale scrum (less), 2021.

[93] Emily Weimar, Ariadi Nugroho, Joost Visser, Aske Plaat, Martijn Goudbeek, and Alexander P. Schouten. The influence of teamwork quality on software team performance.

[94] Jörg Weking, Andreas Hein, Markus Böhm, and Helmut Krcmar. A hierarchical taxonomy of business model patterns. *Electronic Markets*, 30(3), 2020.

[95] Susan A. Wheelan. Group size, group development, and group productivity. *Small Group Research*, 40(2), 2009.

[96] James P. Womack and Daniel T. Jones. *Lean Thinking: Banish waste and create wealth in your corporation.* Simon & Schuster, London, rev. and updated, 1. paperback ed. edition, 2003.

[97] James P. Womack, Daniel T. Jones, and Daniel Roos. *The machine that changed the world: The Story of Lean Production: How Japan's secret weapon in the global auto wars will revolutionize Western industry.* HarperPerennial, New York, NY, 2006.

[98] World Health Organization. Coronavirus disease (covid-19) outbreak, 2021.