TUM SCHOOL OF COMPUTATION,
INFORMATION AND TECHNOLOGY (CIT)

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Information Systems

Challenges and Mitigation Propositions for Effort Estimation in Large-Scale Agile Development

Karla Eleonore Weigelt

TUM SCHOOL OF COMPUTATION,
INFORMATION AND TECHNOLOGY (CIT)

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Information Systems

Challenges and Mitigation Propositions for Effort Estimation in Large-Scale Agile Development

Herausforderungen und Vorschläge zur Problemlösung bei der Aufwandsschätzung in der großangelegten agilen Entwicklung

Author:	Karla Eleonore Weigelt
Supervisor:	Prof. Dr. Florian Matthes
Advisor:	Franziska Tobisch, M.Sc.
Submission Date:	May 15, 2023

I confirm that this bachelor's thesis in information systems is my own work and I have documented all sources and material used.

Munich, May 15, 2023 Karla Eleonore Weigelt

Abstract

The contemporary business environment is characterized by a high degree of uncertainty, particularly triggered by rapid technological progress and constantly changing customer needs. This is especially true in the field of software development, which is of a fast-paced and constantly changing nature. Therefore it is essential for companies nowadays to be flexible and adaptable to remain successful and competitive. As the environment and technology evolved, traditional methodologies, such as the Waterfall approach, have become obsolete due to their limitations in flexibility and adaptability, often resulting in project failure. To counteract these and other disadvantages, new approaches, such as the agile methodology, have been developed that compensate for these disadvantages of traditional methodologies. Originally, agile methodologies were designed for small-scale projects to respond quickly to changes in the environment and customer requirements. The success of applying agile methodologies on a small scale has inspired many organizations to use these in larger contexts leading to the increasing popularity of agile methods on large-scale projects. Applying agile practices on a large scale has its own additional challenges, e.g., the increasing complexity of projects or the maintenance of oversight regarding the project's progress. To address these challenges, accurate predictions, e.g., effort estimations, can support the project managers in tracking the progress of a project enabling delivery within budget and time. In this context, the accuracy of the predictions is of great relevance and importance since any percentage error in the overall effort estimate may lead to project failure. To date, numerous publications regarding effort estimation at a team level exist, but only little with a focus on large-scale agile environments. However, even those lack guidance on how to perform effort estimation in scaling agile environments but only, e.g., research on the estimation techniques and units used in this context. To address these research gaps, this thesis investigates the effort estimation process in scaling agile environments in practice, identifies challenges practitioners face during effort estimation, and derives mitigation propositions. In doing so, this research follows an Action Design Research (ADR) approach to generate prescriptive design knowledge by constructing and evaluating ensemble artifacts in an organizational setting. For this purpose, we conducted a case study, including 16 interviews, on a large project of a major German network and infrastructure company, also involving a large German software company and a German subsidiary of an American consulting firm. After collecting and transcribing the data, a two-cycle approach is applied to coding the data to analyze it then. The main findings of the case study are used to develop the artifact of this thesis - a set of challenges and mitigation propositions for effort estimation in large-scale agile development. Further, findings in existing research, especially on the mitigation propositions, are incorporated into the artifact. The artifact is then evaluated by practitioners and refined by incorporating the experts' improvement suggestions and qualitative feedback.

Contents

Abstract **iv**

1. Introduction **1**
1.1. Motivation . . . 1
1.2. Research Questions . . . 3
1.3. Research Methodology . . . 3

2. Foundation **7**
2.1. Agile Development . . . 7
2.1.1. Movement towards Agile . . . 7
2.1.2. The Scrum Framework . . . 8
2.2. Large-Scale Agile Development . . . 11
2.2.1. Definition of Large-Scale Agile Development . . . 11
2.2.2. Large-Scale Agile Development . . . 12
2.2.3. Scaling Factors and Challenges . . . 12
2.2.4. Large-Scale Agile Standard Software Implementation . . . 14
2.2.5. Difference between Large-Scale Agile Development and Agile ERP-Implementation . . . 14
2.3. Effort Estimation in Agile Development . . . 15
2.3.1. Overview on Effort Estimation . . . 15
2.3.2. Estimation Context . . . 16
2.3.3. Estimation Predictors . . . 17
2.3.4. Effort Estimation . . . 18
2.3.5. Effort Estimation in Scrum . . . 21
2.3.6. Effort Estimation in Large-Scale Software Development . . . 22

3. Related Work **23**
3.1. Effort Estimation in Agile Development: Process, Challenges, and Mitigation Propositions . . . 23
3.2. Effort Estimation in Large-Scale Agile Development: Process, Challenges, and Mitigation Propositions . . . 26
3.3. Effort Estimation in Large-Scale Agile Standard Software Implementation: Process, Challenges, and Mitigation Propositions . . . 30

4. The Case Study **32**
4.1. Study Design . . . 32
4.2. Data Collection . . . 34

4.3. Data Analysis . . . 34
4.4. Case Description . . . 35
4.4.1. The Organizations . . . 36
4.4.2. The Program Hierarchy . . . 38
4.4.3. The Effort Estimation Process . . . 40
4.4.4. The Challenges . . . 48

5. The Mitigation Propositions **54**

6. Artifact Evaluation **62**
6.1. Methodology . . . 62
6.2. Evaluation . . . 63
6.2.1. Evaluation of the Challenges . . . 63
6.2.2. Evaluation of the Mitigation Propositions . . . 69
6.3. Adjustments based on Evaluation . . . 73

7. Discussion **77**
7.1. Key Findings . . . 77
7.2. Limitations . . . 80

8. Conclusion **83**
8.1. Summary . . . 83
8.2. Future Work . . . 85

A. Appendix A **86**
A.1. Key Values and Principles behind the Agile Manifesto . . . 86

B. Appendix B **87**
B.1. Questionnaire of Case Study Interviews . . . 87
B.2. Questionnaire of Evaluation Survey . . . 88

C. Appendix C **97**
C.1. Detailed Results of the Case Study . . . 97
C.2. Detailed Results of the Evaluation . . . 98

D. Appendix D **100**
D.1. Adjustments to the Challenges . . . 100
D.2. Adjustments to the Mitigation Proposition . . . 104

Bibliography **109**

1. Introduction

In the first chapter, the motivation, research objectives, and research approach of this bachelor's thesis are presented. Section 1.1. outlines the motivation for the research topic and discusses the relevance and necessity of this research project. Section 1.2. explains the objectives and research questions that will be answered throughout this thesis. Section 1.3. describes the research approach that will be used to achieve the discussed research objectives.

1.1. Motivation

In the contemporary business environment, which is characterized by high levels of unpredictability resulting from unstable customer needs and rapid technological progress, the ability to be flexible and adaptable has become essential to a company's success [60, 69, 98]. This applies in particular to the field of software development, which is characterized by its fast-paced and rapidly-changing nature. Traditional methodologies, such as the Waterfall approach, have been found to be inadequate in this context due to their limitations in terms of flexibility and adaptability [34]. As a response to this, the agile movement emerged, leading to the development of many agile methodologies, e.g. Extreme Programming (XP) [4] and Scrum [77] in the 1990s and the creation of the Agile 'Software Development' Manifesto in 2001 [35]. The Manifesto defines the most important values and principles for "agile" software methods, according to which agile practices should be lived, designed, and applied. Besides the mentioned benefits of flexibility and adaptability, applying agile methodologies can lead to many more, such as faster delivery [57], higher quality of deliverables [76, 64], increasing customer satisfaction [57, 25], as well as the improvement of team collaboration and communication [57]. This has convinced many companies to change their way of working to applying agile methodologies on small-scale software development projects. The shift from traditional to agile methodologies is also reflected in the latest State of Agile Report [19], with 80% of the surveyed organizations stating that they use agile approaches, while only 24% use traditional approaches in the form of the Waterfall approach. Originally, agile methods were designed for self-organized, small, and co-located teams working on projects with constantly changing and unforeseen requirements [20, 100]. But the success on a small scale inspired and motivated many organizations to apply agile methodologies in larger contexts in hopes of benefiting from these improvements [20, 19]. Although using agile methods brings many benefits, their adoption can be challenging. It requires more than just adopting certain tools and practices, but rather a comprehensive mindset shift [54]. Cultivating this mindset and changing the organizational culture requires a significant investment of time and effort [92]. If this effort is neglected, the risk increases that the organization will revert to old, less

effective development methods and thus not fully realize the benefits of an agile organization [92]. Beyond the challenges present in adopting agile methods on a small scale, there are additional challenges when applying agile practices on a large scale as the complexity of a project increases [20, 62]. Among others, the coordination in multi-team environments [20, 91], dependencies to other existing environments [20, 91], and maintaining the overview of a project [85] are major challenges associated with large-scale development making the adoption of agile practices at scale difficult. To overcome these challenges, there are several approaches, such as the introduction of metrics, which can help to better control and keep track of the project process [21]. To meet customer requirements and deliver project outcomes within the planned duration and costs, accurate predictions such as effort and cost estimations can be helpful [45]. In addition to cost and effort estimation, there is also size estimation and schedule estimation supporting the success of a project and mitigating challenges when adopting agile methods on a large scale [1]. All four estimation types are significant parts of project management which in turn is one of the most important activities in software development [10]. In the context of this thesis, the focus is on effort estimation in large-scale agile development. The effort estimation is based on the complexity, degree of difficulty, and scope of the item being estimated, while the process of estimating is typically performed as a group activity involving the entire team [16, 102]. For estimates to be supportive and contribute significantly to project success, the accuracy of the estimates is extremely important to meet customer needs and avoid overruns [14, 28, 45]. Whereby inaccurate estimates lead to the opposite outcome, namely project failure [28]. However, in agile environments, whether on a small scale or large scale, the most commonly used techniques to estimate the effort are non-algorithmic techniques. Planning poker and expert judgment are the most common examples of non-algorithmic techniques, where the estimates are based on the experience, knowledge, skills, and intuition of experts [12, 80]. Due to factors such as human bias, these types of estimation methods are often error-prone [88] while making accurate predictions is generally difficult. Therefore the activity of effort estimation is considered one of the biggest challenges in agile software development, as well as the appropriate integration of the effort estimation process [28]. In addition, factors such as the complexity of agile software development, influencing factors, and constantly changing requirements pose further challenges in making accurate estimates [15, 47, 66, 73]. In large-scale agile development, additional challenges arise due to scaling factors, e.g. managing the coordination and dependencies of multiple teams [9, 96], as well as cost drivers which are not relevant in agile co-located environments but have to be considered, e.g. temporal, geographic and cultural factors [6]. In the context of effort estimation in agile development, there is numerous research on small scale but very little on large-scale agile development [95], particularly regarding the estimation process and the challenges that arise during it often triggering inaccurate estimates [95]. This bachelor's thesis aims to partially fill this research gap by conducting a case study to examine the general estimation process in a scaled agile environment and identify the challenges that arise during the effort estimation process. Furthermore, within the scope of this thesis, propositions are derived that can be applied to mitigate these challenges. To this end, three research questions were defined, which are described in the next section.

1.2. Research Questions

To fill the research gap outlined in Section 1.1, we formulate three research questions (RQs). These RQs are answered with this research project and also form the structure of this bachelor's thesis.

> **Research Question 1:** How is effort estimation conducted in the case organization?

The first research question aims to describe the current situation in effort estimation in large-scale agile development in practice. To answer this question, a case study was conducted on a large project at a major German network and infrastructure company, in which also a large German software company and a German subsidiary of an American consulting firm are involved. The objective of this research question is to investigate the process of effort estimation and the approach currently used at the case organization. Further, based on the results, a process model for the effort estimation process at the case organization is designed.

> **Research Question 2:** What are challenges in effort estimation in scaling agile environments?

The second research question aims to identify the challenges that are faced in practice when estimating the effort in large-scale agile environments. To answer this research question, the collected data from the case study is used. Subsequently, the identified challenges are evaluated by the practitioners to ensure practical relevance.

> **Research Question 3:** How can these challenges in effort estimation in scaling agile environments be addressed?

The third question aims to investigate how the identified challenges from RQ2 can be approached. For this purpose, a set of mitigation propositions are derived both from the existing literature on this topic and the conducted case study. To add value and ensure practical relevance, the derived mitigation propositions are evaluated by the practitioners.

1.3. Research Methodology

To answer the research questions described in Section 1.2, this bachelor's thesis follows the Action Design Research (ADR) approach by Sein et al. [81], which is visualized in Figure 1.1. The ADR is derived from the Design Science Research (DSR) approach [33], with the difference that the process of developing the artifact and its evaluation is not done in two separate steps but in a continuous iterative process where the evaluation contributes to refining the artifact [81]. Therefore, the goal of ADR can be described as "generating prescriptive design knowledge through building and evaluating ensemble artifacts in an organizational setting" [81]. To achieve this goal and develop an artifact generating prescriptive design knowledge,

Sein et al. [81] define the following four phases: *Problem Formulation*, *Building, Intervention, and Evaluation*, *Reflection and Learning*, and *Formalization of Learning*. The application of all four phases results in one complete ADR cycle. The allocation of the respective chapters of this thesis to the four stages of Sein et al. [81] are visualized in Figure 1.2.

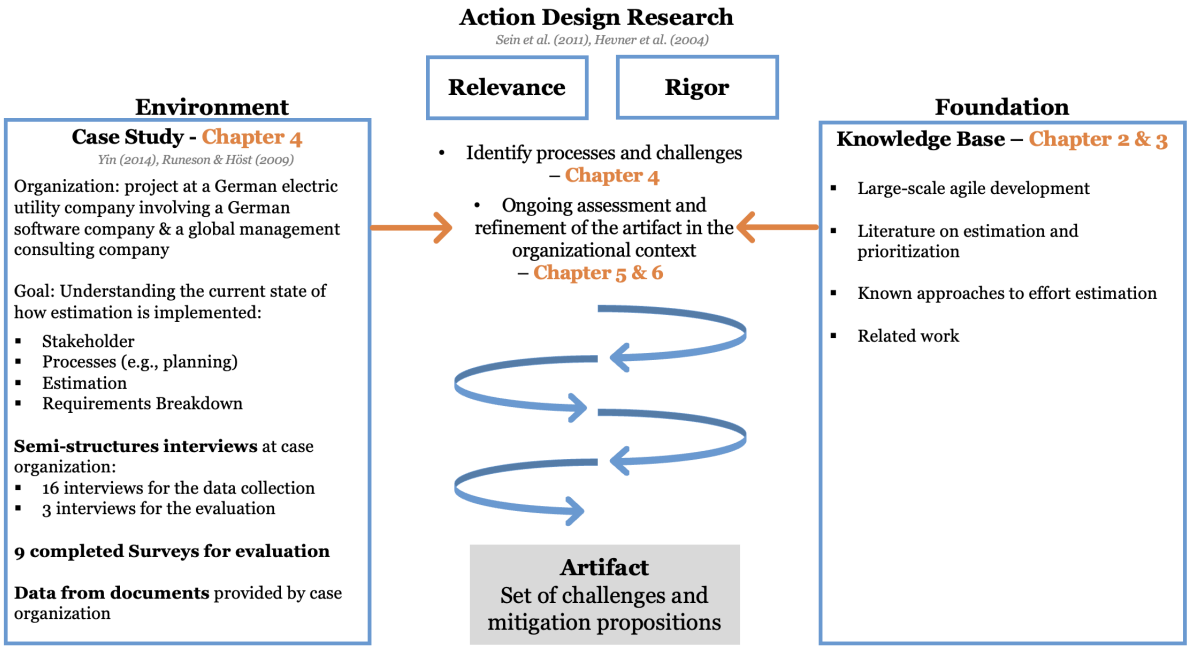

Figure 1.1.: Visualization of the Research Approach of the Thesis

Stage 1: Problem Formulation

In the first stage, the Problem Formulation, the research foundation is identified based on existing research and practical challenges in organizations [81]. Building on this foundation, the research objective is formulated. Due to the lack of research in the context of challenges and mitigation propositions for effort estimation in large-scale agile development, the problem formulation is identified in the form of a case study, which is carried out in Chapter 4 and is addressed through the artifact. The focus is particularly on understanding and investigating the effort estimation process and identifying challenges in this context. The case study follows the guidelines of Runeson and Höst [71], which are explained in Section 4.1.

Stage 2: Building, Intervention, and Evaluation

In the second stage, the Building, Intervention, and Evaluation, propositions in the context of effort estimation in scaled agile environments are identified and formulated, which can be used to mitigate the challenges identified in the problem formulation. The development of

the "Alpha Version" (Figure 1.2) of the artifact is discussed in the last Section of Chapter 4 and Chapter 5, while the first part of Chapter 6 describes the evaluation of the "Alpha Version".

Stage 3: Reflection and Learning

In the third stage, the Reflection and Learning, the qualitative feedback and improvement suggestions from the practitioners are incorporated into the "Alpha Version", resulting in a "Beta Version" of the artifact (Figure 1.2). This stage is explained in the last Section of Chapter 6.

Stage 4: Formalization of Learning

The fourth stage, the Formalization of Learning, can be assigned to the insights generated in the context of effort estimation in large-scale agile development with a special focus on challenges and mitigation propositions as part of this thesis.

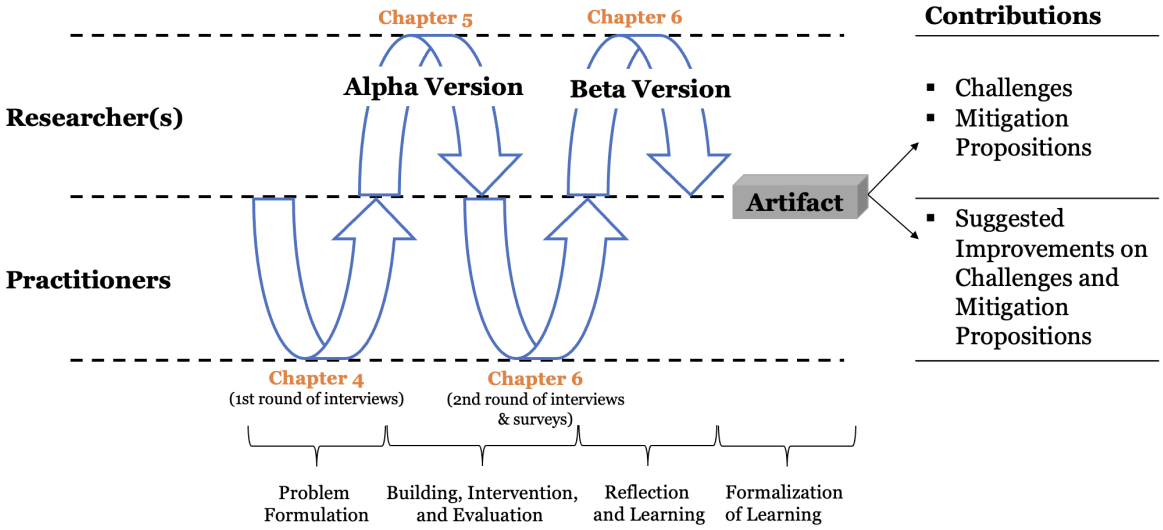

Figure 1.2.: Visualization of the Chapters of this Thesis allocated to the Action Design Research by Sein et al. [81]

Overall, one complete cycle of the ADR stages as described by Sein et al. [81] is performed throughout this thesis (Figure 1.2). The balance of relevance and rigor is of central importance in the context of DSR [33]. By reviewing, utilizing, and integrating existing literature, the research in this thesis is ensured to be rigorous. On the other hand, by conducting a case study in a real organizational setting for the problem formulation, the relevance of this research is ensured. How this balance is achieved in this thesis is visualized in Figure 1.1. The

bachelor's thesis is structured as follows. Chapter 2 explains the scientific foundations and important terminology on which this work is based on. In Chapter 3, related existing research in the context of challenges and mitigation propositions for effort estimation in large-scale agile development is investigated and discussed. Chapter 4 describes the approach of the case study, as well as the case study itself, which was conducted in a large project within a German electric utility company involving a German software company and a German subsidiary of an American consulting firm. Chapter 5 discusses the identified propositions that mitigate the challenges identified in Chapter 4, which together form the artifact of this thesis. Chapter 6 investigates the evaluation of the artifact and describes the adjustments made based on the experts' improvement suggestions. The main findings and limitations of this thesis are summarized in Chapter 7. The thesis concludes with Chapter 8, in which the research questions are answered in summary, and a brief outlook is outlined for potential future research.

2. Foundation

For a better understanding, this chapter explains the terms and concepts relevant to this thesis. In the following, Section 2.1 explains the development and fundamental concepts of agile software development, followed by an explanation of the essential aspects of the agile framework Scrum. Section 2.2 discusses necessary topics related to large-scale agile software development, followed by an overview of agile approaches in standard software. Subsequently, Section 2.3 explains the fundamental components of effort estimation.

2.1. Agile Development

The following will describe the development up to the current state, as well as the advantages and characteristics of agile methodologies. In order to clarify the understanding of agile practices, the most commonly used methodology, the Scrum Framework, will be briefly discussed.

2.1.1. Movement towards Agile

Nowadays, the ability to be flexible and adaptable has become essential for success [60, 69, 98, 19] and remain competitive [69, 98] due to the fast-paced and rapidly-changing nature of the contemporary business environment. This is especially true in the field of software development, which is characterized by unstable customer needs and rapid technological progress. To respond to these changes and remain competitive, many companies are changing their approach to their work practices moving from using traditional methodologies, such as the Waterfall approach, towards agile practices [19, 65]. According to the latest report on the agile state [19], 80% of the surveyed organizations stated that they use agile approaches in comparison to 24% using the Waterfall approach [19]. The reason for this trend is not only due to the limitations of traditional methodologies in terms of flexibility and adaptability [34] but also because of the many benefits that come with successfully implementing agile practices, such as faster and higher quality delivery [57, 76] and increased customer satisfaction [25, 57, 64]. The agile movement emerged in the early 90s introducing agile methodologies and leading to various agile approaches being developed and adapted by organizations to this day [31, 64]. The most commonly used methodologies among organizations are Scrum, Kanban, which has seen an explosive increase in usage in the last two years, and ScrumBan, a hybrid approach of Scrum and Kanban [19]. Further, the results from the latest State of Agile Report show that nowadays, agile approaches are not only applied as originally intended in the environment of software development but also company-wide due to digital transformation [19]. The fact that agile methodologies, such as Scrum, are being applied beyond their original

scope is also supported by Schwaber and Sutherland in their latest version of the Scrum Guide [78].

An important milestone in the history of the Agile movement was the creation of the "Manifesto for Agile Software Development," also known as the Agile Manifesto. In 2001, seventeen people came together, including representatives of previously established "agile" methodologies as well as others who sympathized with the need for an alternative to, at the time, conventional software development processes, which were documentation-driven and heavyweight. This resulted in the "Agile Manifesto," which was the first time the word "agile" was mentioned in the context of software development. It outlines four key values and twelve principles (Appendix A) from which the characteristics of agile practices in terms of software development can be derived, as shown in Table 2.1.

Characteristic	**Associated Key Value and Principle**
Emphasis on collaboration and communication	V1, V3, P4, and P6
Customer-focused approach	V1, P1, and P2
Flexibility and adaptability	V2, V4, and P2
Empowered individuals and teams	V1, P5, and P11
Iterative and incremental development	P3 and P7
Continuous improvement	P12

Table 2.1.: Characteristics of agile approaches and the associated key values and principles (see Appendix A)

In summary, it can be said that the agile approach differs from traditional software development approaches in that development does not revolve around a defined process but rather a short, iterative life cycle that emphasizes continuous learning and constant adaptation to changes and unforeseen [24, 99, 100]. This is also reflected in the key values and principles in the "Agile Manifesto" [35]. Despite all the benefits of applying agile methods, the adoption can be challenging as agile methods require more than just adopting specific tools and practices but rather necessitate a comprehensive mindset [92]. In fact, adopting agile methods often involve changes in the entire culture of the organization [13]. Cultivating a mindset and shifting the organization's culture requires a significant investment of both time and effort [42]. By neglecting this effort, the risk increases that the organization reverts to old, less effective ways of development and prevents them from fully capitalizing on the advantages of being agile [42].

2.1.2. The Scrum Framework

To strengthen the understanding of agile practices, this subsection introduces the most frequently used agile methodology Scrum which is especially relevant to follow the findings presented in Chapter 4. Schwaber and Sutherland, two of the seventeen authors of the "Agile Manifesto," released the first edition of the "Scrum Guide" in 2010. With the guide, they explain the adoption of Scrum and its main components to bring the world closer to

understanding Scrum [78]. Further, they emphasize that patterns, processes, and insights can be discovered, applied, and developed when applying Scrum. The description of these goes beyond the purpose of the Scrum Guide as they are context-dependent and can vary greatly depending on the use of Scrum [78]. Nevertheless, in the following, the Scrum methodology is described in accordance with the latest Scrum Guide by Schwaber and Sutherland [78] and visualized in Figure 2.1.

The Scrum Theory and Values

The Scrum theory is based on empiricism and lean thinking, which means that knowledge is gained from experience (empiricism), decisions are made based on observations (empiricism), and the focus should be on what is essential (lean thinking). To optimize predictability and risk control, Scrum follows an interactive and incremental approach. Collaboration between all involved parties and the living of the five Scrum values - commitment, focus, openness, respect, and courage - is essential for successful implementation. The application of Scrum is based on three empirical pillars: transparency, inspection, and adaptation. Transparency is important to make decisions based on the three formal artifacts (Product Backlog, Sprint Backlog, and Increment; explained below) and forms the basis for inspection. To detect unwanted deviations, inspecting the artifacts and their progress towards the goals is necessary, with the expectation that the Scrum Team will learn something new at each inspection. Adaptation is important to be able to react quickly to problems through adoption. To make rapid adoptions, Scrum provides four events within an overarching event, the Sprint. The involved parties must be sufficiently empowered to make decisions and manage themselves.

The Scrum Team

The Scrum Team is a small, interdisciplinary, and self-organized team consisting of a Scrum Master (SM), a Product Owner (PO), and developers with a typical size of 10 or fewer people. The team is understood as a unit with no hierarchy focusing together on one goal, the Sprint or the Product Goal. There is the possibility that multiple Scrum Teams pursue the same Product Goal. The areas of responsibility of the individual roles vary. The PO is essentially responsible for Product Backlog management. This includes tasks such as creating the Product Goal and entries for the Product Backlog, clearly communicating them to the Scrum Team, and ensuring transparency of the Product Backlog. Thus, if a team member wants to change anything in the Product Backlog, they must try to convince the PO. The developers of a team are responsible for creating the Sprint Backlog and implementing the entries in the Sprint Backlog. And the SM takes a supportive role in introducing and implementing the Scrum theory and practices at both the team and organizational levels. At the team level, the SM supports his Scrum Team in coaching interdisciplinary collaboration and self-management and removing obstacles. However, the SM is also responsible for ensuring that the Scrum Team lives up to Scrum practices, including holding the events. The SM supports the PO at the Product Backlog management level in facilitating the definition of the Product Goal and

promoting collaboration with stakeholders. At the organizational level, the SM is responsible for implementing Scrum, helping with comprehension questions, and removing barriers between stakeholders and teams. All in all, the entire Scrum Team "is responsible for all product-related activities [...] [and] is accountable for creating a valuable, useful Increment every Sprint" [78].

The Scrum Events

There are five Scrum events designed to enable transparency and to inspect and adapt the Scrum artifacts (Product Backlog, Sprint Backlog, and Increment; explained below) in order to achieve the Product Goal. The main event is the Sprint, within which the other four events take place, and during which "ideas are turned into value" [78]. A Sprint is of a fixed length, usually one month, and allows for predictability by inspecting and potentially adapting progress. Predictions can be made, for example, regarding the use of metrics, but it should not be overlooked that what will happen in the future is often unknown [78]. This is where the importance of empiricism comes in, with the approach that knowledge is gained from experience and decisions are made based on observations, as mentioned above. The scope and goal of a Sprint are the results of the Sprint Planning event, which is held before the Sprint with the entire Scrum Team involved. Entries from the Product Backlog relevant for the next Sprint are selected, their scope and completion discussed and then summarized in a Sprint Backlog. In addition to the Sprint Planning event, there is the Daily Scrum event, which takes place daily and creates a plan for the upcoming work for the next workday. This event also allows progress toward the Sprint Goal to be inspected and potentially adapted. The fourth event is the Sprint Review, which aims to present the results of the Sprint to the most important stakeholders, discuss progress towards the Product Goal, and plan the next steps, which can lead to an adoption of the Product Backlog. The involvement of stakeholders allows for quick responses to changes in customer requirements, expectations, and needs. In the final event, the Sprint Retrospective, the Scrum Team collectively reviews what goals were achieved in the Sprint, what problems arose, how these were solved (or not), and what can be refined to improve quality and effectiveness in the future. The length of each event depends on the length of the Sprint, but Schwaber and Sutherland provide guidelines based on a 4-week Sprint. Through this rhythm of events, team communication, and collaboration are strengthened, as well as flexibility and adaptability to respond to changes are given.

The Scrum Artifacts

The Scrum artifacts representing work or value are the Product Backlog, the Sprint Backlog, and the Increment, each including a commitment that provides information to improve transparency and make progress measurable. The Product Backlog is an emergent, ordered list of things needed to achieve the Product Goal, which is the commitment. During refinement activities, the entries in the Product Backlog are broken down into smaller elements and defined in more detail, forming the basis for the Sprint Backlog, which the Scrum Team creates during the Sprint Planning event. Refinement is a continuous activity in which the entries

are supplemented with further details such as description, order, and size, more specifically, estimation [79], which is the responsibility of the developers. The entire refinement process can be influenced by the PO, who can assist the team in understanding each entry. The corresponding commitment, the Product Goal, is a long-term goal that describes the future state of the product and is therefore located in the Product Backlog. As mentioned, the entries in the Product Backlog serve as the basis for the Sprint Backlog, which is "composed of the Sprint Goal (why), the set of Product Backlog items selected for the Sprint (what), as well as an actionable plan for delivering the Increment (how)" [78]. The Sprint Backlog serves as a plan by and for developers to achieve the Sprint Goal and is updated throughout the Sprint as more is learned. The Sprint Goal is part of the Sprint Backlog, created by the team during Sprint Planning, and creates coherence and focus for working together as a team.

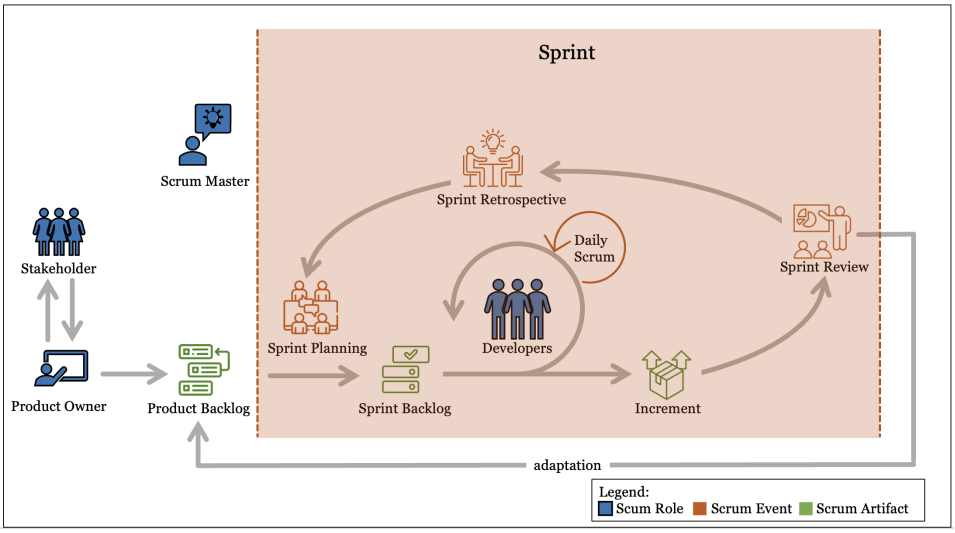

Figure 2.1.: The Scrum Framework

2.2. Large-Scale Agile Development

The following will describe the taxonomy and the development up to the current state of large-scale agile development, as well as the scaling factors and related challenges when applying agile methodologies on a large scale. Subsequently, a brief overview of agile approaches used in standard software is given.

2.2.1. Definition of Large-Scale Agile Development

As applying agile approaches on a large scale is becoming increasingly popular both in practice and academia [24], there is a necessity of having clarity on what large-scale agile development is [21]. The current research on large-scale agile development shows that there

are different understandings of the term "large-scale" and that, to date, no clear definition is applied [22]. The term "large-scale" in the context of agile development ranges from the number of employees or agile teams involved in the project to the duration or costs of a project [21]. Dingsøyr et al. [21] discuss the drawbacks of using the term in dimensions like the project costs, the number of requirements, or lines of code. Since these vary by project, Dingsøyr et al. [21] argue that using the term in this context is inappropriate. They, therefore, suggest using the term "large-scale" based on the number of teams involved, which is related to the coordination overhead. According to Dingsøyr et al. [21], large-scale development includes between 2-9 teams since, for this, a new coordination forum such as the Scrum of Scrum forum is required. Projects including ten or more teams are classified as very large-scale according to the taxonomy, as multiple coordination forums are required. In this thesis, the taxonomy proposed by Dingsøyr et al. [21] is used for (very) large-scale development. To be more specific, this means that in this thesis, large-scale agile development is defined as multiple teams working together on one project using agile methodologies for implementation.

2.2.2. Large-Scale Agile Development

As mentioned above, applying agile methodologies at a small scale can lead to significant improvements in, e.g., customer satisfaction and delivery [57]. Inspired by this, applying agile methodologies in large-scale environments is becoming increasingly prevalent among contemporary companies [20]. Triggered by the growing popularity in this context, there was the need to extend, adapt, and create agile methodologies for large-scale settings [21], resulting in more than 20 frameworks [93] providing guidance and support for the transition and adoption of scaling agile practices. The most common frameworks used in the industry are SAFe (Scaled Agile Framework) and Scrum@Scale/Scrum of Scrums [19]. These differ depending on how scaling and coordination are achieved. SAFe is a comprehensive framework consisting of multiple levels and roles enabling scaling of agile methods in larger companies [75], and Scrum@Scale is mainly geared towards coordinating multiple Scrum teams in larger projects or companies [84] using the Scrum of Scrums approach to coordinate teams to improve communication, reduce dependencies, and avoid duplication of system components [70]. Since many companies adopted scaling agile practices in projects and throughout their whole organization [21], there has been an increasing interest in research resulting in many studies in this context throughout the last decade [92].

2.2.3. Scaling Factors and Challenges

In addition to various approaches for applying agile practices in large-scale development, Kruchten [44, 43] proposed a contextual model for the development of software-intensive systems, which serves as a guide for adoption and adaptation. The model is intended to help ensure that projects that deviate from the "agile sweet spot" but are within the "agile bitter spot" also reap the benefits of applying agile practices [43]. Table 2.2 shows the ideal conditions under which agile practices emerged and are most likely to be successful (agile

sweet spot) and the conditions outside the sweet spot that may cause difficulties (agile bitter spot).

Agile Sweet Spot	**Agile Bitter Spot**
Small team (10-15)	Large team (> 15)
Co-located team(s)	Distributed team(s)
Customer availability	No empowered customer representative
Business application	Embedded real-time systems
New development	Maintenance projects
RAD programming environment	Inefficient programming environment
Short life-cycle	Long life-cycle
Common development culture	Different development cultures

Table 2.2.: Agile Sweet and Bitter Spot by Kruchten [44, 43]

Based on the research by Kruchten [44, 43] in the context of situational agility, Ambler and Lines [2] identified six scaling factors (see Figure 2.2) that align with Kruchten's key points [43]. Factors such as global distribution, increasing organizational distribution, or increasing complexity, both in the domain and technical context, lead to the need for scaling agile practices [2].

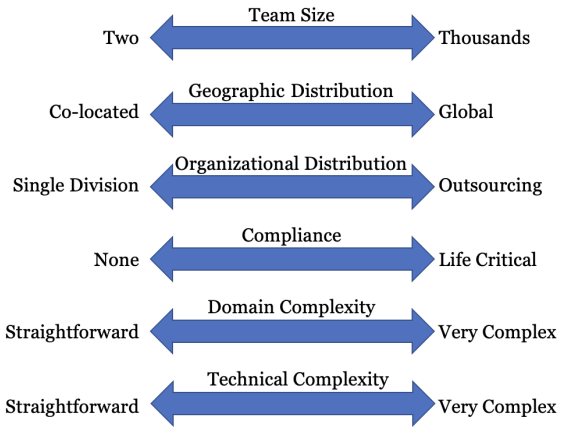

Figure 2.2.: Scaling Factors by Ambler and Lines [2]

Despite the benefits of applying agile methodologies on scale [46, 58], the implementation leads to many challenges [20, 62], especially when the project context significantly deviates from the "agile sweet" [43]. This is also reflected in research that has identified challenges in scaling agile environments, for example, due to increasing organizational scope [11, 23] or complexity [2, 11]. In this context, possible challenges could be the coordination in multi-team environments [6, 20, 91], dependencies to other existing environments [20, 91], and maintaining an overview of a project and project progress [85]. To address some of

the challenges occurring, the use of metrics could be effective. Metrics provide progress monitoring and transparency support [21] that can help control and track the project's progress. Further, to meet customer requirements and prevent project failure due to time or cost overrun, accurate predictions such as effort and cost estimations can be helpful [45]. This thesis pays particular attention to the effort estimation, which will be discussed in Section 2.3.

2.2.4. Large-Scale Agile Standard Software Implementation

Standard software implementation projects differ from software development projects, as in this case, no new software is developed, but off-the-shelf solutions from software providers are adapted to the needs of individual customers through customization and modification [59]. Implementing Enterprise Resource Planning (ERP) systems is one of the most well-known standard software implementation projects. In general, ERP systems are large-scale integrated systems that cover most of the business processes within an enterprise [59] and are implemented in customization, modification, integration, and data conversion phases to meet customer needs [59]. There are several approaches for implementing an ERP project, with traditional methodologies such as the waterfall approach still being commonly used for such projects [50, 74]. In this approach, implementation is carried out in successive phases, and the final result is delivered at the end of the project life cycle [50]. Therefore, there is a lot of time between defining requirements and the completed implementation. This can cause changes in customer requirements and result in the final implementation not meeting the customer's needs [50]. For this and other reasons, agile practices are becoming increasingly popular in ERP projects or ERP implementation projects [29, 50, 74]. However, in research ERP implementation using agile methodologies is a very new area, which is why opinions on applying agile methods in this context are mixed [50]. However, the little literature in this context proves that ERP projects can benefit from using agile practices, among other things, in terms of flexibility in requirements, continuous improvement, faster delivery, and customer satisfaction [50, 74]. In the context of this thesis, a case study is conducted on a project in which a standard ERP solution is implemented using agile practices, especially at a team level. Particular attention will be paid to the performance of effort estimation in the case organization and its associated challenges. Therefore Section 2.3 will cover the scientific foundation and terminologies of effort estimation.

2.2.5. Difference between Large-Scale Agile Development and Agile ERP-Implementation

As mentioned above, purely agile software development projects differ from ERP implementation projects that use agile principles. This subsection briefly discusses the two main differences between these different software projects. In (large-scale) agile software development projects, a deliverable part of the software is developed during a Sprint, which is part of the end product and released at the end of each Sprint [51]. In an ERP implementation, on the other hand, the end product already exists, which is implemented and modified in iterations but not released before the so-called go-live [51]. In implementing such ERP

systems, adjustments or new functionalities are only made if they can generate business value or are legally required. In most cases, such adjustments or developments must be approved by a design authority [51]. A second difference is that while the schedule and other resources are predetermined in (large-scale) agile software development projects, the necessary resources to implement any scope are difficult to estimate, making the scope variable in a (large-scale) agile project [51]. In agile implementation projects, on the other hand, the scope is often predetermined and prioritized, with the scope being delivered in descending order of priority, as long as project resources allow [51].

2.3. Effort Estimation in Agile Development

This section explains the critical components of effort estimation. After briefly introducing the necessity of effort estimation, there is a short overview of the three most important contextual factors around which effort estimation is performed. Then, the size metrics and cost drivers are explained, followed by the estimation techniques and accuracy metrics. Further, effort estimation is described in the context of Scrum and large-scale agile development.

2.3.1. Overview on Effort Estimation

Software estimation is not a novelty established with the introduction of agile practices [10, 102]. On the contrary, estimation is a significant part of project management and one of the essential activities in software development [10]. Software estimation can be distinguished into four types: the first is estimating the entire project's cost, the so-called cost estimation [1]. The second type is estimating the size or complexity of a project, the so-called size estimation, and the third type is estimating the time required to complete the entire project, the so-called schedule estimation [1]. The fourth and most relevant type for this thesis is estimating the resources required to complete a project and deliver a product that meets the customer needs the effort estimation [1, 28]. During the software development process, the focus is on software estimation in terms of cost and effort estimation [97], which are both essential to prevent project failure due to time or cost overrun [45]. However, the cost estimate is highly dependent on the effort estimate [8], which results in the effort estimate being crucial for the implementation of software development projects. In contrast to an agile approach, developing software following a traditional approach, such as the Waterfall approach, is carried out in a linear, sequential manner that does not allow overlapping processes to work on different phases simultaneously [102]. The different approaches to software development result in other effort estimation processes [16]. For example, in the Waterfall approach, the effort estimation is performed by the manager who estimates the workload capacity of a team member by estimating the time it takes to complete a task using, e.g., regression-based models and assigns the task based on the team member's total available time [102]. Whereas in agile software development, the focus of implementing tasks is on iterations in which the entire team performs the effort estimation, and tasks are estimated based on the complexity and degree of difficulty [102, 16]. Due to the completely different approaches of software

development and also effort estimation [16], it was necessary to establish new techniques or adapt existing ones to the agile environment [102], resulting in a variety of techniques that can be used for effort estimation in agile environments [83]. Ziauddin et al. [102] even claim that using estimation techniques developed for traditional sequential software development leads to inaccurate estimates. These, in turn, may lead to overruns in terms of time or cost [55, 56] often resulting from underestimations [28, 88]. Since the success or failure of a project depends on the accuracy of the estimates [28, 40, 56], it is crucial and relevant to make estimates as accurate as possible and to improve the estimation performance continuously. However, making predictions is always a challenge, which leads to the fact that accurate estimation is one of the biggest challenges in software development [28]. Further, integrating the effort estimation process into an agile environment poses another challenge [28]. This is because, in addition to the complexity of agile software development and many cost factors affecting effort estimation [47, 66], there are constantly changing requirements during the development process [15, 66, 73]. Changing requirements often leads to incremental adjustments of the estimates [73], which can be very difficult, especially when this results in moving requirements to another iteration, e.g., Sprint.

2.3.2. Estimation Context

This section discusses the three most important aspects of the context in which effort estimation takes place: the planning level, the estimated activities, and estimation entities [94]. As mentioned in Section 2.1, agile software development is characterized by the incremental development of software in small iterations, which are referred to as Sprints in the Scrum Framework [78, 97, 94]. The subsequent iterations are based on customer feedback and the learning progress from the previous iterations [97, 94]. Accordingly, the planning of iterations and estimation of the effort are done progressively [97, 94] and iteratively. The planning is mainly performed at three levels: release planning, iteration planning, which is referred to as the Sprint Planning Event in Scrum, and daily planning, which is referred to as the Daily Scrum Event in Scrum [96, 94]. The effort estimation process is primarily performed at the iteration planning and release planning level and less on the daily planning level [28, 94]. A software project process consists of several activities required to successfully complete a project, including requirements analysis, design, implementation, and testing of implementations [96]. No standard defines for which of the activities an effort estimate should be performed, which causes it to change from project to project [96]. In most cases, however, the effort is estimated for the interrelated activities' implementation and testing [96]. In contrast, efforts for activities such as requirements analysis, design, and maintenance are less frequently estimated [96]. Especially in large-scale projects involving multiple teams, it is important to clearly define what activities should be estimated to ensure consistency across the project. The corresponding estimation entities are associated with the estimated activities, with requirements being the basic input to the estimation [94]. There are various ways and levels of detail in which requirements can be specified. For example, requirements can be formulated in so-called user stories, which are then estimated [94]. Further, user stories can be divided into smaller tasks, which are then estimated [94]. The estimation entity usually

depends on the planning level at which the estimation is done, the agile approach used, and the team's preference [94].

2.3.3. Estimation Predictors

This section introduces the two main effort predictors in the context of effort estimation, namely size metrics and cost drivers, by briefly giving an overview of each effort predictor.

Size Metrics

Many different size metrics are used for effort estimation [15]. In applying traditional methodologies, Function Points or Lines of Code (LOC) are often used, with the former being the most commonly used size metric. It is not uncommon for size metrics such as Function Points or LOC to be used in agile development environments. Still, when they are, it is often only combined with typical agile units such as Story Points [18, 28, 97]. Nevertheless, using LOC is often criticized as it is based on the developer's efficiency and is therefore poorly generalized [15]. However, in the context of effort estimation in agile development, the use of relative estimation metrics such as Story Points or T-shirt size is recommended, as they allow for comparison between tasks or teams [16]. Relative estimation units also simplify the planning and create a shared understanding of the work [16]. Unlike absolute estimations such as person-hours, relative estimations are independent of individual work speeds, resulting in the effort remaining the same for each team member [16]. However, some researchers argue that using relative estimations often causes difficulties for teams as they often have no reference to the metrics and find them hard to understand, contributing to inaccurate estimations [18, 30, 52]. This can be attributed to the fact that there is no standard for the definition of, for example, Story Points, but instead, it can change from project to project [16, 52]. However, the most commonly used metric in practice is the unit of Story Points [18, 28, 30]. It is difficult to find a clear definition for Story Points [16, 30]. In most cases, however, Story Points are used in the context of User Stories. Each User Story is assigned Story Points based on the effort, complexity, and inherent risk involved in development [36]. Since it is a relative estimation metric, Story Points are assigned in such a way that, for example, a Story estimated at five Story Points requires five times as much effort as a Story estimated at one Story Point [15]. Again, there is no prescribed standard for the size metric in agile development, but in general, the decision of which size metric to use for effort estimation should be carefully considered based on project size and the team's estimation experience [52]. Additionally, the size metric should be clearly defined before the project begins, and it should be ensured that everyone understands the unit feeling comfortable estimating with the selected size metric [30, 102].

Cost Drivers

In addition to the size metric, one of the most important estimations predictors for the required development effort, cost drivers also play a significant role in determining effort

estimates. Considering these factors can help improve the accuracy of estimates [88]. Research shows that the most commonly identified cost drivers in effort estimation are team-related factors [28, 94]. In research, team-related factors refer to factors such as the experience and skills of the team, as well as the team's familiarity with the project, the team size, the team velocity, communication, working hours/days, and the developer's availability [18, 28, 88]. Further, cost drivers in the context of user stories, the project, and technical factors also play a role in effort estimation and should not be neglected [18, 28, 94]. In large-scale agile environments, similar cost drivers impact effort estimation with additional factors that matter due to global distribution in terms of global barriers [10]. This includes temporal, geographical, and cultural factors, e.g., language differences [10]. To improve the accuracy of the estimates, it is recommended to consider any relevant cost driver when estimating effort [88, 96].

2.3.4. Effort Estimation

This section presents frequently used techniques in agile development to estimate effort. Further, the importance of accurate estimation is discussed in detail, and the three most common accuracy metrics which can be used to measure the accuracy of an estimate are briefly explained.

Estimation Techniques

Many techniques have been proposed to estimate the effort of a project [15, 28]. Essentially, these methods can be classified into two categories. The first category includes algorithmic-based techniques, where the estimation process is based on equation and mathematics [28]. This includes data-driven methods such as machine learning, natural network, regression, and functional size measurement. The second category comprises non-algorithmic techniques, where the estimation process is based on analogy and deduction [28]. This often includes methods based on expert opinions, such as Planning Poker, expert judgment, source lines of code (SLOC), and analogy. However, using SLOC is based on the developer's efficiency and therefore has found little acceptance in the agile community despite its simplicity [15]. The advantage of non-algorithmic techniques is that they are not as time-consuming as those based on algorithms [16, 15] and are frequently used in the implementation of planning poker and expert judgment [12, 80]. However, it is often difficult to find suitable experts [16, 15], which among other things, leads to lower accuracy compared to estimates based on data algorithms [28]. Similar to size metrics, there is no standard for estimation techniques. Instead, applying a technique varies from project to project [39, 52] and therefore should be considered based on project size and team experience [52]. In practice, a combination of several techniques is often used to improve the accuracy of the estimates [28, 97]. Nevertheless, the most widely used non-algorithmic techniques in the context of agile development will be briefly discussed in the following:

Expert Judgement is a technique in which experts rely on their intuition and gut feeling to

estimate entities, e.g., a (User) Story [16]. In agile projects, entities often include functionalities that involve different skills. In this case, it is difficult to find a suitable expert who is able to estimate across all disciplines, which is why a combination of experts in a team would be desirable [16].

Analogy is an approach in which the entity is compared to other entities of the same type, e.g., a (User) Story. It should be ensured that each new entity is compared to a selection of already estimated entities [16]. This refers to a triangulation process, where the potential estimate of the new entity is compared to one that was estimated slightly smaller and another that was estimated slightly larger [16]. The estimate is adopted for the new entity if the potential estimate is within a reasonable size range.

Disaggregation is an approach in which a very large entity, e.g., a requirement, is divided into smaller and more easily estimable parts [16]. This ensures that all entities that should be estimated during a planning event are of roughly the same size and can therefore be compared more easily: "Asking 'Is this story fifty times as hard as that story' is a very different question than 'Is this story about one-and-a-half times that one?'" [16]. However, it is not recommended to disaggregate too far, as there is an increased risk of forgetting something [16]. Disaggregation is less of an estimation method and more of an approach often used to simplify estimating large requirements. In the case study (described in Chapter 4), disaggregation is referred to as break down process.

Planning Poker is a technique that combines all three approaches from above, resulting in a fast but reliable estimation [16]. The entire team participates in the estimation process, which the Product Owner usually moderates. At the beginning of the planning event, each team member is given a deck of cards with numbers reflecting the estimate [16]. There are no strict rules for the numbers, but the Fibonacci sequence is frequently used. The Product Owner reads the description of the entity to be estimated, e.g., a (User) Story, and then answers the estimators' questions about that entity to clarify any uncertainties before the estimation process begins [16]. Then the first round of the estimation process begins, and each estimator secretly chooses the card representing their estimate. After everyone has made their estimate, all cards are revealed simultaneously [16]. It is often observed that the estimates differ significantly, resulting in the estimator with the lowest number and the one with the largest number sharing their thoughts with the others and justifying their estimate [16]. After that, the team has a 2-3 minute discussion, and the second round of the estimation process begins. This is done in the same way as the first round, with the result that the estimates often converge after the second round [16]. If this is not the case, the process is repeated exactly as before in the optimal scenario until the estimates converge. This can also be shortened if the group can agree on an estimate together [16].

Accuracy Metrics

As mentioned above, the most commonly used estimation techniques are non-algorithmic heuristic techniques, which often rely on experts' intuition and gut feelings. Such estimates are often error-prone due to, e.g., human bias [88]. In the worst case, this can lead to project failure regarding delivery and costs. While inaccurate estimates can lead to negative effects [28], accurate estimates can contribute to the success of the project [14, 28, 45] and facilitate companies to minimize delays and improve customer satisfaction. Further, with accurate estimates, companies can efficiently allocate resources, reduce costs, and optimize delivery [32, 28, 38]. Therefore, the accuracy of estimates is of particular relevance and importance [61, 45] leading to the need for measurements. In research, several methods have been proposed to measure the accuracy of estimates, such as Magnitude of Relative Error (MRE), Prediction Level PRED(x), Mean Absolute Error (MAE), or Balanced Relative Error (BRE) [28, 96, 94]. These accuracy metrics are rarely used in the industry, instead, it is sufficient to compare the estimated effort to the actual effort [28]. However, in this context, many research papers have calculated the accuracy of the most commonly used estimation techniques using such accuracy metrics. For the sake of completeness, the accuracy metrics identified as the most commonly used in research will be briefly explained below [28, 67]:

Magnitude of Relative Error (MRE) is a measurement that calculates how far the estimated effort deviates from the actual effort. The result of the formula represents the relative error in the estimate, which can, in turn, indicate the accuracy of an estimate [28, 67, 94]. The MRE is defined as [28, 37, 67]:

$$MRE_i = \frac{|ActualEffort_i - EstimatedEffort_i|}{ActualEffort_i}$$

The absolute value of the difference between the actual effort and the estimated effort (of a project *i*) is divided by the actual effort (of a project *i*), resulting in a unit-less value, the MRE (of a project *i*) [28, 67]. The smaller the MRE value, the more accurate the estimate. Accuracy metrics based on MRE are helpful because they are easy to interpret and yet meaningful [37]. Two such accuracy metrics based on MRE are the Mean Magnitude of Relative Error (MMRE) and the Percentage Relative Error Deviation within x (PRED(x)) [67], which will be discussed subsequently.

Mean Magnitude of Relative Error (MMRE) is the most commonly used measurement to assess the accuracy of an estimate [28, 41, 67]. The MRRE is defined as [41, 67]:

$$MRRE = \frac{1}{N}\sum_{i=1}^{N} MRE_i$$

As previously mentioned, the MMRE is based on the MRE and is defined as the average of the sum of all MRE values measured over N data points [41, 67]. The result of the equation indicates the mean relative error in the prediction. For example, a result of MMRE = 0.1

indicates a mean relative error in prediction of 10% [37], which would be within the acceptable level of MMRE-value 0.25 for the accuracy of an estimation proposed by Conte et al. [17]. Again, the smaller the value, the more accurate the estimate. However, MMRE is strongly affected by a few high MRE values leading to research often referring to the median relative error (MdMRE) [37].

Percentage Relative Error Deviation within x (PRED(x)) is a measurement that is often used as a complementary criterion to MMRE [28] calculating the proportion of MRE-values that fall within a selected threshold (x) to the total number of N projects [28]. In most research studies, x is set to 0.25 [28, 37] indicating that the PRED(0.25) result reflects the percentage of MRE-values that fall within the threshold of 0.25 respectively 25%. For example, the PRED(0.25) = 80 indicates that the relative error equals 25% or less for 80% of all MRE-values taken into account.

It should not be overlooked that accuracy measures such as MMRE or PRED(x) are not always best suited for accurate estimation [82]. Therefore, Shepperd and MacDonell [82] suggest using measures based on residuals. Fernández-Diego et al. [28] conducted a systematic literature review examining studies focused on effort estimation. Among other things, they summarized the results of the accuracy of the most common estimation techniques considering MMRE and PRED(x) (see Table 2.2). The PRED(x) metric results suggest that using data-driven techniques to estimate the effort results in a more accurate estimate than using non-algorithmic techniques like Planning Poker [28]. However, in practice, less time-consuming non-algorithmic techniques in the form of expert judgment or Planning Poker are preferred [12, 80].

Estimation Method	Accuracy Metric	Mean	Median	Range
Non-algorithmic:				
Planning Poker	MMRE	0.41	0.39	0.07, 1.07
	PRED(x)	62.80	73.00	33.00, 83.00
Expert Judgement	MMRE	0.22	0.22	0.12, 0.33
Algorithmic:				
Machine Learning	MMRE	0.58	0.43	0.06, 2.04
	PRED(x)	73.39	80.95	38.10, 100.00
Neural Network	MMRE	0.23	0.12	0.03, 1.58
	PRED(x)	88.83	94.87	57.14, 100.00
Function Size Measurement	MMRE	0.41	0.28	0.06, 1.20
	PRED(x)	86.21	80.63	78.00, 100.00

Table 2.3.: Accuracy summary statistics by Fernandez-Diego et al. [28]

2.3.5. Effort Estimation in Scrum

After outlining the most important aspects of effort estimation, the following will briefly discuss the process of effort estimation in Scrum according to the Scrum Framework by Schwaber and Sutherland [79]. Schwaber and Sutherland [79] provide little guidance on

performing effort estimation. Still, they emphasize that a Product Backlog item should have an estimation, in addition to the attributes description and order. Primarily, the Product Backlog items should be prioritized or ordered by the Product Owner and subsequently estimated by the team. The Product Owner only assists in the estimation process in case of comprehension questions. In the Sprint Planning Meeting, the team predicts which items it can deliver in the upcoming Sprint, selects them, and breaks them down into smaller working items. During this process, the team is responsible for estimating the effort of each working item before they are moved to the Sprint Backlog. Schwaber and Sutherland [79] recommend using daily meetings to re-estimate Sprint Backlog items if necessary and keep the Sprint Backlog up to date. The Sprint Backlog can be changed during a Sprint, but care should be taken to adjust the estimates of working items accordingly and estimate newly added working items. Suppose a working item is incomplete at the end of a Sprint, which should be avoided. In that case, the item should be re-estimated and moved back to the Product Backlog, causing the already completed work to lose value [79]. Overall, the team is responsible for all estimates, and the Product Owner can support the team in the estimation process, as mentioned above [79]. The Scrum framework generally provides little guidance on how the estimation process should be performed, which techniques can be applied, and which units can be used. To this end, Schwaber and Sutherland mention in the first chapter that "specific strategies for using the Scrum framework vary and are described elsewhere" [79]. It should be noted that the aspects of effort estimation discussed in the previous sections have been identified in most cases in agile environments where the Scrum methodology was applied.

2.3.6. Effort Estimation in Large-Scale Software Development

The effort estimation process, the estimation techniques applied, as well as the size metrics used for estimating in large-scale agile development are similar to small-scale agile projects [9, 96]. More specifically, planning Poker and expert judgment methods are commonly applied for effort estimation in large-scale agile development, using Story Points or task size to estimate the effort [9]. The difference in the process is that cost drivers should be considered during estimation, which do not play a role in agile co-located environments, such as cost drivers in the context of global barriers, e.g., temporal, geographic, and cultural factors [10], as describes above. In addition, the estimation process in large-scale agile development presents challenges that are less relevant on a small scale, such as managing the coordination and dependencies between multiple teams [6, 7]. Furthermore, long-term estimates, i.e., estimates for several months, pose significant difficulties for teams in scaled projects, as the scope of requirements compared to smaller projects is immense. This can lead to teams being generally skeptical about estimates [27], resulting in inaccurate estimates and a lack of commitment. Unlike research in the context of effort estimation in agile development, there is little research on effort estimation in large-scale agile development, particularly on the process and challenges faced during effort estimation [95]. With this thesis, we attempt to fill this gap by pursuing the research objective of understanding how effort estimation is performed in scaling agile environments, identifying what challenges are encountered during the estimation process, and deriving propositions to mitigate these challenges.

3. Related Work

After outlining the theoretical foundation and explaining key concepts relevant to this thesis in the previous chapter, this chapter covers the related work. Here, publications that have dealt with effort estimation and its challenges are presented. It should be noted that this chapter only provides a snapshot of what exists in the literature.

3.1. Effort Estimation in Agile Development: Process, Challenges, and Mitigation Propositions

Mallidi and Sharma [52]

Mallidi and Sharma [52] conduct a literature review to introduce agile estimation techniques in the context of the estimation unit Story Points. Mallidi and Sharma [52] explicitly discuss the agile estimation methods Planning Poker, T-Shirt Size, Dot Voting, Bucket System, Large/Uncertain/Small, Ordering Method, and Divide Until Maximum Size or Less, which can all be used for relative units. Mallidi and Sharma [52] focus on estimation techniques used in the agile Scrum framework, in which the estimation process is typically performed in a dedicated Scrum Meeting where a responsible person, often the Product Owner, selects a requirement from the Product Backlog. The selected requirement is then discussed and estimated by the entire team estimates based on one of the described techniques. Mallidi and Sharma [52] emphasize that the entire team decides on the estimate as a group decision, resulting in no individual being responsible for an inappropriate estimate. According to Mallidi and Sharma [52], the advantage of Story Points lies in comparability. By using Story Points, the work performed by a team during a Sprint, the so-called velocity, can be compared with the velocity of other teams. The authors note that the Scrum Master decides on the appropriate estimation technique based on the project scope, the experience of the resources, more specifically, team members, and the project size. In summary, Mallidi and Sharma [52] provide details on when which estimation technique is applicable. Finally, Mallidi and Sharma [52] discuss the challenges related to estimating using Story Points and provide suggestions on mitigating them. The table below summarizes the identified challenges and proposed mitigations [52].

Challenge	Mitigation Propositions
Lack of story point estimation prediction: Teams do not know how to predict the story point from the backlog item	Considerable training for the team on how to work with Agile and do story point estimations
No standardize estimations technique: Agile Scrum estimates are relative in size and not defined any size. The definition of story point is different from project to project	Scrum Master selects appropriate estimation techniques based on project size and experience of the team
Lack of metrics like size, effort, and velocity from another project	Management has to support the team to use agile Management tools to log the activities (e.g., efforts, estimates, velocity) for further use
Overestimation of Sprint if already not done previously the same type of work: Most of the time, developers inflate the story points if the team is working on new technology	Scrum Master must moderate the team and provide the relevant information on new technology to the team, not to inflate the stories
Cultural and communication bandwidth: How the resource has understood the requirement is a primary factor for Agile delivery	Scrum Master needs to conduct daily standup meetings with the Product Owner for clearly understating of the requirement to be delivered on time
Too big projects or backlogs	Split the requirements into smaller tasks and create more small teams, working individually to deliver the common goal. In large projects create Scrum-of-Scrums/SAFe teams
Non commitment of the resources	Commitment can be addressed by creating a self-managing and self-organizing team allowing the team to be creative, innovative, and acknowledged for their expertise
Project Management Integration	Agile process is not suitable for projects requiring a plan-driven approach. Create a hybrid Agile approach to integrate Project Management aspects into plan-driven projects

Table 3.1.: Challenges and Mitigation Propositions by Mallidi and Sharma [52]

Tanveer et al. [88]

Tanveer et al. [88] conducted a case study to examine and understand the accuracy of the estimation process in an agile development environment. Therefore, Tanveer et al. [88] performed interviews with three development teams working at the German multinational software company SAP. Tanveer et al. [88] focused mainly on the potential influence of changes to an existing software artifact and factors that could contribute to effort overhead to improve existing estimation methods in terms of accuracy. They found that there was no other research at the time that had focused on the same issues. With the first research question, Tanveer et al. [88] aimed to examine the general estimation process within the agile teams, including the methods used, the reasons for estimation, the people involved, factors that influence estimation and the process, as well as the tools used during the estimation process.

They found that all participating teams conducted their estimations at the Sprint planning level using tracking and management tools such as JIRA to create a shared understanding of the tasks to be completed. With the second research question, Tanveer et al. [88] aimed to identify potential relevant factors to consider when estimating to improve the accuracy. The findings showed that factors considered relevant were the developer's experience in estimating and implementing, the developer's knowledge of backlog items, dependencies between backlog items, as well as the complexity of backlog items and their impact on parts of the system to be developed. Based on these results, Tanveer et al. [88] decided to define a third research question to investigate which tools could support the estimation process. For this, five predefined aspects were presented: influential factors, impact analysis, performing estimation, visualizing estimation, as well as expert interaction and updating estimation. The interviewees were asked to evaluate all five aspects by rating "Required," "Required with conditions," or "Counterproductive," with a focus on whether these aspects might require tool support. Almost all aspects require some form of tool support, albeit with conditions. Several interviewees rated the aspect performing estimations as counterproductive and explained that pre-set estimates could lead to the team not thinking about and actively discussing estimates themselves. Overall, the results of Tanveer et al. [88] show that considering the factors mentioned above during the process can improve the estimation accuracy. Furthermore, the estimation performance of a team and the estimation process can be improved if certain aspects, such as influencing factors or impact analysis, are supported by tools. Tanveer et al. [88] see great potential in using impact analysis to improve estimation effectiveness. Therefore, Tanveer et al. [88] developed a methodology that considers these factors and quantifies the impact of changes to estimate changes in agile software development effectively in the research work of Taveer et al. [87]. Further, the developed methodology is supplemented by guidelines for using impact analysis [86]. In a further study [89], the proposed framework [87] and an associated mock-up were evaluated by the interviewees from the initial research [88]. Overall, the practitioners rate the framework as very useful for supporting the effort estimation process and the associated mock-up as easy to handle. In addition, practitioners believe that visualizing the factors during the estimation process provides more objectivity in estimates than subjective estimation based on expert opinions [89].

Ziauddin et al. [102]

Ziauddin et al. [102] set out to develop an effort estimation model that is better suited to overcome the challenges in effort estimation in agile projects as it is based on the characteristics of the agile methodology. To measure the accuracy of the developed approach, Ziauddin et al. [102] use empirical data from 21 software projects from 6 software companies. Effort estimation can be influenced by many factors that make accurate estimation difficult. Ziauddin et al. [102] recommend focusing only on factors that can be influenced by the team and placing less emphasis on uncontrollable factors. Ziauddin et al. [102] consider the factors of size and complexity of a user story. Since the developed approach is based on user stories, Ziauddin et al. [102] choose the size metric Story Points for the size factor. Ziauddin et al. [102] provide guidelines for both size and complexity. For example, a very small story that

requires only a few hours of effort should be rated with a value of 1 Story Point, or a story that is very straightforward and clear (no unknowns), that can be implemented with basic programming skills, and that requires no research, should be given a complexity value of 1 according to the guidelines. However, according to the effort estimation model, the effort of an individual story is calculated by the product of its size and complexity values. Adding up the effort of all user stories results in the effort for the entire project. In addition to the effort, Ziauddin et al. [102] include factors such as team velocity, project duration, and development costs in their model and again give guidelines for the range of values for some of those factors. Further, Ziauddin et al. [102] introduce a metric, the so-called span of uncertainty. It is based on a team's confidence level, indicating their confidence regarding their estimates. For example, a level of 100% means the calculated time is the most probable. The confidence level can be used to calculate a lower bound, the optimistic point, and an upper bound, the pessimistic point, which in turn is used by Ziauddin et al. [102] to calculate the impact of confidence on time. Overall, the accuracy metrics MMRE and PRED(MMRE) calculated with the empirical data of the 21 software projects indicate that the estimates performed with the developed model are accurate. However, Ziauddin et al. [102] mention that the model should be further investigated and may still contain weaknesses.

3.2. Effort Estimation in Large-Scale Agile Development: Process, Challenges, and Mitigation Propositions

Bick et al. [6]

Due to the nature of large-scale agile development, the coordination of multiple teams is required, which often poses a challenge. Bick et al. [6] analyze the overcoming of this challenge through a multiple case study conducted at the German software company SAP between October 2013 and December 2014. In particular, Bick et al. [6] identify the coordination mechanisms and practices applied to mitigate the challenge of inter-team coordination in scaled agile environments. Bick et al. [6] aim to fill the research gap in this context by answering the research question, "How inter-team coordination is achieved in large-scale agile development systems." Inter-team coordination is essential to identify dependencies between different teams, which poses a significant challenge in estimating effort. If dependencies on other teams are not clarified in advance, it can lead to delays and overruns that can negatively affect the success of a project. For this purpose, Bick et al. [6] comparatively analyze the approaches to managing dependencies between teams through more than 60 interviews with key roles, such as Product Owners or Scrum Masters, from five different development programs. Each program has a different setup and approach to managing inter-team coordination. Bick et al. [6] refer to the five programs as traditional case, cloud case, distributed case, co-located case, and modular case. The interview participants explained their perception of the quality of coordination within their program. In the traditional case, the quality of coordination was perceived differently. Some saw the quality as in need of improvement but acceptable, while others perceived the quality as highly problematic. In

the other four programs, the interview participants rated the quality and management of inter-team coordination as good to very good [6]. Overall, the analysis shows that approaches to inter-team coordination vary greatly. In the traditional and distributed cases, coordination focuses on top-down planning, while the co-located and modular cases pursue bottom-up adaptation. In the cloud case, Bick et al. [6] identify elements of both approaches. Additionally, Bick et al. [6] analyze the proactive management of dependencies between teams. Different practices are used to actively manage inter-team dependencies in the cloud, distributed, and co-located cases, while no active management of dependencies was identified in the other cases. Best practices include, for example, recurring meetings to jointly plan all development activities, which also strengthens collaboration. The results indicate that proactive management of inter-dependencies by both management and team members seems to work best when the focus is on collaboration and iteration. Bick et al. [6] recommend holding iterative planning activities, such as effort estimates, especially at higher levels, well in advance of each Sprint. In addition, to solicit and incorporate feedback from experienced team members to improve coordination between teams in all constellations, whether top-down or bottom-up.

Bick et al. [7]

Research often sees a hybrid approach combining traditional and agile practices as a solution to minimize challenges in large-scale development. However, these approaches often fail, but it is unclear why. In this regard, Bick et al. [7] investigate how and why a hybrid approach results in inefficient coordination and project failure. To achieve their research objective, Bick et al. [7] conduct a case study with 13 teams within a large-scale development at a global enterprise software company. Within the software company, there were individual development teams that worked according to agile methodology, whereas activities related to coordination between multiple teams were performed using traditional methods. Bick et al. [7] use the term "coordination" in the sense of managing dependencies and the term "inter-team coordination" in the mind of activities between multiple teams within a multi-team system. The main goal is to explore what triggers ineffective coordination in a hybrid approach. Based on the findings, Bick et al. [7] create an explanatory process model of effective coordination within a hybrid setting. The model does not represent the optimal solution for effective coordination but instead explains the origin of significant challenges to effective coordination. Bick et al. [7] identified the lack of dependency awareness as the main reason for ineffective coordination and the failure of hybrid methods. This mainly results from the misalignment of planning activities such as specification, prioritization, estimation, and allocation between the agile teams and the inter-team level. This misalignment leads to many challenges in the implementation phase. For this thesis, the estimation planning activity is of particular interest, so the other factors, specification, prioritization, and allocation, will not be discussed in detail below. However, it should not be overlooked that these factors still play an essential role in large-scale agile development and are somehow related to estimation. Bick et al. [7] briefly describe how the estimation process is conducted at the case organization. The team estimates the open tasks with a high degree of granularity at the

beginning of a Sprint in a joint discussion on a team level. Team-level estimates are adjusted in case, e.g., new information is available. A rough estimate is made at the inter-team level, relying on the responsible parties' experience with multi-team systems. It was found that the case organization had no standardized procedure for sharing information between teams. Consequently, team-level estimates were not transparent, so teams could not consider each other's estimates in their planning [7]. This lack of transparency led to additional problems, as teams could not predict when they would depend on or receive tasks from other teams. Bick et al. [7] describe this estimation misalignment as "imprecise top-down effort estimation and a lack of feedback mechanisms for bottom-up estimation adjustment." This underlines the necessity to link traditional top-down planning closer to agile bottom-up planning. To mitigate this problem, the misalignment of planning activities, and the resulting lack of awareness of dependencies, Bick et al. [7] suggest holding cross-team planning workshops or unit-wide retrospectives to manage the dependency awareness better. The research by Bick et al. [7] highlights the value and importance of an accurate estimation process to minimize challenges arising from poor effort estimation, resulting in successful project implementation and delivery.

Usman et al. [95]

As there is little research in the field of effort estimation in large-scale agile environments, Usman et al. [95] conduct an exploratory longitudinal case study to examine how effort estimation is carried out in large-scale distributed agile projects. Therefore, Usman et al. [95] investigate the estimation process in a scaling environment, analyze the accuracy of the estimates, and identify factors that influence the accuracy. In addition, they identify the factors' effect on the estimates' accuracy. To increase the reliability of the data, they collect data from archived research, as well as unstructured and semi-structured interviews. The case study was conducted in a very large project involving over 180 employees distributed across multiple countries and teams. The project is concerned with developing and maintaining a complex telecommunication software at the company Ericsson. The development teams apply agile practices in their daily work. At the same time, the project managers followed a hybrid approach of agile and plan-driven practices to coordinate and manage the work of the development teams. The data analysis reveals that the effort estimation process is carried out in two stages, involving many individuals with various roles. The expert judgment and the analogy method are applied in both stages to estimate the effort. In the first stage, everything necessary for the second stage is prepared, and initial estimates are made. If a special unit approves the estimates, the implementation phase begins, and the second stage estimates are performed. The findings show a tendency towards underestimation in both stages, with the estimates in the second stage being more accurate [95]. Further, Usman et al. [95] identify the following factors influencing the effort estimation: team maturity, multi-site development, customer priority, and the size and complexity of requirements. Aspects such as the coordination of multiple stakeholders, lack of detail in requirements, requirements with a large scope, and new team members immaturity, along with the usual challenges in a scaling environment, lead to significant complexity and challenges in the

effort estimation process. However, despite these findings, Usman et al. [95] identify that re-estimation positively impacts the accuracy and improves the estimates. To address some of these challenges, Usman et al. [95] recommend considering the factors that influence estimates during the estimation process and involving all concerned stakeholders, especially mature teams with the necessary technical knowledge and expertise, in the estimation.

Evbota et al. [27]

In a qualitative case study at Ericsson AB, Evbota et al. [27] conduct semi-structured interviews to examine the challenges related to collaboration while planning large-scale agile development projects. In this study, Evbota et al. [27] present a model that describes the relationship between different types of challenges. They mentioned that every agile team, regardless of its scale, must be able to estimate the required work, prioritize it, and then bring these insights together in a good plan for the upcoming iterations. Further, they identify challenges in all three parts of planning, while the challenges in the estimation are the focus of this thesis and therefore discussed in more detail. However, it should be noted that prioritization and planning are equally relevant when planning a large-scale agile development project. However, according to the results, it is challenging to make long-term estimations because the amount of content is too large. This challenge leads to teams being generally skeptical of estimations. In addition, the enormous scope and fast pace lead to a high amount of troubleshooting which is not always predictable and affects the available resources during an iteration. Another challenge identified by Evbota et al. [27] is the systematic monitoring of the estimation process. If there is a lack of systematic monitoring, the process can take a long time without significant progress being made. In the context of cross-functional teams, an unstructured estimation can, in the worst case, lead to team members being responsible for estimating items that do not fit their role and about which they have no knowledge. Further, findings showed that the learning curve for making a comprehensive agile estimation is steep, especially for new team members, as estimations are based on learning from past iterations and experience. In addition to challenges in technical abilities, like estimation, prioritization, and planning, Evbota et al. [27] have also identified challenges related to the context of planning, including work environment, team-building, and team spirit. Further, results show that the lack of suitable information channels makes exchanging knowledge about decisions in large-scale development difficult. Also, dependencies between teams and the associated cross-team communication and coordination are identified as significant challenges. A possible mitigation is coordination meetings, which, however, the study's interviewees criticized as boring and too short. Evbota et al. [27] emphasize the importance of building trusting relationships between all stakeholders, especially in large-scale agile development projects. Overall, typical agile ceremonies are well-received at the team level, mitigating some challenges, but the real difficulty often lies in cross-team communication. The model proposed by Evbota et al. [27] indicates that the ability to plan in a large-scale agile environment depends on the team's skills and the context in which the teams operate.

3.3. Effort Estimation in Large-Scale Agile Standard Software Implementation: Process, Challenges, and Mitigation Propositions

Ömüral and Demirörs [59]

Ömüral and Demirörs [59] conduct a literature review in which they examine literature that deals with the methods of effort estimation for ERP projects, as well as their validation and limitations. For this purpose, Ömüral and Demirörs [59] limit themselves to online journals between 2000 and 2016. Ömüral and Demirörs [59] justify their research by stating that until their study, no systematic review specifically addressed applied methods for effort estimation in ERP projects. The research objective is to understand the effort estimation methods used in ERP projects and identify potential areas that can be improved. Ömüral and Demirörs [59] examine 41 publications in detail and classify and compare the methods each publication considers. The results show that none of the 12 analyzed publications that deal with effort estimation methods address the same effort estimation method. Still, the most commonly used size metric in effort estimation is Function Points in the form of COSMIC Function Points and IFPUG Function Points [59]. Furthermore, Ömüral and Demirörs [59] identify that 9 of the 12 publications validate the estimates, most commonly based on historical data [59]. However, Ömüral and Demirörs [59] argue that ERP systems evolve based on customer and industry expectations, noting effort estimation for ERP projects should be based more on customer needs than on historical data [59]. Overall, it should be noted that none of the publications address applying agile practices to implement ERP projects. As of 2016, no research appears to have been found in the context of effort estimation in large-scale agile ERP implementation. This suggests that applying agile practices in standard software implementation is a new area of research. The need for new approaches to effort estimation for the ERP domain, such as the agile methodology approach, is also emphasized in some of the publications [59]. With this insight, the research objective of this thesis can be justified.

Madanian et al. [50]

Madanian et al. [50] argue that typical failures in implementing ERP systems can be addressed by applying agile methodologies. Therefore they examine the critical success factors (CSFs) that can lead to success in agile ERP development and implementation. With their research, Madanian et al. [50] aim to create the basis for further research in this context and specifically lay the groundwork for establishing an agile industry-standard model for ERP development and implementation. For this purpose, Madanian et al. [50] conduct a systematic literature review to identify the CSFs in agile ERP development and implementation. Unlike previous research, Madanian et al. [50] focus on identifying CSFs of ERP projects while considering the adaption of agile methods for the development and implementation process. Madanian et al. [50] limit the data collection to research published after 2012 to examine the contemporary trends and identify current CSFs. To identify the context from various perspectives, studies of different types, such as case studies or literature reviews, are considered. For the analysis,

Madanian et al. [50] categorized common factors from various sources into themes, including team size, agile methodology, challenges, and success. The analysis results indicate a growing trend in using agile methods when developing or implementing ERP systems, such as Scrum, Oracle Unified Method, and SAP activate, as well as hybrid approaches that combine agile and traditional methods [50]. Unsurprisingly, the benefits of using agile methods in ERP development or implementation are similar to those of (large-scale) agile software development, including customer satisfaction, flexibility and scalability, faster delivery, and continuous improvement. Madanian et al. [50] identified the most commonly mentioned and important CSFs as communication and collaboration, stakeholder/user involvement, flexibility and adaptability, testing, and progress monitoring in terms of planning, feedback, and meetings. Madanian et al. [50] mention that these factors are essential to shorten the time frame and reduce the costs of ERP projects. However, Madanian et al. [50] did not mention anything about effort estimation and its challenges. Still, some of the identified CSFs can be helpful for successful effort estimation and mitigating occurring challenges. To fully answer their research question, Madanian et al. [50] plan to conduct further research in which data will be collected with quantitative interviews to identify additional CSFs. Overall, it can be observed that effort estimation in large-scale agile standard software implementation, such as ERP projects, seems to be a new area. Which once again justifies the research objective of this thesis.

This chapter provided an overview of existing research in the areas relevant to this thesis. The review of related work indicates a wealth of research on effort estimation in agile development at the team level. However, when it comes to the existing research on effort estimation in large-scale agile development, it was found that there were mainly publications that dealt with the estimation methods and size metrics employed. Still, research investigating the estimation process and the challenges associated with effort estimation is limited. In contrast, no research dealing with effort estimation in agile or large-scale agile standard software implementation was found. To partially fill this research gap, we conduct a case study described in the following chapter. With the case study, we investigate the effort estimation process in a large-scale agile setting, identify challenges faced by practitioners during effort estimation, and derive mitigation propositions to address these challenges.

4. The Case Study

This chapter presents the case study, which is intended to answer the first two research questions (discussed in Section 1.2). Specifically, the study objective, as well as the planning and preparation of the case study, are described in Section 4.1. Section 4.2 presents the data collection, and Section 4.3 discusses the data analysis, particularly the coding process. In Section 4.4, the case organization, as well as the setup of the program, is described. And at the end of this chapter, in Section 4.5., the case study results are presented, specifically the effort estimation process within the case organization and the identified challenges.

4.1. Study Design

The previously presented literature and frameworks associated with effort estimation in scaling agile environments and its challenges (discussed in Chapters 2 and 3) form the theoretical frame of relevance and ensure that the research is rigorous. To ensure the relevance of this research, the case study is used to investigate the effort estimation process and the occurring challenges during the estimation process in an organizational setting. The case study's approach adheres to the guidelines and recommendations by Runeson and Höst [71], tailored specifically for case studies within the software engineering context.

The case study's findings are then used to build the artifact of this bachelor's thesis: a set of challenges in effort estimation in large-scale agile development and mitigation propositions to address these challenges. Since the case study is carried out in an organizational setting, it is ensured that the delivered artifact is of practical value and relevance.

Regarding Yin [101] and Benbasat et al. [5], it is appropriate to conduct a case study methodology when studying a contemporary phenomenon in a natural setting. As already mentioned, the topic of effort estimation in scaling agile development is relevant to practitioners and researchers. Further, the case study is conducted at a program (discussed in Section 4.4) within a large active organization to understand the contemporary phenomenon in its real-life context, i.e., its natural setting. Yin [101] and Benbasat et al. [5] mention that case studies are appropriate when "how" or "why" questions are posed [5, 101, 63]. In this case, both research questions (Discussed in Section 1.2) to be answered by the case study are "how" questions: **RQ1: How is effort estimation conducted in the case organization?** and **RQ2: What are challenges in effort estimation in scaling agile environments?** Further, Benbasat et al. [5] emphasize that using the case study methodology is appropriate when the problem

is practice-based and when the experience of the involved people is important. Both points apply to our research. So, based on these key criteria and others for assessing appropriateness, applying the case study methodology is evaluated as suitable to fulfill the study objective and gain insight into practice.

According to Lethbridge et al. [48], data sources triangulation was achieved by collecting not only first-degree data in the form of interviews but also from third-degree sources such as documentation, presentation slides, and company websites provided by our contact person who is part of the program management. By taking different perspectives on the study objective, a more comprehensive representation is obtained, and the accuracy of the research is increased [71]. To collect the first-degree data, semi-structured interviews were conducted, which allowed for deviation from the planned order of questions, creating a natural and open conversation. Nevertheless, a questionnaire was designed as a basis for the structure of the interviews and to ensure that all topics were addressed during the interview. The entire questionnaire can be found in Appendix B.1. According to the guidelines of Runeson and Höst [71], the interview sessions were divided into three phases:

1. General Information
The first phase was a rather structured section including questions on the interviewees' current role within the program, their experience in agile and large-scale agile IT projects, and their organization's experience in agile and large-scale agile IT projects.

2. Effort Estimation within the Program:
The second phase, the central part of the interviews, covered questions on the effort estimation within the program, e.g., the time and scope of the effort estimation, the involved people during effort estimation, the estimation objects, and, most importantly, questions on the challenges the practitioners faced during the estimation process

3. Discussion:
The third phase was led by an open discussion on how research can support the effort estimation process in the industry.

In initial meetings, our contact person, who is part of the program leadership within the program, introduced us to the critical processes of the program and the program itself to ensure that we had a certain level of knowledge before the interviews. Further, additional meetings with our contact took place in which we introduced the research and its objectives and discussed the preferred roles of the interview participants. Our contact shared a brief overview of the research with those who fit the preferred roles and provided us with a list of potential interviewees. Subsequently, we invited these to the interview series, including the questionnaire, to ensure that the interview process would be smooth and efficient. The criteria the interviewees had to meet were that they had knowledge about the effort estimation process within the program, which automatically covered the aspects that they were part of

the program and worked in large-scale agile development environments.

4.2. Data Collection

The interviews took place between November 27, 2022, and December 31, 2022, and were all conducted remotely using video communication tools. In total, 20 experts participated in the interview series, with the longest interview lasting 50 minutes and the shortest lasting 27 minutes. The average duration of all the interviews was 41 minutes. The length of each interview depended on the interviewee's knowledge of the effort estimation process within the program and to what extent the interviewee had dealt with the questionnaire in advance. Two interviews were conducted as group interviews with three interviewees each. To stay within the planned time frame of 60 minutes, it was decided, with the consent of all involved people, that the interviewees of the group interviews provide the answers to the questions on the general information part via email afterward. Two researchers participated in each interview to ensure that all questions were adequately covered and that the time limit of 60 minutes was not exceeded. The interviews started with an introduction round and a brief overview of the research objectives. In addition, the interviewees were asked permission to record the interviews for analysis purposes, which all agreed to. As mentioned in Section 4.1., additional data was also collected from third-degree sources, such as presentation slides and documentation which our contact person provided.

4.3. Data Analysis

The primary goal of the analysis is to draw conclusions from the data while maintaining a coherent chain of evidence [71], i.e., that the reader should be able to track the progression from the gathered data to the resulting conclusions and findings [71, 101]. This section details the methods used to analyze the qualitative data to meet this goal.

Transcription

After the interviews, the corresponding recordings were transcribed. During transcription, sensitive data relating to individuals or companies were replaced with anonymized names. At the request of an interviewee, one interview was conducted in German. For the sake of completeness and to facilitate the data analysis, the transcript of this interview was translated into English. As previously mentioned, the participants of the group interviews submitted the answers on the first part of the interview via email. The given information was added to the beginning of the transcripts of the respective interviews. The transcription process was carried out using the qualitative data analysis software MAXQDA [1].

[1] https://www.maxqda.com/de

Coding and Analysis

The process of coding and analysis was performed according to the guidelines and recommendations of Miles et al. [53] and Saldaña [72]. First, a general overview of the data was obtained by reading the data set in the form of the transcribed interviews and additional data sources such as the presentation slides and documents. Following Miles et al. [53] and Saldaña [72], the interview transcripts were coded in a two-cycle approach using a combination of deductive and inductive coding. In the first cycle, descriptive codes were assigned to the text segments that summarized these in short sentences [53]. In the second cycle, higher second-cycle codes were assigned to the first-cycle codes. Further, additional second-cycle codes were assigned inductively as new insights were gained during the analysis of each interview's individual data [53, 72]. Once new codes were assigned for the second cycle, the previously coded data had to be re-coded. Therefore, an incremental approach was followed in coding, with the final coding resulting from the coding of all interviews. One researcher conducted the coding process and iteratively discussed it with a second researcher. Finally, both researchers analyzed the final coding results. Uncertainties during the analysis process were discussed among the researchers and resolved by consensus. If uncertainties regarding the organization, experts, effort estimation process, or challenges could not be resolved between the researchers, experts from the program were consulted for clarification. Both coding and analysis of the interviews were performed using MAXQDA [2] qualitative data analysis software and Microsoft Excel [3].

4.4. Case Description

To understand the significance of the findings in empirical research in software engineering, it is crucial to provide the context [26]. For this, the organizational environment and the context of the case study are outlined in this section. The case study was conducted at a very large program within a German network and infrastructure provider. The merger of the two largest providers in Germany triggered this program. To achieve greater efficiency, it was essential to standardize the different IT and ERP systems across the merged company. Thus, the program was launched to make the core processes around billing and market communication more efficient and uniform across the company by implementing an ERP cloud solution. To realize the objective, a cloud ERP system transformation was carried out using an existing standard ERP cloud solution from a large German software company which was further tailored to the company's needs and requirements. Not only were agile methodologies applied to implement the transformation as fast and efficiently as possible, but the company also worked closely with the ERP system provider - the large German software company - and a German subsidiary of an American consulting firm. This resulted in over 350 people being involved in the program.

[2] https://www.maxqda.com/de
[3] https://www.microsoft.com/de-de/microsoft-365/excel

4.4.1. The Organizations

The Network and Infrastructure Company - NetInfrCo

As mentioned above, the ERP Cloud System transformation was conducted at the Network and Infrastructure Company. After the merger in 2019, the company had over 72,000 employees (State: 2021) and is one of Europe's largest network and infrastructure providers, with a revenue of 77.4 billion EURO (State: 2021). The merged company, referred to as the holding company, and its subsidiaries are present in several locations throughout Europe, all of which are part of the program, i.e., each of these companies undergoes an ERP transformation. In total, 290 employees from the holding and subsidiary companies were involved in the program and were responsible for the transformation, implementation, and customization. For this, the employees mainly took on roles in agile teams. In sum, we interviewed 12 people from the Network and Infrastructure Company, with interviewee BC1 being an employee of a subsidiary company. The interviewees have key roles such as Scrum Master, Product Owner, Solution Architect, or Developer working in agile and large-scale agile software development between 1 - 10 years (see Table 4.1.). The interviewee BC1 acts as a Product Owner for the transformation within the subsidiary company and as a Business Contact for the corresponding subsidiary within the "central" program of the company-wide transformation. For simplicity, only data from the perspective of the Business Contact will be considered, and therefore, this interviewee will be assigned to the holding company (see Table 4.1.). The most frequently mentioned time frame to the question on the company's experience in both agile and large-scale agile software development is 6-10 years (see Appendix C.1., Columns 5 and 7). In the following, all companies are referred to as "company" irrespective of whether they are holding companies or subsidiaries.

The Software Company - SoftwareCo

The German Software Company is the provider of the cloud-based ERP system in the transformation program and supports the Network and Infrastructure company in carrying out the program. The company has more than 111,000 employees (State: 2022) and is the world's largest European and third largest publicly traded Software Company with a revenue of 30.9 billion EURO (State: 2022). In total, 40 employees have supported the ERP transformation both in the agile teams and at the program leadership level. In sum, we interviewed six people from the Software Company, with interviewee SolAr3 being an employee of an external partner. Since SolAr3 performs the same tasks and activities as the Solution Architects of the Software Company and the external partner works closely with the Software Company, SolAr3 is assigned to the Software Company for simplicity (see Table 4.1.). The interviewees have roles such as Solution Architect, Scrum Master, and Scrum Coach in the agile teams and, e.g., Program Manager at the leadership level (see Table 4.1.). The interview participants' experience in agile and large-scale agile software development ranges from 3 to 15 years (see Table 4.1). When asked about the company's experience with agile and large-scale agile software development, no consistent time frame is cited, with no responses covering a time frame of less than six years. (see Appendix C.1., Columns 5 and 7).

The Consulting Firm - ConsultCo

The German subsidiary of the American consulting firm is involved in the transformation program in a supporting role. The consulting firm has 25,000 employees worldwide (State: 2021) and has a revenue of 9.3 billion EURO (State: 2021). In total, 20 people were working in the transformation program within the Network and Infrastructure company. The employees were mainly responsible for the Project Management Office (PMO), acting in a coordinating role at the program leadership level and not directly in the agile teams. We interviewed two people from the consulting firm who have different levels of experience in agile software development. Still, both have 6-10 years of experience in large-scale agile software development (see Table 4.1.). The answers to the question of the company's experience in agile software development differ, but both interviewees gave a value of 11-15 years of experience in large-scale agile software development (see Appendix C.1., Columns 5 and 7).

No.	Company	Role	Alias	Experience in agile software development (years)	Experience in large-scale agile software development (years)
1	SoftwareCo	Agile Coach and Scrum Master	SM1	6 - 10	6 - 10
2	SoftwareCo	Project Manager	PJM1	6 - 10	6 - 10
3	SoftwareCo	Solution Architect	SolAr1	3 - 5	3 - 5
4	NetInfrCo	Developer	Dev1	1 - 2	1 - 2
5	NetInfrCo	Product Owner	PO1	1 - 2	1 - 2
6	NetInfrCo	Product Owner	PO2	3 - 5	3 - 5
7	SoftwareCo	Product Manager	PM1	6 - 10	3 - 5
8	ConsultCo	PMO	PMO1	11 - 15	6 - 10
9	NetInfrCo	Product Owner	PO3	6 - 10	6 - 10
10	NetInfrCo	Developer	Dev2	3 - 5	3 - 5
11	NetInfrCo	Scrum Master	SM2	6 - 10	3 - 5
12	NetInfrSubCo	Business Contact	BC1	1 - 2	1 - 2
13	SoftwareCo	Solution Architect	SolAr2	3 - 5	3 - 5
14	ConsultCo	PMO	PMO2	6 - 10	6 - 10
15	SoftwareCo	Solution Architect	SolAr3	11 - 15	6 - 10
16	NetInfrCo	Product Manager and Product Owner	PM2	6 - 10	6 - 10
17	NetInfrCo	Business Process Expert, Test Coordinator, and Training Coordinator	BPE1	3 - 5	3 - 5
18	NetInfrCo	Solution Architect	SolAr4	6 - 10	6 - 10
19	NetInfrCo	Product Manager	PM3	3 - 5	3 - 5
20	NetInfrCo	Scrum Master	SM3	6 - 10	6 - 10

Table 4.1.: Overview of the interview participants

As Table 4.1 shows, the interviewees gave different answers to the companies' experience in agile and large-scale agile software development. This can be attributed, on the one hand, to the uncertainty of some interviewees, as mentioned by some themselves during the interview, and on the other hand, to the size of the respective companies. If time frames were stated above, then these are the ones most frequently mentioned by the interviewees of the respective companies.

4.4.2. The Program Hierarchy

The first phase of the program took place in 2019, during which preparations and planning took place for the first realization phase, which began in 2020. As mentioned above, the ERP was to be transformed company-wide, which meant that representatives of almost all companies of the NetInfrCo, as well as stakeholders of the SoftwareCo and ConsultCo, were involved in the first phase, referred to as pre-implementation phase. With the goal of a fast and effective transformation, a well-planned "Program Set-Up" was established, allowing for the transition to the Cloud ERP System within 15 months per company. The objective of "Going Live" in 15 months was achieved by dividing the transformation work among responsible teams and efficiently organizing the program. The "Program Set-Up," shown in Figure 4.1., is briefly discussed in the following.

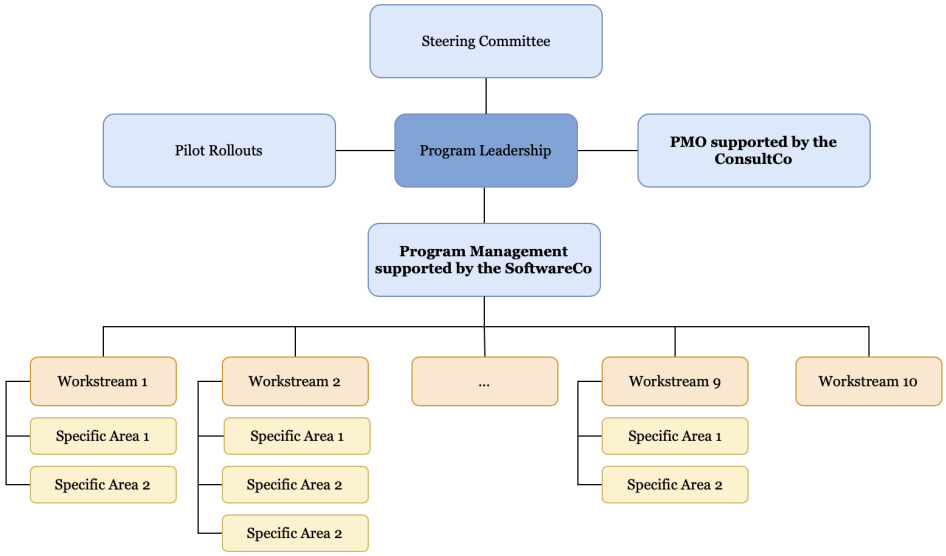

Figure 4.1.: The Program Set-Up simplified and generalized

The Program Leadership

The responsibility for tasks related to the overall organization and coordination of the program, monitoring the progress, and steering the program falls on the program leadership. This

leadership group comprises the Steering Committee, Project Management Office (PMO), Program Management, and Pilot Rollouts. The consulting firm assists the PMO, while the software company supports the Program Management. The transformation of the ERP system is essentially carried out by the respective workstreams (discussed in more detail below), which are mainly supported by the Program Management in important decisions and planning related to the transformation. In total, three individuals from the program leadership participated in the interview series, including one from the software company involved in the program management and two from the consulting firm engaged in the PMO. Since the program leadership is not directly involved in the estimation process and the focus of this case study is on this process, the limited number of interviewees from the program leadership is not considered problematic.

The Workstreams

Tasks regarding customization and implementation are distributed among responsible teams. Therefore, the "Program Set-Up" comprises ten designated workstreams, each responsible for a specific business area. Within most of these workstreams, further sub-divisions have been made. Each workstream has one designated Product Manager, who is not part of the agile Scrum teams but acts as the link between program leadership and the Scrum teams within the respective workstream. The Product Manager provides assistance and guidance to their teams if needed and keeps track of the progress of all Scrum teams within their workstream, taking on a coordinating and supportive role. Additionally, they are responsible for organizing dependencies between their workstream and others. In this program, the agile Scrum team typically consists of a Product Owner, a Solution Architect, 2 to 5 IT team members responsible for technical tasks, and 2 to 5 team members responsible for documentation and functional specifications, making the team size 6 to 12 members in total. The teams are responsible for the implementation, planning, and estimation of the tasks assigned to them. Most interviewees were members of a Scrum team and, therefore, also involved in the corresponding workstream. To increase the validity of our results (discussed in Chapter 4.5), we interviewed members from five different workstreams who were associated with different Scrum teams within their respective workstreams. It should be noted that some interview participants held more than one role within the program, e.g. SM1 or PM2.

Cross-Team Roles

In addition to the roles within the workstreams or program leadership, there are also cross-team roles, such as Scrum Masters or Agile Coaches, which employees of the software company frequently took on. Their role was to support the teams and program leadership with their expertise in agile methods and practices and to ensure that the agile values and principles are used in the right way and are understood by every member of the program. Further, they introduced agile work methods and estimation techniques to the teams.

Regarding the size of the program and the number of people and Scrum teams working

together using agile methodologies to implement and adapt the standard cloud solution, this program can be classified as a (very) large-scale agile program according to the taxonomy of this thesis, as mentioned in Section 2.1. However, it should be noted that this program is not one in which new software is developed but an ERP transformation in which one of the software company's existing standard software solutions is implemented and adapted accordingly. For the adaptation of the standard software, requirements were defined in the pre-implementation phase in 2019. Further, in the planning phase of each company, more requirements are specified if necessary(discussed in Chapter 4.5). All requirements are then classified to determine whether they only needed to be adapted accordingly or did not yet exist in the standard solution and had to be newly developed to meet the customer's needs.

4.4.3. The Effort Estimation Process

For such ERP transformation projects, the software company, whose standard solution is to be implemented, has developed an approach in the form of a framework. This framework is a hybrid approach consisting of parts of the Waterfall methodology and parts of the Scrum methodology (*SolAr3*). It was developed to support customers during the implementation, migration, or system conversions of standard software provided by the software company. It includes six phases that an optimal transformation should go through. As the thesis focuses on the effort estimation processes using agile practices and the challenges that occur during the process, only the preparation and realization phases are relevant in which agile practices are applied, especially in the latter. The remaining phases are irrelevant in the context of this thesis and will, therefore, not be discussed further.

The planned duration of the company-wide transformation is set to five years (*PMO1*). As described in Section 4.4, the planned duration of a company's transformation is 15 months, including the preparation phase. To efficiently complete the company-wide transformation within the planned five years, the program envisages the rollout of several companies in parallel within these 15 months (*PMO1, PO3*). In the following, the companies whose transformation is planned for the following year are referred to as "rollout companies." The overall estimation process within the case organization is visualized in Figure 4.2.

It should be noted that the results may vary in some cases because the interviewees work in different teams, which, due to the nature of the agile methodology, are self-organized and self-determined, which can lead to the use of different approaches. However, the analysis showed that there were only a few differences but overall the approach to estimating the effort within the teams is very similar. During the analysis of the estimation process and planning events at the highest level, the roadmap planning, there were some disagreements among the interviewees. This is not surprising, as most interviewees operate at a team level and are generally uninvolved at the highest level. Nevertheless, these discrepancies had to be eliminated. For this purpose, our contact person, part of the program leadership, has ensured clarity by providing accurate information. Further, the idea behind story points is to estimate requirements based on their complexity, degree of difficulty, and scope for

estimates to be free of influencing factors such as the skills of individual team members. Therefore, normalizing story points to, e.g., person days does not strictly adhere to pure agile best practices. However, in large projects like the case study program, each individual has a different experience using story points resulting in different understandings of this size metric. This makes it difficult to, e.g., compare teams, calculate metrics correctly, or for teams to estimate with this size metric. As a result, at the beginning of the program, the program leadership decided to normalize story points to person days so that all of the over 350 people involved have a shared understanding and reference for the estimation unit.

The Preparation Phase

The preparation phase is performed in the fourth quarter of a year and forms the basis for the realization phase for the following year (*Dev2, PM2, PO2, PO3*). In addition to organizational preparations such as staffing or tooling, an important event called the "Fit-To-Standard Workshop" takes place. During this workshop, the rollout companies, whose transformation is planned for the following year, familiarize themselves with the standard ERP solution and gather their requirements based on the functionalities of the standard solution reflecting their needs for the ERP system (*PJM1, Dev2, PM2, SolAr4, PM1, PO3, PM2*). These requirements are then assigned to one of the four classifications (*SolAr4*): Fit, Gap, Non-Functional, and WRICEF, an acronym for the sub-classifications Workflow, Report, Interface, Conversation, Enhancement, and Form combining these in one classification. For example, a "Fit" is a requirement that already exists in the standard solution and only needs to be further adapted through configuration or customization. While a "Gap" is a requirement that does not exist in the standard solution and needs to be developed and implemented from scratch. Most interviewees agree that the classification of requirements does not play a role in effort estimation. However, three of the interviewees believed that classification has an impact on the effort estimation, as it is clear in advance that a "Gap" is more complex and therefore requires more effort to implement than, for example, a "Fit" (*SolAr1, Dev2, SolAr2*).

> *"Yes, it has an influence because some gaps normally need more documentation. And normally are related to real developments and not only to customizing or changing settings on the systems. So it's more to do for these objects and more to test and also to document." (SM3)*

However, after classification, the requirements are discussed, validated, and prioritized as long as a design authority has approved them. The scale low, medium, high, and very high is used for the prioritization. For example, requirements considered legal bindings are prioritized as very high, while requirements considered "nice to have" are prioritized as low. Then, the prioritized requirements are moved into the backlog, *"creating the initial backlog" (PJM2)*, and assigned to the responsible workstreams. As explained by our contact person, the program follows a "Design to Budget" approach, which seeks the most cost-effective solution for individual components during development. This implies that there may also be cases where low or medium prioritized requirements are removed from the backlog. However, the initial prioritized backlog, specifically the result of the Fit-To-Standard Workshop, is

documented, forming the basis for the following year (*Dev2, PM2, PO2, PO3*).

> *"As an example, we have, let's say, 800 gaps given in an Excel, and then we discuss it with the rollout entities. And in some cases you might come to the conclusion, no, that is, for example, no gap because it's already available or they won't get it or whatever. So then the gap will not be further processed in the project." (PO1)*

Within the program, there are three levels at which the effort estimation is performed: the requirement level, the work package level, and the work item level. The respective objects are estimated at all three levels, partly using different units. To estimate, there are three designated events: Roadmap Planning at the requirement level, Wave Planning at the work package level, and Sprint Planning at the work item level. It should be noted that the estimation at the next higher level serves as the basis for the underlying level. In other words, the estimates on the requirement level form the basis for the estimation on the work package level, which in turn forms the basis for the estimation on the work item level. These three levels or events will be explained in detail below.

> *"We have requirements, work packages, and work items, and we estimate all three of them. We use story points and effort points which are theoretically two different, whatever. But in the end, throughout the whole program, it's aligned that one story point equals one effort point equals one man day." (SM3)*

The Roadmap Planning

The Roadmap Planning takes place after the Fit-to-Standard Workshop (*PM1, PO3, PM3*) and is the event where effort estimation is first carried out (*PO1, PO3, PM3*). This planning event is conducted once for each rollout company before the realization phase (*SM1, Dev1, PO1, Dev2, SM2, PM3*). During this planning event, the prioritized requirements from the initial backlog are estimated in IT effort in person days (*SM1, Dev1, SolAr2, SM3*). At this point, little is known about the implementation of the requirements, which is why the effort is only estimated very roughly (*SM1, PJM1, Dev1, PO2, PM1, PMO1, SolAr2, PM2*). In this process, the stakeholders confer, discuss the requirement, and agree on an estimate based on their experience, knowledge, and gut feeling. To minimize the complexity of the estimation process, the number of involved individuals is kept as small as possible (*PMO2*) and is limited to selected members, which usually include the Product Manager, the Product Owner, Solution Architects, and IT members of the respective workstreams and teams, as our contact mentioned. The result of the Roadmap Planning, the Product Backlog with prioritized and initially estimated requirements, forms the basis and starting point for the realization phase.

The Realization Phase

The realization phase is limited to 9 months and divided into three so-called Waves, each lasting three months (*PMO1*). A Wave consists of three regular Sprints in which the respective requirements are implemented and an additional Sprint in which no new requirements are implemented but the already implemented requirements are tested and the next Wave is planned (*SM2, SolAr2*). Throughout the realization phase, there are two recurring events in which effort is estimated, the Wave Planning and Sprint Planning. All interviewees mentioned that the entire Scrum team participates in both events and is jointly responsible for the effort estimation. Further, the objects on both levels are estimated in Story Points. Initially, some teams tried out estimation techniques like Planning Poker (*PJM1, SolAr1, PO2, PM1, Dev1, SM2, BC1, SolAr2, SolAr3, SolAr4*) but have then replaced these techniques with a simple expert judgment based on knowledge and gut feeling. As the analysis showed, expert judgment is used by all interviewees, although some teams also combine this technique with an analogy, comparing the objects with the estimates of previous and similar objects (*SM1, Dev1, PO2, Dev2*).

> *"Now, for most of us it has proven the best approach to be a rather hand on combination of gut feeling, experience based assessment and also recycling of old estimates for similar tasks from the past." (SM1)*

The Product Owner does not actively participate in the estimation but is responsible for prioritizing the items, coordinating the team, moderating the estimation process, and supporting the team during the process with his expertise from a business perspective (*PO1, PO2, PM1, PO3, SM2, SolAr3, SolAr4*). To guarantee the collaboration of all workstreams or Scrum teams and to achieve consistency, the Story Points were normalized for the entire program, as mentioned above. Accordingly, one Story Point is translated into one person day, i.e., 8 hours. In addition, the Program Leadership has set further guidelines regarding the capacity of the teams, whereby non-productive activities have been taken into account. For example, a "full" team member responsible for implementing the requirements has 13 Story Points capacity per Sprint. Keeping this in mind, both planning events are discussed in the following.

The Wave Planning

The first Wave Planning Event takes place after the Roadmap Planning so in the last quarter of the year. Therefore, the first Wave is already planned before the realization phase begins. The second and third Waves are planned in the Wave Planning Events, which take place during the fourth Sprint of the previous Wave, as mentioned above. The basis for these events is the main outcome of the Roadmap Planning Event, the roughly estimated and prioritized requirements placed in the Product Backlog. Each requirement must be broken down into smaller parts, so-called work packages. Each work package is a functional subset of a requirement that can be completely implemented within a Wave, i.e., three months (*Dev1*). Thus, as all interviewees mentioned, a requirement can be broken down into n work packages. If the work packages

do not have a high level of granularity, it may be necessary to move them into the next Wave. This can lead to challenges and, in the worst case, result in delays in the realization phase, as *PMO1* mentioned. After decomposition, each work package is discussed to ensure a common understanding and then estimated by the entire team. When estimating the effort of a work package, the initial estimate of the corresponding requirement is divided among the *n* work packages, with the distribution not being even but based on the complexity, risk, and effort of each work package (*PJM1, PO1, PM1*). Therefore, each work package is compared to the other (*n - 1*) work packages (*PJM1, PO1, PM1*). At the work package level, the team has more information about the actual implementation of the items, which allows for adjustments to be made to the estimates by the entire team if the initial estimate of the requirement is too high or too low (*PM1, PMO2*). In case of under/overestimation, the adjustment is only made at the work package level and not retroactively for the requirement level (*SM1, PM1*). *PJM1, SolAr1,* and *Dev1* mentioned that the work packages are additionally estimated with value points, which are intended to help the Product Owner prioritize the work packages in the Product Backlog. For this purpose, the program leadership has predefined a value points scale, according to which the work packages should be estimated. For example, a work package with the functionality of "reducing the complexity of processes and thus saving time in the overall process" should be estimated with three value points corresponding to a medium priority. At the Sprint level, work items are implemented, a functional subset of a work package that should be fully implemented within one Sprint (*PJM1, Dev1, SM2*). The breakdown of the work packages into *n* work items is done in the same way as the breakdown of the requirements into work packages (*SM1, PO1, PM1*). This decomposition and the allocation of the work items to the respective three Sprints within the Wave are also part of the Wave Planning (*SM1, PJM1, PO1, PM1*). During the event, particular focus is placed on ensuring a common understanding of the work packages to be implemented. Each Wave Planning event is followed by a Wave Plan Presentation meeting planned by the PMO. The Product Owner of each Scrum team presents their planned Wave to discuss dependencies between workstreams or other Scrum teams and make appropriate changes if needed, as the majority of interviewees mentioned.

> *"So we have the effort estimation in the requirements, and we break it down in the work package and try to divide the effort into the work package. Also, for the work items, we do the same. Only in case we have some unforeseen items we can increase it, but we will not then touch the requirements if we say the requirement is 50. And then in the work package, we see it's more than we adjust the work package, we not adjust backward the requirement." (PM1)*

The Sprint Planning

At the beginning of each Sprint, the Sprint Planning event is conducted, where the next Sprint is discussed and planned with the entire Scrum Team, as every interviewee mentioned. The work items defined and created during the Wave Planning serve as the starting point for this event, which are further discussed in detail to ensure that each team member understands each work item before estimating it. As some interviewees noted, an insufficient breakdown

of work items, as stated above with work packages, can result in work items having to be moved to the next Sprint, causing delays if not resolved at an early stage. The effort estimation process is carried out similarly to the one on the work package level, where the estimates of the respective work packages serve as the basis for the estimates at the work item level. So, the total estimate of a work package is divided among its corresponding work items. Estimates are adjusted if the effort of the work package was under/overestimated due to a lack of information, as the majority of interviewees noted. The estimates on the work item level are the most accurate as the most information on the actual implementation of each item is known. Most interviewees mentioned that the experts on the team are the option leaders giving estimations based on their experiences and gut feeling, which are then challenged, discussed, and finalized by the entire team in case of discrepancies which is the most time-saving approach. As mentioned above, the entire team is involved during the effort estimation process, with the Product Owner supporting and not actively participating in the estimation (*PO1, PO2, PM1, PO3, SM2, SolAr3, SolAr4*). Optionally, work items can be broken down into tasks estimated in hours, but none of the interviewees mentioned that their team specifically estimates tasks. In addition to the adjustments made during the Wave and Sprint Planning events, program leadership has determined that at least one Product Backlog Refinement meeting of 6 hours should be held in each Sprint. During the Backlog Refinement, estimations, prioritization, and the decomposition into work packages or work items are adjusted, and newly added requirements are discussed and estimated (*SolAr1, PO3, PM1, Dev1, BC1, SolAr3, PM3*). Therefore it is ensured that changes are reacted to quickly, that sudden influencing factors such as dependencies, over/underestimation, and sudden events affecting the system are not neglected, and that the Product Backlog is continuously adjusted and up-to-date (*SolAr1, PO3, PM1, Dev1, BC1, SolAr3, PM3*). Several teams did not hold this Backlog Refinement meeting for 6 hours straight but spread it out over the Sprint, for example, three meetings of 2 hours each (*PJM1, SM2*). Besides the Product Backlog Refinement meeting, *SM3* mentioned that his team adjusted the effort estimations during daily meetings if necessary.

> *"Within the sprints or within the waves, there is this refinement, where you look at how things have gone so far, what we have done so far, or what will happen in the future so that you can possibly make adjustments if you see that there have been changes." (SM2)*

The Documentation and Monitoring

Among all the interviewees, two tools were mentioned that are used for documentation and monitoring throughout the program in the context of planning and effort estimation. The software company offers a tool that provides tools, content, and services to support customers in implementing their systems. This tool is the primary tool or, as some said, the "single source of truth" (*PJM1, SolAr1, PO1, PM1, PO3, SolAr2, SolAr3*). The tool is used, among other things, for documenting estimates, the Wave and Sprint Planning, and maintaining the Backlogs. This tool creates transparent communication and provides data that can be used for analysis purposes or comparisons for future estimates (*Dev1, PMO1, SM3*). However, most

agreed that this tool does not provide any support during the estimation process but only records the result of it. In addition to this tool, Excel [4] is used, especially at the requirement level, to maintain and document requirements along with their classification, prioritization, and estimation (*PO1, PO2, PM1, PO3, SM2, PM2, PM3, SM3*). Additionally, any dependencies that are already apparent during the Fit-to-Standard workshop are documented and maintained in this Excel too. Building on the tool provided by the software company, the consulting firm built an Excel dashboard for reporting reasons based on the existing project data from the tool to visualize certain metrics and the project process (*SM1, PMO1, PMO2, BPE1*). This dashboard was more important and relevant for the program leadership than for the workstreams or teams, as most respondents noted. Further, the analysis showed that hardly any team uses metrics to monitor the effort or the estimation accuracy, arguing that this is only relevant for program leadership. However, *BPE1* states that a completion metric is used within his team:

> *"We have some kind of percentage of completion measure that we apply in our wave planning [...] so we rely on or we apply certain percentage of completion values inside the different work packages. So not to lose the full overview, what has been done, what needs to be done."*

Although almost everyone agrees that the actual effort expended for implementation is relevant to improve the estimation performance, it is not actively tracked, as it is challenging and complex. *BC1* notes that his team discusses the actual effort but does not actively track it.

[4] https://www.microsoft.com/de-de/microsoft-365/excel

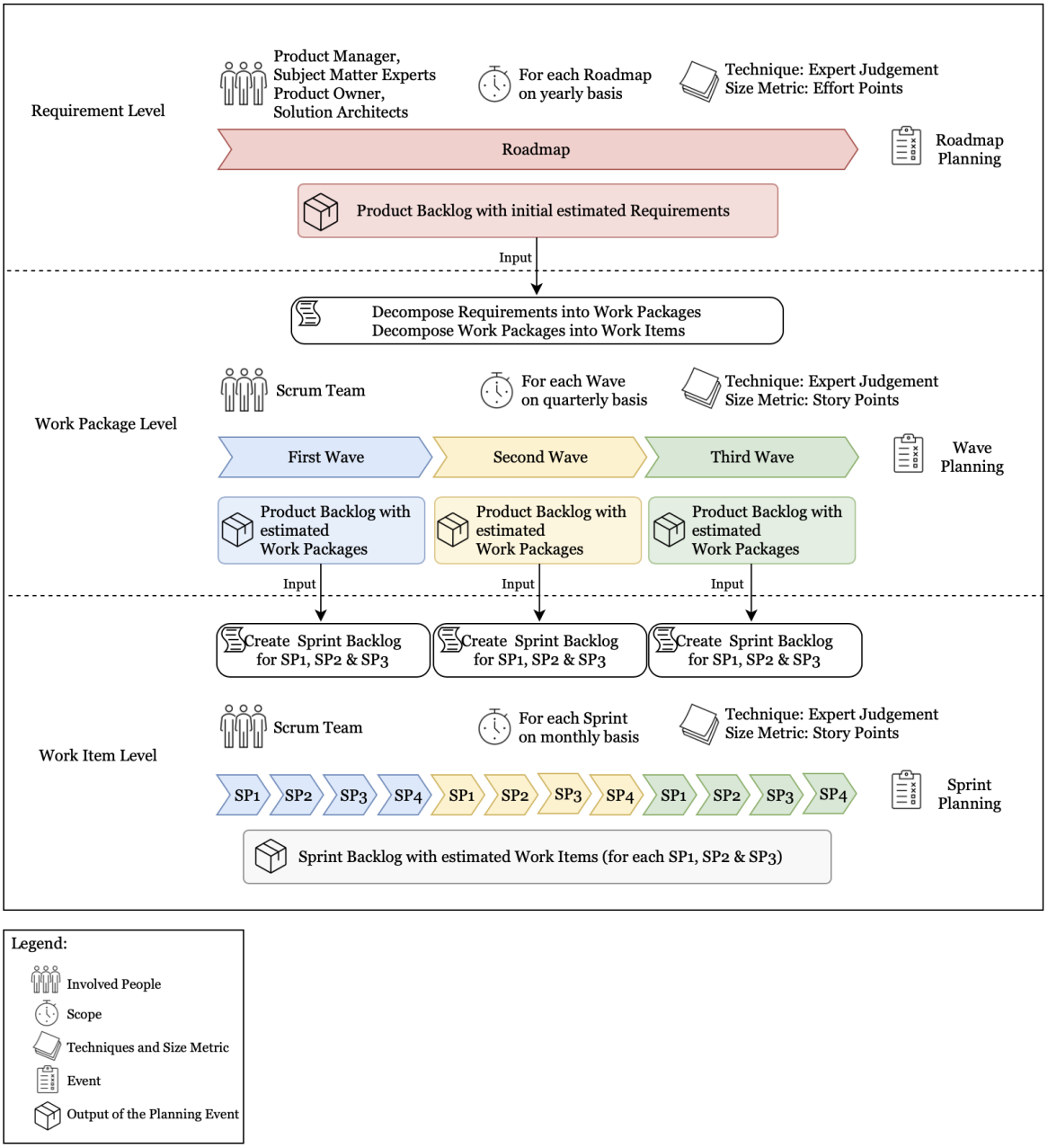

Figure 4.2.: Visualization of the Effort Estimation Processes at the Case Organization

4.4.4. The Challenges

This section summarizes all identified challenges (Table 4.3), from which the most frequently mentioned challenges in effort estimation are described in detail. A total of 25 challenges were identified, not all of which are typical for large-scale agile development. However, to ensure a comprehensive representation of the current situation in the context of effort estimation, they were included in the collection of challenges. In an additional column in Table 4.3, challenges typical for large-scale agile development are marked with the abbreviation LSAD. As mentioned above, the underlying case study involves a standard ERP implementation with the application of agile practices. Therefore, it was decided to also mark challenges in the context of effort estimation that are typical for large-scale agile standard software implementations. These are abbreviated as SS in the additional column in Table 4.3. During the analysis, specific clusters were identified that led to an overarching categorization. Therefore, the 25 challenges were assigned to four main categories, which are briefly described in Table 4.2.

Category	**Description**
Program Setting	This category includes challenges related to the program setting that result from preliminary decisions made at the highest level, the program leadership
Collaboration	This category includes challenges related to the collaboration of all levels (team, workstream, and program leadership)
Expertise	This category includes challenges related to the expertise, which is often neglected or missing during the estimation process and hinders accurate estimation
Uncertainties and Information Deficit	This category includes challenges related to uncertainties and an information deficit, which exists especially at the requirement level

Table 4.2.: Identified Top Categories

One of the most frequently mentioned challenges is the **time restriction (C2)**, which has been assigned to the program setting category since the time is linked to the predefined duration of the program. The general time restriction within the entire program leads to several challenges in the context of effort estimation, which were mentioned by *PJM1, PM1, PO3, SM2, BC1, SolAr2, PM2*. Primarily, an accurate estimation using the expert judgment method requires time to discuss the individual requirements, work packages, or items and then agree on the estimation *(PM1, SM2, BC1)*. The discussion is often shortened due to time restrictions by not estimating in the entire team but rather having experts take the lead and estimate the items, resulting in an estimation process that is not held as ideally as it should be when following the agile methodologies *(PJM1)*. In the context of time restriction, *(PO3)* mentioned that meetings suitable for adjusting estimations and discussing learnings are not held due to time restriction. Building on this, *SolAr2* mentioned that identifying dependencies is time-consuming. However, having additional meetings to estimate these dependencies and include them in the estimation is difficult because time is already scarce. Additionally, in this context, the estimation process in teams is often seen as overhead because it takes away time from the actual implementation work, as *SM2* says:

> *"The colleagues aren't all that enthusiastic about all the events we have anyway. It always takes up a lot of time and they want to have more time to work. I can understand that, but it's important to look at it again and again, and to dedicate yourself to it."*

Another challenge is **considering dependencies to other teams, workstreams, and systems in the estimation (C14)**. This challenge has been assigned to the collaboration category since it relates to the collaboration between different teams, workstreams, and systems, with existing dependencies being one of the most well-known challenges in large-scale agile development. In total, *PJM1, SolAr1, Dev1, PO1, PO3, SolAr2, PM3* mention this challenge. *PJM1, SolAr1,* and *SolAr2* state that, especially in the system's modules that build on and interact with each other, there are many dependencies that are difficult to identify in an early stage. Additionally, the interviewees *PJM1, SolAr1, Dev1, PO1, PO3, SolAr2,* and *PM3* mention the challenge of considering dependencies between other teams and workstreams during the estimation process. *Dev1* summarizes the problem of dependencies as follows:

> *"Some areas that have to be covered might be black boxes for you. So you have no idea of what has to be estimated. [...] Maybe you don't know what somebody else has to do in the third-party application. But you as the estimator you have to also calculate it within your estimation. But you can't, you can't know it basically."*

The most frequently mentioned challenge is the **unclear or incomplete specification of the requirements (C23)**, which is not typical for large-scale agile development but was mentioned by ten interviewees *(SM1, SolAr1, PO1, PMO1, SM2, SolAr2, PMO2, SolAr3, SolAr4, PM3)*. All interviewees who expressed this challenge agree that this challenge is one of the main reasons why the effort estimation process is such a challenge for the teams. Especially at the requirement level, there is a huge information deficit, which hinders accurate estimation on work package and work item level. The information deficit is manifested in incomplete, weak, and unclear functional descriptions of the requirements, which provide little information about what is to be implemented. *SolAr3* gives an extreme but straightforward example to illustrate such descriptions and the associated challenge:

> *"[...] We are just handed over some sentences, we need to have, as an example, we need to have that button in blue. But we do not know where does that button now need to be positioned? What functionality does that button have?"*

All challenges are shown in Table 4.3, including a brief description based on what the interviewees said, whether the challenge is typical for (large-scale) agile or standard software implementations (abbreviated with CL for classification), and which of the interviewees mentioned this challenge. The classification of whether a challenge is typical for (large-scale) agile was performed based on the values and principles of the agile methodology [35], the existing literature, and our expertise in this context. The classification of whether it is a

challenge typical for standard software implementation was also based on the challenges mentioned in the existing literature and our expertise.

ID	Name	Description	CL	Interviewees
Program Setting				
C1	Project setting	A fixed, large time frame requiring initial estimates and the tight tying of estimates to the budget limit the flexibility of estimations and lead to inaccurate initial estimates	SS	PJM1, Dev1, Dev2, SM2, SolAr2, SM3
C2	Time restrictions	There is a lack of time for estimating requirements, but the process of estimating effort and identifying dependencies is very time-consuming	LSAD	PJM1, PM1, PO3, SM2, BC1, SolAr2, PM2
C3	Unclear responsibilities	Lack of knowledge about the responsibility in terms of person, team, or workstream for certain requirements makes effort estimation difficult		PO2
C4	Pressure and control by management	Pressure and control from management, resulting in inaccurate estimates to meet management expectations and feeling pressured to be faster than management's original estimates		SM1, SolAr1
C5	Lack of measures to improve estimations	Lack of measures/metrics that measure planned and actual efforts resulting in less potential for improvement in future estimates		Dev2
C6	Inappropriate tool support	The complexity and inflexibility of the tool, as well as the lack of features that could support the estimation process, makes the requirement breakdown process, planning, and documenting of estimates difficult	LSAD	PM1, PO3, SolAr2
C7	Monitoring of estimations and actual efforts	The monitoring of estimates and actual effort is difficult due to the complexity and size of the program, the lack of commitment to transparency, the lack of keeping the backlog updated, and the wrong granularity of estimation objects	LSAD, SS	SM1, PMO1, PMO2

C8	Difficulty to estimate in story points unit	Even if companies define a standard for story points, the definition can change from project to project leading to teams having difficulties understanding them. Further, teams often find it difficult to use relative size metrics to estimate effort	LSAD	PJM1, SolAr1
Collaboration				
C9	Lack of team commitment	Lack of team commitment in terms of the general resistance to the program's way of working and the necessity to convince new members that the program's way of working is a good approach. Further, team members not participating in the estimation due to the lack of knowledge and experience, as well as teams seeing the estimation process as overhead	LSAD	SM1, PJM1, PO2, PO3, SM2
C10	Lack of knowledge about contact partners in the beginning	At the beginning, there was a lack of knowledge about contacts (e.g., experts) required for the effort estimation process		PM1
C11	Efficient communication despite spatial distribution and language barriers	Language barriers hindering the correct understanding of requirements and performing the estimation process in virtual meetings makes estimating difficult	LSAD	SM1, SolAr1, PO1
C12	Having a correct and common understanding of requirements needed for estimations	A correct and common understanding of the quality criteria and scope of the requirements, as well as the requirements themselves, is necessary for estimating the requirements. The lack of this understanding or language barriers between individuals, teams, and workstreams complicating the understanding makes estimating difficult		PO2, PM1, PO3, Dev2, PMO2
C13	Difficulty to estimate additional overhead such as meetings and explanations	Difficulty to estimate additional overhead (e.g., meetings, explanations) due to the size and complexity of the program	LSAD	Dev1
C14	Considering dependencies to other teams, workstreams, and systems in the estimation	Dependencies to other teams, workstreams, and systems make it difficult to estimate, especially when there is a lack of knowledge regarding the dependencies	LSAD	PJM1, SolAr1, Dev1, PO1, PO3, SolAr2, PM3

Expertise				
C15	Subjectivity of estimates	Estimates are based on subjective criteria such as the individual knowledge of the estimators		SM1, PM1
C16	Lack of experts involved in estimates	There is a need for expert involvement during the estimation process, but the unavailability or absence of experts on the team makes estimation difficult		SolAr1, PO1, PM1, SolAr2
C17	Lack of involvement of experts in top-level estimations	Lack of involvement of experts with the knowledge and experience in top-level estimation (e.g., roadmap planning) leads to inaccurate initial estimates		BC1
C18	Lack of knowledge and experience regarding effort estimation	Teams often lack experience and knowledge regarding effort estimation, making accurate estimation difficult		PJM1, PO1, PM1, BC1, SolAr2
C19	Neglection of relevant factors when estimating	Management neglects relevant factors such as dependencies in initial estimates or non-functional requirements. In addition, during budget planning, only the factor of the size of the organization's structure is considered, but not the complexity of the requirement and the associated effort	LSAD	SM1, Dev2, SolAr2
C20	Adjustment of effort estimates	Unfinished items have to be moved as a whole to the next iteration (e.g., Wave, Sprint), leading to the challenge that the backlog of the previous iteration is missing the effort already spent on the item and the estimate has to be adjusted in the next iteration	LSAD	SolAr2
Uncertainties and Information Deficit				
C21	Information deficit in the initial estimation of large, complex requirements	Information deficit regarding requirements (especially in the beginning) due to size, dependencies, and uncertainty make estimating difficult	LSAD, SS	Dev1, PO2, PMO2
C22	Information deficit regarding new requirements	Information deficit regarding new requirements ("Greenfield-Requirements") for which the implementation is uncertain and no comparable requirements are available		Dev1
C23	Unclear or incomplete specification of the requirements	Unclear specification and an information deficit in the functional description of a requirement makes estimating difficult		SM1, SolAr1, PO1, PMO1, SM2, SolAr2, PMO2, SolAr3, SolAr4, PM3

C24	Missing knowledge about resources in terms of people involved in the implementation	Lack of knowledge about the person(s) working on a requirement, their availability, as well as their skills in estimation makes estimating difficult	LSAD	PO3, SolAr4
C25	Unforeseen changes	Unforeseen changes in terms of system or process-related problems during implementation, ad-hoc requirements, changes in the timeline, or people leaving the program make estimating difficult	LSAD	PO1, PO3, SM2, BC1, SolAr3, BPE1

Note: Challenges that are typical of agile or large-scale agile due to agile values and principles are abbreviated with LSAD. Challenges that are typical of standard software implementation are abbreviated with SS. The column in which the classification, whether typical LSAD or SS, is abbreviated with CL

Table 4.3.: Identified Challenges in Effort Estimation - Case Study Results

5. The Mitigation Propositions

In this chapter, the identified mitigation propositions that can be implemented to address the challenges described in the previous chapter are discussed. These were elaborated from the interview data and existing literature in the context of effort estimation in agile or large-scale agile development and then assigned to the challenges. Only challenges directly addressed by the mitigation propositions were considered in this regard. Some challenges affect each other, allowing for indirect mitigation if a related challenge is directly mitigated. This indirect mitigation was not considered in the assignment. In addition, the mitigation propositions typical of agile or large-scale agile due to agile values and principles were identified and highlighted in bold in Table 5.1. To not exceed the scope of the thesis, only a few propositions mitigating the most challenges are briefly explained below.

One proposition mitigating a total of 12 challenges is **early and continuous communication between all levels: teams, workstreams, and program management (M5)**, which is evidently typical agile or large-scale agile, as communication and collaboration are essential principles of the agile methodology [35]. Early and continuous communication across all levels can reduce challenges, particularly in the categories of *collaboration* and *uncertainties and information deficit*. The interviewees *Dev1, PO1, Dev2,* and *PM3* emphasized that early and continuous communication across all levels can clarify uncertainties early and allow teams to proactively seek help in the form of experts, for example. Thus, challenges such as **unclear responsibilities (C3)**, **difficulty in having a correct and common understanding of requirements (C12)**, **unclear and incomplete specification of requirements (C23)**, and others can be overcome through early and continuous communication, not only between different teams but across all levels. *PM3* described early communication and collaboration as a proposition to mitigate the challenge **considering dependencies to other teams, workstreams, and systems in the estimation (C14)** as follows:

> *"[...] Once we know there is a dependency towards any other workstream, we are supposed to register this dependency and discuss it with the other workstream [...]. So they know at least, and we know they know. Does it make the estimation better? A bit. Yes. [...] And then it's like communication. So whenever you see something is upcoming, don't wait until the last minute. But just raise your hand. Let them know that some task is coming." (PM3)*

Bick et al. [6] mentioned that communication and collaboration across all levels help to meet the planning timeline across all levels, which can mitigate the challenge **project setting (C1)**. Additionally, experienced and qualified project managers help promote communication across all levels, thus avoiding communication problems and misunderstandings [90].

Include people with experience in estimating effort and reacting to unforeseen in the estimation process (M9) was assigned to a total of 11 challenges. By involving experts with experience in effort estimation, the time required for estimation can be reduced (C2), and difficulties with size metrics such as story points can be minimized through their knowledge and expertise (C8). Furthermore, experts have the necessary experience to make the right decisions and react appropriately when unforeseen changes occur (C25). In addition to these challenges, involving experts can mitigate others such as **lack of team commitment (C9)**, **efficient communication despite spatial distribution and language barriers (C11)**, **having a correct and common understanding of requirements needed for estimations (C12)**, **difficulty to estimate additional overhead such as meetings and explanations (C13)**, and **lack of knowledge and experience regarding effort estimation (C18)**. By involving experts in top-level estimations, the initial estimates at the requirement level can become more accurate (C17), increasing the accuracy of estimates at the work package and work item level. *BC1* describes the lack of involvement of experts in top-level estimations and Roadmap Planning (C17), from which mitigation proposition M9 can be derived, as follows:

> *"[...] if you plan something like a roadmap, you have to plan it like not on the top level. You have to involve like your professions in the lower levels, and you can avoid estimation challenges if you involve the right people. If not, and some people plan the whole roadmap without the professions, without the people with knowledge, it's very challenging." (BC1)*

Similar was found in the literature. Several researchers recommend involving all relevant stakeholders and experts in the estimation process [6, 15, 90, 95]. Mallidi et al. [52] also suggest involving the Scrum Master in the estimation process, as they believe they are responsible for communicating the information about new requirements.

The proposition **tool support to automate the estimation process (M16)** mitigates a total of nine of the identified challenges in the context of effort estimation. Using available data and tools to automate the estimation process can reduce the time of the estimation process (C2), the complexity, and the incomparability, as well as provide more transparency, as *SM1, SolAr1, PM1*, and *PMO1* mentioned. By automating the estimation process by, e.g., providing initial estimates, as suggested by the literature [3, 88], not only the process and estimates can be improved, but the challenge **lack of team commitment (C9)** due to lack of knowledge can be mitigated. Moreover, the challenge **subjectivity of estimates (C15)**, which is common in effort estimations using expert judgment as an estimation technique, can be addressed. Further, the challenges **monitoring of estimations and actual efforts (C7)**, **difficulty to estimate additional overhead such as meetings and explanations (C13)** or **considering dependencies to other teams, workstreams, and systems in the estimation (C14)** among others can be mitigated by implementing M16. If such tools are used, they should be easy to use, and the data should be integrated in a suitable way [88]. *SM1* also sees the provision of rough estimates as a helpful starting point for the process:

> *"[...] What I do, what is actually quite effective, even though I do not know what the functional work is about, I just say a number, because my colleagues who have the expertise to challenge that number are much faster. If they see a number that is too high or too low, they can tell me why it is too high or too low. So starting with a blank sheet is always hard, and I was required to have some sort of automated pre-field. If the numbers are always 50% up, it's much faster because you just correct something, but you do not have to think too much about something." (SM1)*

However, Tanveer et al. [88] point out that using initial estimates, for example, may increase the risk of the team not thinking and discussing pro-actively during the estimation, and therefore suggest using automation in the form of initial estimates more as a stimulus.

Another mitigation proposition identified is the **use of supporting techniques during the estimation process (M19)**, which is evidently typical for agile or large-scale agile projects due to the mentioned techniques, e.g., Planning Poker. Such techniques promote collaboration and continuous learning, essential principles of the agile methodology [35]. *Dev1* and *PMO2* expressed that techniques such as Planning Poker or Spike Stories can improve estimation accuracy and help to have a common understanding of each item.

> *"So at least for me, but maybe not speaking for my company, it's really just this planning poker. Yeah, I know this from research. So theoretically, I think this can be helpful to get the common understanding of more people and get the estimation closer to the reality as just the estimation of one person." (Dev1)*

However, in each case, the technique applied should be chosen based on the project size and the team's experience [52]. By using such techniques, challenges such as **difficulty to estimate in story points unit (C8)**, **lack of team commitment (C9)**, or **information deficit regarding new requirements (C22)** can be mitigated. In addition, techniques such as silent grouping, where each team member estimates items silently and independently, i.e., without prior discussion, and only those items for which there is no consensus on the estimates are discussed by the team, can be used to minimize lengthy discussions during the estimation process, thereby reducing the estimation time (C2) [68]. Generally, using such techniques for estimating leads to group discussions and knowledge sharing, allowing the whole team to learn from each other and develop a general understanding of the estimation process and estimates. Tanveer et al. [88] share a similar view, which is reflected in the results of their work that developing a common understanding is often more important than making estimates.

The only mitigation proposition that was found solely in the literature [15, 52, 88, 89] and not mentioned by the interviewees is the **use agile metrics and store estimates for improvement and future use (M21)**. Again, it is evident that this is a typical agile or large-scale agile

mitigation proposition, as applying agile metrics, as well as storing estimates, not only creates transparency but also encourages a process of continuous learning and improvement. By using metrics such as velocity, the team's activities are recorded and made available for future use. Further, by storing data obtained through these metrics, uncertain estimates can be better predicted, and the team can reflect on their estimation performance and learn from it. This proposition can mitigate several challenges, including **difficulty to estimate in story points unit (C8)**, **adjustment of effort estimates (C20)**, **information deficit in the initial estimation of large, complex requirements (C21)**, or **information deficit regarding new requirements (C22)**.

In addition, all mitigation propositions are shown in Table 5.1, including a brief description and the associated challenges. Many of these propositions mentioned by the interviewees were also found in the literature related to effort estimation in agile or large-scale agile development. This strengthens the validity of our results. Some mitigation propositions were directly mapped to challenges through the interviews or existing literature, while some assignments were based on our knowledge and expertise. In most cases, the explanation of why a proposition can mitigate a challenge is intuitive and can be inferred from the description of the respective proposition. Further, most mitigation propositions, 12 out of 21, were identified as typical for agile or large-scale agile development based on the agile values and principles [35].

ID	Name	Related Challenge	Description
M1	Adding a buffer to estimations in case of uncertainty	C21, C22, C23	Add buffer for uncertainty to the estimation (*PMO2, PM2*);
M2	Additional phase before the implementation phase to check requirements in detail (feasibility and quality)	C12, C14, C21, C22, C23	Add an additional design phase before implementation to check requirements on a detailed level and the feasibility (*SolAr1*); Similar to the propositions of adding a design phase, literature recommends conducting a pre-implementation phase with the customer to increase the success of the ERP implementation project [90]
M3	Cultivation of a dependency list	C12, C14	List all dependencies between teams, workstreams, and systems and understandings of the requirements in a dependency list (*Dev1, Dev2, PM3*);

M4	**Deep dive sessions and workshops to clarify requirements and needed resources**	C2, C3, C12, C14, C21, C22, C23, C24	Add additional meetings, workshops, or deep dive sessions to clarify the understanding of requirements and discuss resources needed in terms of people involved for implementation (*PO1, PO2, PO3, Dev2, PM2*); It would be advisable to have the customer present in such additional meetings to resolve ambiguities [96]. To reduce the number of additional meetings, research [52, 90] mentions that daily stand-up meetings could be sufficient to clarify an understanding of the requirements. Further, Tanveer et al. [88] find that developing a common understanding, for which such meetings can be constructive, is often more important than making estimates.
M5	**Early and continuous communication between all levels: teams, workstreams, and program management**	C1, C3, C10, C11, C12, C14, C16, C18, C21, C22, C23, C24	Foster communication in an early stage between all levels to clarify understandings and actively ask the program management for support and experts (*Dev1, PO1, Dev2, PM3*); Literature [6] mentions that communication and collaboration at all levels can help to address planning time frames across all hierarchical levels. To address the challenge of communication issues and misunderstanding, a skilled and qualified project manager could be helpful [90].
M6	**Events to recap on learning and document those**	C5, C8, C9, C18, C20	Use events (e.g., Sprint Review) to discuss lessons learned and document findings to improve in the next iteration (e.g., Sprint) (*BC1*); This is also recommended in the literature in terms of feedback sessions [3]
M7	Guidelines and standard estimates for tasks uniform across all teams	C2, C8, C12, C15, C18	Provide more guidance and establish a standard for estimation components that are uniform across all teams (e.g., documentation, testing)(*PM1, Dev2*); Tanveer et al. [88] propose to define standards, e.g., for the complexity or story size to improve estimates

M8	Consider additional effort regarding organizational and process factors during estimation	C12, C19	Consider not only effort directly associated with the implementation but also from an organizational and process perspective such as communication costs or resources needed for the implementation during the estimation (*PMO2*); In addition, the research [3, 88, 89, 96, 95] recommends considering factors that influence the effort estimation process, such as the implementation experience of the developer, candidates with a high degree of optimism, or the estimation knowledge of team members. Further, all development activities, including, e.g., non-functional requirements that require significant effort or testing effort, should be considered in the estimates [96]
M9	Include people with experience in estimating effort and reacting to unforeseen in the estimation process	C2, C8, C9, C11, C12, C13, C16, C17, C18, C22, C25	Include experienced people in the team who know how to estimate, react to unforeseen, and moderate the estimation efficiently and well and involve experts in top-level estimations (e.g., roadmap planning) (*Dev1, PM1, PO3, BC1, SolAr2*); Similar is suggested by the literature: All concerned stakeholders should be involved in the effort estimation process [15, 90, 95] and all planning activities should gather and incorporate feedback from experts [6]. Especially the Scrum Master should be involved in the effort estimation process moderating the team and providing the relevant information on new requirements [52]. In case experts are not available, then Hill [36] proposes to compare pending requirements with a collection of already estimated requirements
M10	Measures to convince the team that effort estimation is a group/team activity	C9, C18	Introduce measures that convince the team that effort estimation is a group activity regardless the individual knowledge (*PJM1*);
M11	Normalization of story points (to person days)	C8, C9	Normalize the size metric story points to person days to assure that everyone has a reference and understanding to the size metric (*PJM1*); Normalizing story points in person days is one option to overcome difficulties when estimating in story points, but generally it should be ensured that there is a clear and standardized definition of story points within the project [30]

M12	**Plan requirements for an appropriate time frame to improve the accuracy of estimates**	C1, C20, C24	Plan requirements for an appropriate time frame (e.g., not too far in advance) to improve the accuracy of estimates (*PO3, PM3*); However, research recommends to conduct planning activities, such as effort estimation, with sufficient lead time to identify certain factors, such as dependencies, at an early stage [6, 7].
M13	**Platforms and meetings to identify and discuss dependencies**	C14, C19	Include meetings and communication platforms to discuss and identify dependencies (*PJM1, SolAr1, Dev1, SolAr2*), e.g., cross-team planning workshops or unit-wide retrospectives [7, 6]
M14	**Reduce time tracking pressure for employees**	C4	Consider allocating more time for networking among employees so that they are more open to honest time recording (*Dev1*);
M15	**Support and motivation by Scrum Masters and Agile Coaches**	C8, C9, C11, C13, C18, C20, C22, C23, C25	Scrum Masters and Agile Coaches to underline the importance of the estimation process, keep the team motivated and support the estimation process (*PJM1, PMO1*); Mallidi et al. [52] recommend a training program for the team to familiarize the team with the agile way of working and the use of relative estimation metrics, e.g., story points. In addition, it could be helpful to empower the team in their self-organized nature, to be creative and innovative, and to acknowledge their expertise [52].
M16	Tool support to automate the estimation process	C2, C6, C7, C9, C13, C14, C15, C18, C23	Use available data and tool support to automate the estimation process to reduce time, complexity, and incomparability and provide more transparency (*SM1, SolAr1, PM1, PMO1*); In this case, it is essential that the tool is easy to use and data is integrated in a proper way [88]. Research [3, 88] mentions that analogies with similar projects and providing initial estimates can also improve the process and estimates
M17	**Tool/feature to support the estimation process**	C2, C6, C7, C8, C9, C11, C12, C14, C15	Add features (e.g., Post-its, Scrum Boards) to the tool to support the estimation process (*PM1*); By adding features that visualize aspects from, e.g., impact analysis or proposed estimates, the challenge of subjectivity can be mitigated and the estimation process, as well as risk management, can be supported [88, 89]

M18	**Tracking of actual efforts**	C5, C7, C15	Track the actual effort to understand reasons for shifting requirements to next iteration (e.g., Sprint) and improve estimates in the future (*PM1, PMO1*); In addition, using accuracy metrics, such as actual effort, can improve the overall planning process, as well as the development of shared understanding [88]
M19	**Use of supporting techniques during the estimation process**	C2, C8, C9, C11, C12, C15, C18, C21, C22	Using techniques (e.g., Planning Poker, Spike Stories) to improve estimate accuracy and question their accuracy based on uncertainty and understanding (*SM1, Dev1, PMO2*); In this case, the techniques should be selected based on the project size and experience of the team [52]. Techniques like silent grouping will minimize lengthy discussions, which in turn reduces the time spent on estimating [68] or applying impact analysis helping to identify the impact of implementing a requirement on the existing system can lead to an improvement of the estimation process [88, 89]
M20	**Use T-Shirt size as an estimation unit to avoid difficulties to estimate in story points and trust the team more**	C4, C8, C9	Use the size metric T-Shirt size to prevent difficulties in estimation, to assure that everyone has a reference and understanding to the size metric, and to give trust to the team (*SolAr1*); Research [30] proposes to use well defined, objective size measurement to avoid misunderstanding of story points and assure that everyone understanding to the size metric [30]
M21	**Use agile metrics and store estimates for improvement and future use**	C5, C7, C8, C9, C18, C20, C21, C22, C25	Several researchers [15, 52, 88, 89] recommend to use agile metrics, e.g., velocity, to record activities for further use and predict uncertain estimates better, as well as store estimates to track, learn, and reflect the estimation performance

Note: Mitigation propositions that are typical of agile or large-scale agile due to agile values and principles are highlighted in bold

Table 5.1.: Mitigation Propositions to address the identified Challenges in Effort Estimation - Results from Case Study Results and Existing Literature

6. Artifact Evaluation

6.1. Methodology

The objectives of the evaluation are to collect qualitative feedback and improvement suggestions from practitioners to refine the artifact further, as well as to assess the artifact in terms of its practical relevance. In the case of this thesis, the artifact developed is a collection of identified challenges and mitigation propositions. To achieve these goals, three semi-structured interviews were performed with selected participants from the initial interviews. In addition, a survey was conducted, in which the 17 remaining participants from the initial interviews could evaluate the artifact and provide their qualitative feedback online. In total, nine experts participated in the online survey. Table 6.1 provides an overview of the experts who participated in the evaluation, as well as whether they participated in an interview or an online survey. The information provided by the online survey participants about their role and experience in large-scale agile IT projects may differ from those in the initial interviews due to reasons such as internal role changes or the fact that they had more time to reflect on their experience.

No.	**Role**	**Alias**	**Experience in agile software development (years)**	**Interview (Duration h:m) \Online-Survey**
1	Project Manager	PJM1	6 - 10	Interview (0:58)
2	Solution Architect	SolAr1	3 - 5	Interview (0:51)
3	PMO	PMO1	6 - 10	Interview (0:36)
4	Scrum Master	SM1	6 - 10	Online Survey
5	Agile/Scrum Coach	AC1	6 - 10	Online Survey
6	Product Owner	PO1	3 - 5	Online Survey
7	Product Owner	PO2	3 - 5	Online Survey
8	Solution Architect	SolAr2	3 - 5	Online Survey
9	Solution Architect	SolAr3	11 - 15	Online Survey
10	Product Owner	PO3	3 - 5	Online Survey
11	Product Manager	PM1	16 - 20	Online Survey
12	Agile/Scrum Coach	AC2	11 - 15	Online Survey

Table 6.1.: Overview of the evaluation participants

The online survey was divided into three parts, with the first part being a shortened version of the General Information section of the initial interviews. Respondents were asked to

provide information about their role and experience with large-scale agile development. This additional inquiry was necessary because the online survey was conducted anonymously. The first part of the General Information was not included in the semi-structured interviews, as this information had already been obtained through the initial interviews. The second and third parts of the online evaluation were the same as the first and second parts of the interview evaluation. The only difference is that the nature of the semi-structured interviews allowed for a more natural and informative conversation. Thus more detailed feedback could be obtained compared to the online survey. However, since the content of both evaluation methods was the same, the results do not need to be distinguished further and will be treated as equal in the analysis throughout the sections of this chapter. The second part of the online evaluation and the first part of the interview evaluation involved assessing the identified challenges. While the last part of the evaluation involved assessing the mitigation propositions. In both cases, respondents were asked to agree or disagree with the identified challenges and mitigation propositions using a five-point Likert scale [49] with the response options *strongly agree* (1), *agree* (2), *neither agree nor disagree* (3), *disagree* (4), and *strongly disagree* (5). The online survey contained a text box where respondents could provide qualitative feedback or improvement suggestions in addition to each challenge and mitigation proposition. In the semi-structured interviews, feedback naturally emerged during the conversation. The detailed questionnaire of the online survey, which was used as a basis for the structure of the semi-structured interviews (excluding the General Information section), can be viewed in Appendix B.2. Subsequently, the collected data from the evaluation was coded and analyzed. After transcribing the interviews, the transcripts were coded in a two-cycle approach using a combination of deductive and inductive coding according to the guidelines of Miles et al. [53] and Saldaña [72]. Similar to the coding procedure of the initial interviews, descriptive codes were assigned to relevant text segments of the feedback [53] in the first cycle. In the second cycle, the higher codes of the second cycle were mapped to the codes of the first cycle, grouping them inductively by applying pattern codes for recurring concepts [53, 72]. Once new codes were assigned for the second cycle, the previously coded data had to be re-coded. A similar procedure was followed with the qualitative feedback from the online survey, which did not need to be transcribed before coding. The results of the analysis are presented in detail in the following sections, with the results of the evaluation of the challenges discussed first, followed by the results of the mitigation proposition.

6.2. Evaluation

6.2.1. Evaluation of the Challenges

As part of the evaluation, practitioners are to assess the relevance of the identified challenges and agree or disagree on whether these challenges represent a challenge in the context of effort estimation within the case organization. The evaluation results may vary, as the ratings are based on individual opinions, which are additionally influenced by their experience in large-scale agile development, specifically in effort estimation, their role within the program,

and the level at which they operate. The statistical evaluation of the challenges is presented in Figure 6.1. To avoid exceeding the scope of this thesis, only selected challenges will be discussed in more detail. More specifically, those where opinions differed and there was no unanimity, as these differing views are particularly interesting.

C2: Time restriction
In challenge C2, opinions were divided. While about 58% of respondents agreed or strongly agreed that time restriction is generally a problem in the context of effort estimation, a third disagreed. Although more than half of the respondents at least agreed, the standard deviation is 1.16, which is because the third who disagreed with the challenge increased the variance and thus the standard deviation accordingly. Overall, the average evaluation on C2 is 2.58, and the median is 2, corresponding to the response option "agree." Among the respondents, it was emphasized that identifying dependencies on other teams or workstreams is very time-consuming and often challenging for teams due to the time limitations (*SolAr1*). Two of the four respondents who disagreed argued that time is always a problem because time is generally limited. Still, the purpose of the effort estimation process is to exchange information about the respective requirements to understand all perspectives on the topic and avoid misinterpretations or problems in the later implementation (*PJM1, PM1*).

> *"[...] Sometimes there is feedback from the teams that this estimating process or the entire agile elements at all are blocking them from the real work. But, then we always try to give them the understanding that this is also part of this real work because the effort estimation itself, from my perspective, just doesn't have only the purpose to have a certain value of story points. But more important, even is the exchange on the topic, the different views you get in this process." (PJM1)*

In addition, the time spent on the estimation process is a small part compared to the total time available for implementing and realizing the requirements (*PJM1*).

C5: Lack of measures to improve estimations
The statistical evaluation of challenge C5 shows a standard deviation of 1.21 with an average and a median of 3, which corresponds to a "neither agree nor disagree" response. While C5 was evaluated by one respondent with "strongly agree" and one with "disagree," four agreed or disagreed with C5 being a challenge in the case organization. *SolAr1* argues that metrics or measures to measure the planned and actual efforts are constructive to improve the estimation performance for future estimations and strongly agrees. *SolAr1* sees the potential to make accurate estimates in the future not fully realized due to the lack of these metrics or measures. While *PJM1* argues that *"record the effort and assign them to a certain work package, I think that's always difficult. It means a lot of administrative additional work, which is usually, I've seen it in another project, it's hardly accepted by the teams. There is the risk that [...] people tend to record times according to what is expected to be recorded because, especially when you work on several work packages, you have to assign your records and to certain work packages. I think not many do it that*

accurately [...]" and therefore strongly disagrees on C5 being a challenge in the context of effort estimation. *PMO1* also disagrees, as based on observations within the program, estimates' reliability and quality have improved over time. However, *PMO1* further mentions that the assessment, in this case, is context-dependent, and C5 could be a challenge in a different context.

C6: Inappropriate tool support
The statistical evaluation results in a standard deviation of 1.48 and an average and median of 3.0. Based on the comment, "Fancy tools like [*] [5]" (*PO3*), it can be assumed that the respondent wanted to choose "strongly agree" instead of "strongly disagree," which would change the statistical evaluation accordingly. However, *PJM1* argues that the main tool of the program provided by the software company, includes additional functionalities such as transport management or process documentation, which are not relevant for the documentation of agile activities. Therefore, *PJM1* justifies his evaluation of "neither agree nor disagree" by stating that there may be better tools for documenting and tracking purely agile activities, but for the end-to-end process of the program, this tool is the best available solution. *PMO1* shares a similar opinion and disagrees that C6 is a challenge. On the other hand, *SolAr2* strongly agrees that inappropriate tool support is a challenge and recommends using separate tools *"one for steering the agile teams and a separate one for the budget planning and reporting"*.

C8: Difficulty to estimate in story points unit
The evaluation results on C8 show an average and median of 3.08 and 3.0, respectively, corresponding to the response option "neither agree nor disagree." The results yield a standard deviation of 1.16. Among all responses, "disagree" is most commonly chosen for evaluation C8, with one-third of the responses, while the other assessments are mainly distributed between "neither agree nor disagree" and "agree." One of the respondents who disagreed on difficulties in estimating with story points argued that it was decided very early in the program to normalize story points to person days, which gave everyone in the program a reference and understanding of the size unit. Therefore there were no challenges in this context for effort estimation (*PMO1*). However, *PMO1* also mentioned that the normalization is not an agile best practice, but that in a program with more than 300 people involved, it was the easiest way to use a size unit that everyone, regardless of their experience in estimating, could use to estimate. *AC1* shares the same opinion and also disagrees on C8 being an challenge. *PJM1* mentioned that because they initially observed this challenge in all teams within this program, it was decided to normalize story points and agreed that C8 is a challenge in effort estimation:

> *"That was also the reason why we decided to somehow set story points as an equivalent to person days because we also saw this or had this challenge [...]. I think we managed it somehow, but it's still, or it's definitely a topic where it's worth investing in a common definition at the very beginning and also in the communication or somehow distributing*

[5] * is a placeholder to ensure that sensitive data is protected. In this case * is a placeholder for the name of the tool provided by the software company

information to the entire team and making sure that everybody, really everybody, gets it. And this is, of course, a challenge. Absolutely."

C18: Lack of knowledge and experience regarding effort estimation
The evaluation of C18 reveals some inconsistencies. 50% see C18 as not being a challenge within the program. In contrast, 50% at least agree that a lack of knowledge and experience regarding effort estimation in the team leads to challenges in estimating. Therefore, the median is in the middle, at 3.0, while the average is 2.92 due to the one "strongly agree" assessment. Respondents argue that there was a great variance in experience within the teams and the program itself. Further, it was observed that the teams' estimates were good, resulting in C18 not being a challenge that practitioners faced within the case organization (*PJM1, PMO1*). Particularly striking was the distribution of the respective assessments of whether C18 is a challenge practitioners face during effort estimation. All respondents, except one, who disagreed, had roles that were not directly part of a Scrum team, such as people from program leadership, Scrum Masters, or Agile Coaches. While the respondents who agree that C18 is a challenge during estimation all have roles within the Scrum teams, such as Product Owner or Solution Architect.

C24: Missing knowledge about resources in terms of people involved in the implementation
The distribution of the evaluation for C24 shows an average of 3.08 and a median of 3.0, resulting in a standard deviation of 1.08. What is generally interesting about challenge C24 is that it arises due to the normalization of story points. The idea behind relative estimation measures such as story points is that items are estimated based on their complexity, degree of difficulty, and scope rather than individual factors such as the skills or experience of the individuals involved. This is because a developer in the field for more than 15 years can implement an item faster than one with only one year of experience. Certainly, estimates are influenced by such individual factors but should not be estimated based on these. *SolAr1* precisely describes this difficulty:

"From my perspective, the effort estimation should not be done on the availability of an expert or on the one who is responsible for that. [...] If I know that, let's say, a developer or someone who has a lot of experience then it might be less effort [...]. So the effort estimation should not be done based on the one who has knowledge or what kind of knowledge the person has. So from my perspective, I would disagree."

This aspect is not decisive whether C24 is a challenge because the story points were normalized to person days in the program, which leads to the estimation being based on individual factors such as the availability or skills of the persons. Suppose this information about the persons involved in the implementation is not available during the estimation. In that case, challenges arise, making effort estimation difficult. This explains why nearly 41% agree on C24 being a challenge that practitioners face during effort estimation.

Regarding the remaining challenges not discussed in detail above, the experts mostly agreed with some minor exceptions, which affected the standard deviation accordingly. A detailed analysis of the quantitative results, more specifically, the distribution of the answers, is shown in Appendix C.2.

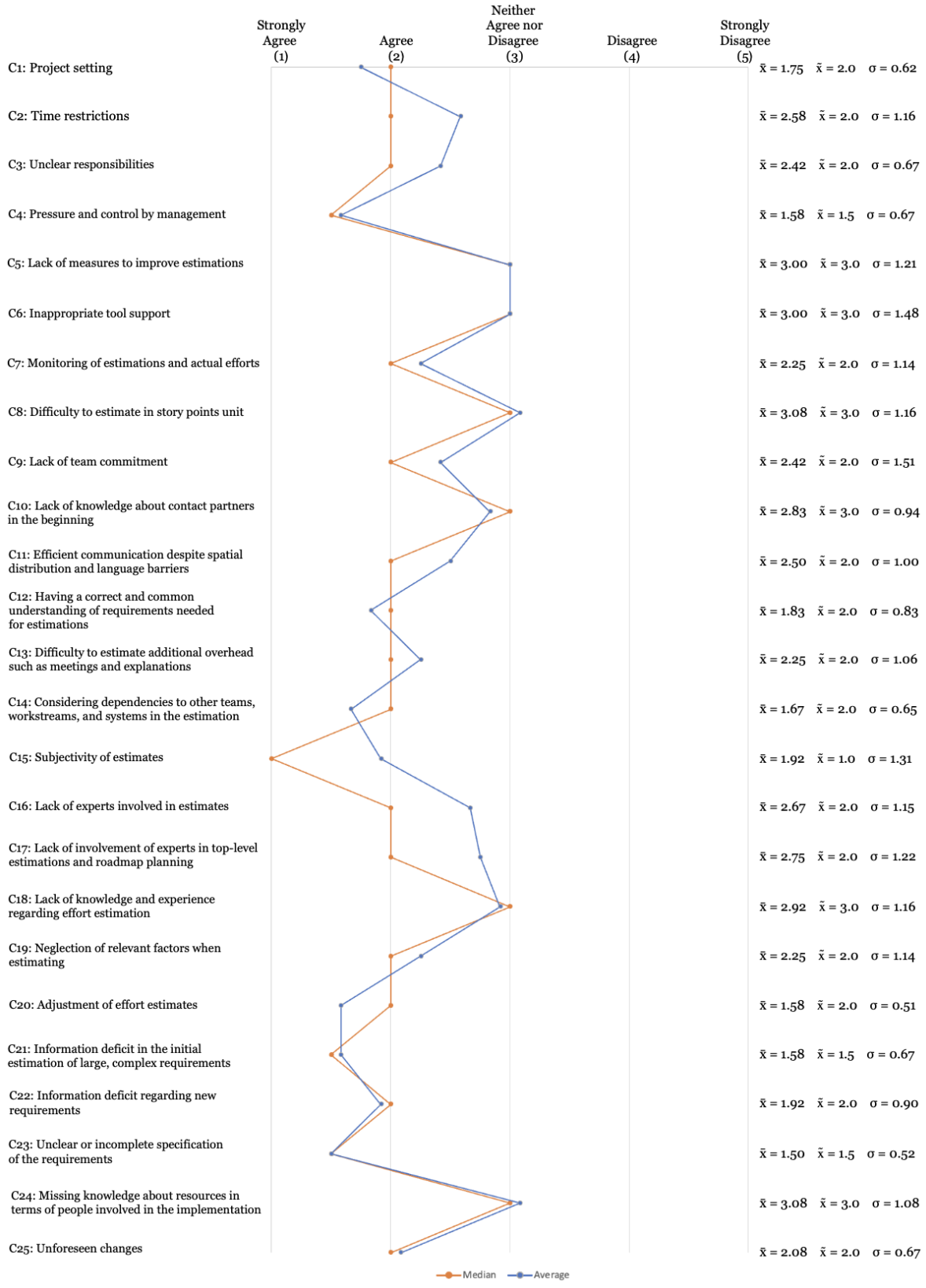

Figure 6.1.: Evaluation Results of the identified Challenges

6.2.2. Evaluation of the Mitigation Propositions

As part of evaluating the mitigation propositions, practitioners are to assess the relevance of theses and agree or disagree to what extent these can mitigate challenges in the context of effort estimation, based on their opinion. To keep the scope of the interviews and online surveys as small as possible while still ensuring a high level of quality, it was decided not to explicitly list the specific challenges the presented propositions can mitigate. We justify this decision by stating that the assignment of challenges to the respective mitigation propositions should be especially intuitive for practitioners in large-scale agile development, particularly in effort estimation. However, the mitigation propositions were identified through the data collected from the initial interviews and existing literature in this context. As with the challenges, there may be variances in the results of the evaluation of the mitigation propositions, as ratings are based on individual opinions that are influenced by factors such as experience in the context of effort estimation in large-scale agile development, the individual's role, and the level at which they operate. The statistical evaluation of the mitigation propositions is presented in Figure 6.2. In the following, only the mitigation propositions where opinions differed and there was no unanimity will be discussed in more detail.

M2: Additional phase before the implementation phase to check requirements in detail (feasibility and quality)
The mitigation proposition M2 shows the highest standard deviation of 1.5 in the evaluation results. The median has a value of 2.5, and the average rating is 2.92. 6 out of 12 respondents (strongly) agree that M2 is a proposition that can mitigate challenges related to effort estimation. Three respondents mentioned that there was such an additional phase before the program officially started *(PJM1, SolAr1, AC2)*. *SolAr1* emphasizes that there was half a year to discuss requirements in detail and therefore *"be able to do a better effort estimation"* and strongly agrees that M2 is a proposition mitigating challenges in effort estimation. *PMO1* states that such a phase, called the design and delivery phase, took place in a rather integrated way, where the design and implementation phases overlapped, and therefore neither agrees nor disagrees. Further, *PMO1* warns that *"if you spend too much time over talking about design and not going forward, then you can also lose a lot of time in detailing things out. And once you start implementing with a system integrator, they still don't understand it because they were maybe not part of the phase before."* In summary, three experts strongly disagreed that an additional phase where requirements are discussed in detail helps mitigate challenges and obtain more accurate estimates. They believe it does not lead to any further benefit and prefer to continue planning at the highest level, focusing on development *AC1, PO2, PO3*.

M10: Measures to convince the team that effort estimation is a group/team activity
The results of the evaluation of M10 show an average of 2.67 and a median of 2.5, indicating an average response between agree and neither agree nor disagree, with a slight tendency towards the latter. Overall, 6 out of the 12 respondents at least agreed that taking measures to convince teams that the estimation process is a group activity, independent of the experience and knowledge of individual team members, would be helpful. While one person strongly

disagreed and one disagreed, it was noticed during the analysis of the results that some comments from the online survey indicated that the description of M10 was not entirely clear. It is likely that measures were interpreted as measurements, which may have caused misunderstandings. One such comment is:

> *"Everyone will be concentrated on estimated figures and not on development and will search for any mitigation if the estimation is not met" (PO3)*

However, *PJM1* mentions that a possible measure could be that the Scrum Master could remind the teams that the opinion of each team member is valuable for the estimation and motivates the members who do not actively participate for reasons like too little experience or knowledge.

M18: Tracking of actual efforts
During the analysis of the evaluation of mitigation proposition M18, opinions are divided regarding tracking actual efforts. Overall, 50% of the respondents agree that this proposition mitigates challenges in the context of effort estimation. According to *SolAr1*, the actual effort can help to improve future estimates and agrees with this proposition. Additionally, *SolAr1* expresses concern that tracking actuals at a too detailed level would be rejected by some team members. These concerns are confirmed by those who voted against M18. On the one hand, it is mentioned that it would mean too much additional overhead *AC2*, and tracking actuals would lead to more discontent among employees *AC1*. On the other hand, actual efforts do not reflect the reason for shifting requirements *PJM1, PO3*. The average response of all respondents is 2.92, and the median is 2.5, indicating an average response of rather "neither agree nor disagree" while the standard deviation is 1.08

M19: Use of supporting techniques during the estimation process
The results of the evaluation of M19 show a median of 3 and an average of 2.92, indicating an average response of "neither agree nor disagree," with the distribution of response options of "strongly agree," "agree," "disagree," and "strongly disagree" similar to that of M18. Concerns about M19 being a proposition mitigating challenges by using techniques such as Planning Poker to achieve more accurate estimates are expressed in the direction that Planning Poker was used at the beginning, but it did not add any additional value *AC1, AC2*. However, why such techniques could not add value differ, *AC1* suspects that the program setting does not allow it, while *AC2* believes that the lack of time is the biggest problem. Nevertheless, 42% agree that applying techniques can mitigate challenges faced during the effort estimation. In the context of the evaluation of challenge C15, subjectivity of estimates, *PMO1* mentions:

> *"Mitigation is then to get more than one person thinking about an estimate and planning poker, these kind of things and discussing in the team. To objectify [the subjectivity] a bit."*

M20: Use T-Shirt size as an estimation unit to avoid difficulties to estimate in story points and trust the team more
Among all mitigation propositions, M20 has the strongest tendency to the response option "disagree" with an average of 3.25, a median of 3, and a standard deviation of 1.06. While one-third of the respondents see T-Shirt size as a proposition that can mitigate challenges in effort estimation, a quarter holds a different opinion. Among those who agreed, however, one pointed out that while using T-Shirt size could simplify the estimation process for team members, it would make tracking actual efforts more difficult (*SolAr1*). Additionally, one who agreed mentioned that using T-Shirt size instead of story points is particularly suitable in an early phase, i.e., for higher-level estimates *PMO1*. *PJM1* shares the same view, but disagrees due to the imprecise nature of T-Shirt size, the additional work required to relate the respective estimates to budget resources, and the fact that everyone has a different understanding of what, e.g., a T-Shirt size S means. On the other hand, *SolAr2* notes that the T-Shirt size is very similar to Story Points if they are not normalized, which somewhat weakens the concerns about understanding what, for example, a t-shirt size S means.

As with the challenges, the experts were essentially in agreement, with a few minor exceptions regarding the remaining mitigation proposals not discussed in detail above. A detailed analysis of the quantitative results, more specifically, the distribution of the answers, is shown in Appendix C.2.

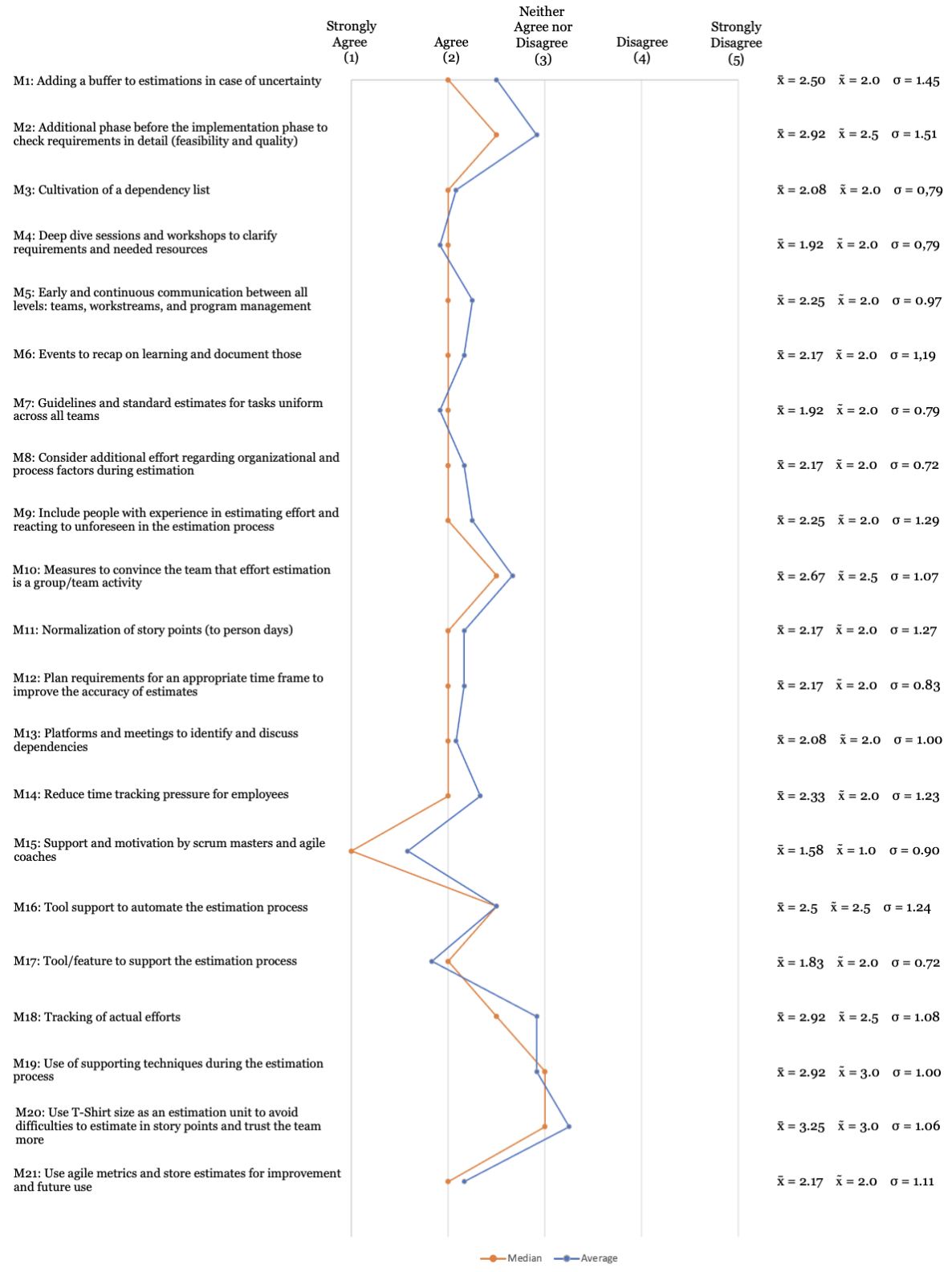

Figure 6.2.: Evaluation Results of the identified Mitigation Propositions

6.3. Adjustments based on Evaluation

As part of the evaluation, practitioners provided feedback and improvement suggestions. These are incorporated into the artifact to refine it further, resulting in the final artifact of this thesis (Version B), which can be found in Appendix D. As mentioned above, the feedback and improvement suggestions from the three expert interviewees were more extensive than those from the online surveys, which can be attributed to natural conversations taking place during the interviews.

In the following, the adjustments to the artifact will be discussed.

During the analysis of the evaluation results, it was observed that incorporating the feedback and improvement suggestions led to three different adjustments, which we divided into three categories. The first category includes adjustments in which further challenges are assigned to mitigation propositions not considered in the first version of the artifact. In some cases, this new assignment leads to further changes in the description of the respective mitigation proposition. These adjustments were assigned to the category *adjustments to the challenges and mitigation propositions*. Adjustments that affect only the challenge or mitigation proposition are assigned to the second and third categories.

Adjustments to the Challenges and Mitigation Propositions

In the feedback from practitioners, it was mentioned that the challenge **project setting (C1)** was attempted to be mitigated in advance through an additional design phase before the implementation phase, which is why we added C1 as a related challenge to the mitigation proposition **additional phase before implementation phase to check requirements in detail (feasibility and quality) (M2)**. Additionally, it was observed that the challenge **unclear responsibilities (C3)** can be mitigated by the proposition **support and motivation by Scrum Master and agile coaches (M15)**, according to the practitioners. By adding C3 as a related challenge to M15, the description of M15 was adjusted with the following addition: *Further, provide a description of each role so that it is clear who is responsible for what*. Another adjustment we made is assigning the challenge **adjustment of effort estimates (C20)** to the mitigation proposition **tool/feature to support the estimation process (M17)**. It was mentioned multiple times as an improvement suggestion to use two different tools for the operations of agile activities and budget planning because the primary tool used in the case organization does not provide sufficient support for agile activities. The lack of functionality often leads to challenges such as the C20. Based on this adjustment, we added the following sentence to the description of M17: *Make sure to use sufficient tools for the agile activities*. The last two assignments we adjusted are the challenge **difficulty to estimate additional overhead such as meetings or explanations (C13)** and the challenge **unforeseen changes (C25)** to the mitigation proposition **add an additional buffer to estimations in case of uncertainty (M1)**. In both cases, feedback was received that attempts were made in the program to mitigate these challenges by advising teams to add additional buffers in case of potential uncertainties

affecting the estimates. All of these adjustments are in the final version of the artifact of this thesis, which is shown in Appendix D.

Adjustments to the Challenges

Based on the feedback and improvement suggestions of the experts, changes have been made to the descriptions of four challenges in total. During the evaluation of the challenge **monitoring of estimates and actual efforts (C7)**, several experts mentioned that not enough time is spent on accurately breaking down the objects to be estimated, which makes monitoring of actual efforts difficult. Therefore, we have adjusted the aspect *"wrong granularity of estimation objects"* in the description to *"the little time invested in the breakdown of estimation objects leading to wrong granularity"*. Regarding the challenge **lack of team commitment (C9)**, practitioners mentioned that even experienced team members often have different understandings of what agile is, and this understanding often does not match the program's ways of working. Therefore, we added the following sentence to the description of C9: *"Also, team members with experience mostly have different understandings of what is agile, often not matching the specific program's way of working"*. During the evaluation of the challenge **lack of involvement of experts in top-level estimations (C17)**, feedback from respondents indicated that the absence of experts in top-level estimations did not necessarily lead to inaccurate initial estimates. But it was mentioned that it increased the risk of inaccurate top-level estimates. Therefore, we adjusted the aspect *"leads to inaccurate initial estimates"* in the description to *"increases the risk of inaccurate initial estimates"*. During the evaluation of the challenge **missing knowledge about resources in terms of people involved in the implementation (C24)**, feedback indicated that estimates should not be based on factors such as the availability or skills of people involved in the implementation, but rather based on the complexity and scope of the object to be estimated. However, in the case organization, story points were normalized to person days, so estimates were based on such factors. Therefore, we adjusted the description of C24 as follows: *"As long as the estimation unit is based on person-days, a lack of knowledge about the person(s) working on a requirement, their availability, as well as their skills in estimation makes estimating difficult."* All of these adjustments are in the final version of the artifact of this thesis, which is shown in Appendix D.

Adjustments to the Mitigation Propositions

Adjustments were made to the seven mitigation proposition based on the experts' feedback and improvement suggestions. During the evaluation, concerns were raised regarding the mitigation proposition **adding a buffer to estimation in case of uncertainty (M1)**. These concerns relate to the risk that it may not be possible to trace which team has added a buffer to their estimates and which has not. Furthermore, the amount of buffer added to the respective estimates is not recognizable. To minimize this risk, we suggest adding the following sentence to the description of the mitigation proposition M1: *"Additionally, it would be helpful to provide a standard or guideline that includes explanations of specific uncertainties, as well as their magnitudes to be considered in estimation, to avoid inconsistencies between different teams"*.

In the mitigation proposition **events to recap on learning and document those (M6)**, it was noted that employees already have many meetings scheduled in their calendars. Experts suggest that measures should be taken to keep the number of meetings to a minimum to ensure enough time to work on implementing the items. Therefore, we add the following sentence to the description of M6: *"However, make sure that the number of meetings remains reasonable so that the teams have enough time to implement their work"*. Two adjustments were made to the mitigation proposition **including people with experience in estimating effort and reacting to unforeseen events in the estimation process (M9)**, based on the feedback from the experts. It was mentioned that involving external experts not part of the program is often helpful to challenge the estimates and gain further insights. Therefore, we have added the following sentence to the description of the mitigation proposition M9: *"Further, it is beneficial to involve experts not involved in the program to challenge the estimates and gain further insight from them"*. Additionally, it was mentioned that in the program, special efforts were made to mitigate language barriers by involving people with the necessary technical and business knowledge while also being fluent in both the native language of the company and at least English. This is helpful to clarify any misunderstandings during estimation due to language barriers and share a common understanding of the items. Therefore, we added another sentence to the description of this mitigation proposition to address this issue: *"In case of language barriers, make sure to include people with the business/technological logic and language skills to avoid misunderstandings and difficulties during estimation"*. As mentioned in the previous section, it is assumed that there has been a misinterpretation of the word "measure" in the online survey for the mitigation proposition **measures to convince the team that effort estimation is a group/team activity (M10)**. This possible misinterpretation is due to the fact that English is not the native language of all respondents. Since the meaning is correct in the context of M10 and there is no other objection to the proposition, we have decided not to make any adjustments in this regard. However, it should be noted that with "measures," we do not mean measurements but actions that convince the team that the estimation process is a group activity. During the evaluation of the mitigation proposition **reduce time tracking pressure for employees (M14)**, it was mentioned several times that employees generally feel pressured to meet certain expectations from management within a specific time frame. To generalize M14 and not just limit it to the topic of time tracking, especially since many programs do not require tracking of work hours, both the name and description have been adjusted as follows:

> **M14: Reduce pressure for employees**
> Reduce pressure for employees, e.g., by allocating more time for networking so that they are more open to honest time recording or not setting management expectations too high for teams to implement a certain amount of requirements per iteration (e.g., Sprint).

During the evaluation of the mitigation proposition **tool support to automate the estimation process (M16)**, concerns were raised that automating initial estimates for estimation objects

can lead to teams adopting the initial estimates without discussing them, resulting in a lack of shared understanding for each item. Therefore, it was decided to address these concerns in the description of M16 by adding the following sentence: *"In case of providing initial estimates, try to avoid that teams adopt the initial estimates without discussing the requirements"*. Apart from the difficulty of tracking actual effort in such large programs, it was repeatedly expressed during the evaluation of the mitigation proposition **tracking of actual efforts (M18)** that there is a risk that teams may feel controlled and reject this metric as long as the program leadership tracks it. Additionally, tracking actuals could lead to the problem of employees manipulating the numbers to meet the management's expectations. However, experts see the potential of improving the estimation performance by tracking actuals as long as it is done within the team. Therefore, the description of M18 was adjusted as follows: "Track the actual effort to understand reasons for shifting requirements to the next iteration (e.g., Sprint) and improve the estimation performance in future estimates (PM1, PMO1); In addition, using accuracy metrics, such as actual effort, can improve the overall planning process, as well as the development of common understanding [88]. Tracking actuals on a higher level, e.g., management level, may increase the risk of teams feeling controlled or pressured and thus reject the tracking of actuals. Therefore consider tracking the actuals on a team level". The last adjustment made to the artifact was an addition of a sentence to the description of the mitigation proposition **use T-shirt size as an estimation unit to avoid difficulties to estimate in story points and trust the team more" (M20)**. During the evaluation, it was expressed that T-shirt size is more suitable for high-level estimations rather than low-level estimations, as the estimation with T-shirt size is too imprecise for effort estimates on the sprint level. Therefore, the description of M20 was supplemented with the following sentence: *"This is most useful for high-level planning estimates"*. All of these adjustments are in the final version of the artifact of this thesis, which is shown in Appendix D.

7. Discussion

7.1. Key Findings

Effort estimation is performed on three different levels
In the case organization, we have observed that the effort estimation process is conducted at three levels. The basis for the items to be estimated is derived from a so-called fit-to-standard workshop, where the desired requirements are discussed and compiled together with the customer. The initial estimation of these requirements takes place at the highest level before the realization phase during the so-called Roadmap Planning. In this estimation, selected individuals provide rough estimates of the requirements, which are then documented. The entire team then conducts the estimation process at the intermediate level before each three-month iteration, the so-called Wave. Before estimating the requirements more accurately on the Wave level, these are broken down into smaller parts, referred to as work packages. The estimation of the work packages is based on the initial estimations of the respective requirements, which are further refined and adjusted if necessary. At the lowest level, the entire team estimates the work items before each four-week iteration, the so-called Sprint. The work items are the result of breaking down the corresponding work packages. And again, the estimates of the higher-level items, the work packages, serve as the basis for the estimates at the lowest level, which are refined and adjusted further if necessary. The accuracy of the estimates increases from level to level resulting in the work item level being estimated the most accurately. This is because, at this point, the most information and a clearer understanding of how the items will be implemented and realized are known.

Refinement meetings are used to adjust the estimates and prioritization
As mentioned above, it has been observed that the accuracy of the estimations, particularly at the highest level, is initially not very high. The case organization does not consider this to be critical, as refinement meetings are intended to be used to adjust the estimates at a later stage in case of inaccuracy. The program leadership has mandated that these refinement meetings are held at least once per Sprint, lasting 6 hours each. These meetings ensure that the estimates become increasingly accurate and largely mitigate significant disadvantages, such as project failure, resulting from inaccurate estimations. Over/underestimations are not retrospectively adjusted for the initial estimates, but only for the currently estimated or processed objects, i.e., the initial estimates on the requirement level are not adjusted accordingly, but only those on the work package or work item level. Additionally, these meetings ensured that changes are reacted to quickly, that influencing factors such as dependencies, over/underestimation, and sudden events affecting the system are not neglected, and that the Product Backlog is continuously adjusted. Such unforeseen or influencing factors may require

adjusting the objects' prioritization order, which is also carried out during these refinement meetings.

Often opinion leaders or highly skilled experts decide on the estimates instead the whole team
In this thesis, it has been emphasized multiple times how important it is in the practice of agile methodologies that the entire team is involved in the estimation process, regardless of their technical knowledge and skills. However, in the case organization, it has been observed that, in order to expedite the estimation process, either the high-skilled experts perform the estimations without involving the team or take on the role of opinion leaders. Consequently, estimations are often recorded based on their assessment without further discussion within the team regarding the estimation objects. The purpose of the estimation process is to have each object discussed in a way that everyone on the team understands, thereby avoiding misunderstandings during implementation. Accelerating the process to ultimately document an estimation of the estimation objects in the system can lead to more effort being expended during implementation in order to resolve misunderstandings or even worse. Therefore, it makes sense to invest the time at the beginning of each iteration to conduct the estimation process with the entire team to ensure that everyone understands what will be implemented in the next iteration and thus preemptively resolve misunderstandings. Moreover, such an approach results in estimations not being objectified through multiple perspectives but instead being based on one or two subjective assessments.

The granularity of the estimation objects is not done accurately
The results of the case study indicate that one of the most common challenges practitioners face is the breaking down process of the respective estimation objects. During the initial interviews, only one interviewee mentioned this challenge, but the results of the evaluation show that all respondents either agree or strongly agree with this challenge. If the breakdown of estimation objects is done in an inaccurate granularity, it leads to over/underestimations or, in the worst case, requires the entire estimation object to be moved to the next iteration. Shifting estimation objects necessitates an adjustment of the respective estimation and can lead to further challenges, such as delays or inaccurate tracking of metrics, e.g., team velocity. If an estimation object, such as a work item, is not yet in the processing phase and is moved to the next iteration without being worked on, it does not pose a problem. Therefore, it is worthwhile to invest time in the breakdown process and divide estimation objects into smaller parts that can certainly be completed within one iteration, even considering unforeseen circumstances.

Dependencies between teams/workstreams are difficult to manage and remain one of the biggest challenges
As mentioned in the second chapter, dependencies between teams are one of the major challenges in large-scale agile development. Therefore, it was unsurprising that this issue would also pose a problem in effort estimation within the case organization. Seven partic-

ipants identified dependencies on other teams/workstreams during the initial interviews as a challenge. In the evaluation, 11 out of 12 respondents agreed or strongly agreed with this challenge in the context of effort estimation. We observed that the case organization had made great efforts to mitigate this problem by identifying dependencies during the pre-implementation phase, maintaining a dependency list to document all dependencies, and conducting meetings to address this issue. However, the varying opinions regarding whether these measures successfully mitigate the problem of dependencies suggest that an optimal strategy for managing and resolving this issue has not yet been found.

Unclear and incomplete functional descriptions of requirements lead to significant challenges in the estimation process
Another challenge that complicates accurate effort estimation is the presence of unclear and incomplete functional descriptions of certain requirements. This challenge was mentioned by approximately 50% of the interviewees during the initial interviews, and during the evaluation, everyone agreed or even strongly agreed with this issue. When the description of requirements is incomplete or unclear, teams are unsure about what exactly needs to be implemented. Consequently, performing accurate estimations of these items become challenging. To mitigate the challenge of weak functional descriptions, additional overhead in the form of meetings or workshops is required to discuss the respective requirements, ensuring that the teams understand what needs to be implemented. However, unforeseen and additional overhead introduces further challenges in effort estimation, which can be avoided in advance through precise documentation and description of each requirement.

Pressure and control by the management
One of the principles of agile approaches is the autonomy of teams and trusting them to carry out the work in their way [35]. Especially in large projects involving multiple levels of collaboration, a certain degree of control is required to ensure a frictionless project process and achieve goals such as budget and time. In the evaluation, it was stated that finding the ideal level of control where the teams do not feel controlled and pressured but management can still perform their duties and track the project is a significant challenge. Unfortunately, we observed that teams in the case organization do feel controlled and pressured to meet management expectations. Accordingly, 11 out of 12 respondents agree or strongly agree that pressure and control by the management pose a challenge within the program. It is argued that this pressure particularly affects the accuracy of the estimations. As a result, inaccurate estimations are often made to meet management's expectations rather than reflecting reality.

Support and motivation by Scrum Masters and Agile Coaches is a good proposition to mitigate challenges in effort estimation
A Scrum Master is responsible for supporting the organization and teams in implementing agile activities in an agile project, regardless of size. Through the assistance and motivation provided by the Scrum Masters/Agile Coaches, certain challenges during effort estimation can be mitigated. The evaluation demonstrates that most respondents either agree or strongly

agree with this statement. Overall, eight out of nine supporters strongly agreed that the support could alleviate certain challenges and lead to more accurate estimations. These findings within the case organization further reinforce the recommendation found in the existing literature [52] to utilize Scrum Masters/Agile Coaches to mitigate challenges during effort estimation. Mallidi et al. [52] suggest training programs to familiarize the team with the agile way of working and empower the team in their self-organized nature, to be creative and innovative, and to acknowledge their expertise.

The use of an appropriate tool can mitigate challenges in effort estimation
The selection of an appropriate tool for agile activities having features like Post-its or Scrum Boards to support the estimation process can mitigate challenges. Ten out of 12 respondents agreed or strongly agreed with this mitigation proposition. During the initial interviews, it was also mentioned multiple times that the tool used within the program was not sufficiently supportive of the estimation process. Some interviewees noted that the tool's complexity and inflexibility posed a challenge and that it would be better to separate the budgeting processes from the pure agile activities using different tools for each operation. According to the evaluation results, it was not unanimously observed that the tool posed a challenge. Nevertheless, an appropriate tool that supports the estimation process and other agile activities appears to be relevant and can help mitigate challenges faced by practitioners during effort estimation.

7.2. Limitations

This section discusses the quality of research in the context of this bachelor's thesis and the possible threat to validity. During the data collection and analysis, guidelines from Yin [101] and Runeson and Höst [71] have been applied to enhance research quality. Among others, it is recommended to use multiple sources for data collection, which was realized in this thesis through the use of first-degree data collected during semi-structured interviews and third-degree data provided in the form of presentation slides, project documentation, as well as internal wiki pages [71, 101]. Further, Yin [101], as well as Runeson and Höst [71], recommend maintaining a chain of evidence, which was implemented through coding and analysis of the collected data using guidelines from Saldaña [72] and Miles et al. [53]. To ensure the reliability and authenticity of sources, only electronic sources in the form of internal sources were used, as recommended by Yin [101]. In the following, the threat to validity is discussed according to the classification of Runeson and Höst [71], based on that of Yin [101], classifying the validity in construct validity, internal validity, external validity, as well as reliability.

Construct Validity

The aspect of construct validity reflects the extent to which the research truly represents what the researcher has in mind and what is being investigated according to the research questions.

According to Runeson and Höst [71], construct validity is compromised, for example, when researchers and interviewees do not interpret interview questions in the same way. Data triangulation was employed to minimize the threat to construct validity by using various sources for data collection and ensuring a chain of evidence through coding and analysis of the data based on the guidelines of Saldaña [72] and Miles et al. [53]. Additionally, interview participants were selected to have different roles in examining the research topic from multiple perspectives. Further, if it was noticed during the interviews that interview questions were being interpreted differently than intended, they were addressed and resolved during the interviews improving the construct validity. This active response was unfortunately not possible in the online surveys. Still, misunderstandings or misinterpretations were mitigated by providing a detailed description of the data collection, how the respective challenges and mitigation propositions were identified, and a description of the challenges and mitigation propositions themselves.

Internal Validity

The aspect of internal validity concerns the investigation of causal relationships, i.e., when certain factors influence the investigated factors, there is a risk that the investigated factors may also be affected by other third factors. According to Runeson and Höst [71], validity is at risk, for example, when the researcher is unaware of the extent to which third factors influence the investigated factors. In the context of effort estimation, it is difficult to identify causal relationships as there are many confounding factors [95]. However, possible threats to internal validity were minimized by applying data triangulation, i.e., factors mentioned in related work were considered, and experts involved in effort estimation in the case organization were consulted. In the qualitative part of the thesis, the semi-structured interviews, threats to internal validity were mitigated by involving three researchers in designing the interview questions and questionnaire (investigator triangulation) to reduce investigator bias. In addition, interviewee bias was minimized by interviewing experts with different roles and incorporating data from third-degree sources such as presentation slides and documentation (data triangulation).

External Validity

The aspect of external validity deals with the extent to which the results can be generalized. In this thesis, a case study was applied, and as a result, the findings are strongly bounded to the context of this case study. Additionally, the case study is limited to a project within a company in which all interviewees were involved. During the estimation process, especially at the team level, the teams were able to create their way of estimating independently of other teams. However, they were indirectly influenced by specific guidelines from program leadership, such as the normalization of story points. It should also be noted that the project is a large-scale agile standard software implementation project, not a typical large-scale agile software development project. Nevertheless, we attempt to mitigate the potential threat of external validity through a detailed description and presentation of the research context to

make it easier to compare the current case with similar cases.

Reliability

The aspect of reliability deals with the extent to which data collection and analysis depend on researchers. According to Runeson und Höst [71], validity is at risk if, for example, it is unclear how the data was coded or if the interview questions or questionnaires are vague. Therefore, the study should ideally yield the same results if conducted by another researcher. To mitigate this threat to reliability, the approach peer debriefing was used, i.e., three researchers designed the interview questions based on the guidelines of Runeson and Höst [71] and then discussed with a company representative to minimize reliance on any particular individual. The data analysis was conducted according to the guidelines of Saldaña [72] and Miles et al. [53], which are documented in detail in Chapter 4 and are thus comprehensible and repeatable for any researcher. In addition, the interview questionnaire and the evaluation questionnaire can be found in Appendix B. However, it should be noted that replicating this study could lead to different results, as the case study was only conducted within one program where all levels work together and likely face the same problems. The organizational structure, inter-team coordination, estimation methods, and many other factors can significantly impact the results and may lead to different challenges.

8. Conclusion

This chapter summarizes the answers to the respective research questions and provides an outlook on future research in the context of challenges and mitigation propositions for effort estimation in large-scale agile development.

8.1. Summary

In this section, the three research questions will be answered in summary.

RQ1: How is effort estimation conducted in the case organization?

The collected data from the interviews at the case organization and documents such as presentation slides were evaluated and analyzed, providing the information to answer the first research question. The results showed that within the case organization, effort estimation is carried out in three main events at different levels. At the highest level during the Roadmap Planning event, a selected group of participants estimates the requirements roughly in IT effort points. This group usually consists of the Product Manager, Product Owner, Subject Matter Experts, and Solution Architects of the respective teams. This planning event takes place on an annual basis, so once for each rollout company. At the intermediate level during the Wave Planning event, the Scrum teams break these requirements into smaller units, called work packages, and estimate them more accurately in Story Points. The Product Owner of the team only participates in the estimation in a supporting role. The initial estimates of each requirement serve as the basis for the estimates of the associated work packages, which are adjusted if the initial estimation is too high or too low. Further, the work packages are additionally estimated in value points by the Scrum team to simplify the Backlog prioritization for the Product Owner. This planning event takes place every quarter, so once at the beginning of each Wave. At the lowest level during the Sprint Planning event, the already broken-down work items are estimated in story points by the Scrum team, with the Product Owner again participating in a supporting role. Here, the estimation of the work package serves as the basis for the estimates of the associated work items and is adjusted in case of over- or underestimation. This planning event takes place on a monthly basis, so once for each Sprint. The accuracy of the estimation increases with the available information leading to the estimates at the work item level being the most accurate while those on the requirement level are the roughest or least accurate. In addition to these planning events, there are other events, such as the Wave Planning Presentation to identify dependencies or

the Backlog Refinement, where estimation and prioritization are adjusted. Excel[6] is used to document the Backlog, estimations, and prioritization, especially at the requirement level, as well as a special tool provided by the software company for the purposes of such projects.

RQ2: What are challenges in effort estimation in scaling agile environments?

With the analysis of the collected data from the interviews, 25 challenges were identified and then assigned to the four categories program setting, collaboration, expertise, and uncertainties and information deficit. Eight challenges related to the program setting were identified, with **time restrictions (C2)** being the most frequently mentioned challenge in this category. Further, six challenges were assigned to the category collaboration, with **Considering dependencies to other teams, workstreams, and systems in the estimation (C14)** being the most frequently mentioned challenge. Additionally, six challenges were identified as related to the category expertise, while five were identified as emerging due to uncertainties and information deficits. Overall, the most frequently mentioned challenge by far was **unclear or incomplete specification of the requirements (C23)**, which was mentioned by ten respondents and was assigned to the category uncertainties and information deficit. Furthermore, 13 of these 25 challenges were categorized as typical challenges that arise during effort estimation in large-scale agile development projects due to the agile values and principles, using our expertise and existing literature in this context. The challenges **project setting (C1)** and **monitoring of estimations and actual efforts (C7)** have been classified as typical for standard software projects due to the characteristics of implementation projects. In the results of the evaluation, it was observed that five challenges were almost unanimously agreed upon or strongly agreed upon. These challenges are **project setting (C1)**, **pressure and control by management (C4)**, **considering dependencies to other teams, workstreams, and systems in the estimation (C14)**, **adjustment of effort estimates (C20)**, and **unclear or incomplete specification of the requirements (C23)**. The challenge that performed worst in the evaluation is the challenge **missing knowledge about resources in terms of people involved in the implementation (C24)**. Experts did not exclusively disagree, but in total, five (strongly) disagreed, while none strongly agreed. Based on the feedback and improvement suggestions of the experts collected during the evaluation, adjustments have been made to the descriptions of four challenges. Refining the respective descriptions only included small adjustments to create more clarity which can be found in the final version of the artifact in Appendix D.

RQ3: How can these challenges in effort estimation in scaling agile environments be addressed?

The data used to answer the third research question came from the initial interviews and existing literature in this context. A total of 21 mitigation propositions were identified, each of which was associated with challenges that could be directly mitigated by these. Specifically, only challenges that a particular proposition can mitigate have been considered, not the

[6]https://www.microsoft.com/de-de/microsoft-365/excel

challenges that can be mitigated by mitigating another challenge on which it depends. The assignment of challenges to mitigation propositions was done through direct mention of challenges during the interviews, existing literature, as well as our expertise and knowledge in this field. The majority of all mentioned mitigation propositions during the interviews were also found similarly in the existing literature. The mitigation proposition **use agile metrics and store estimates for improvement and future use (M21)** was the only proposition identified exclusively through the literature, i.e., none of the interviewees mentioned this proposition during the interviews. 12 out of these 21 identified mitigation propositions were classified as typical large-scale agile due to the agile values and principles. The evaluation did not reveal a clear best mitigation proposition, as the ratings were mixed. However, for four out of the 21 mitigation propositions, none of the experts disagreed or strongly disagreed. These propositions were **deep dive sessions and workshops to clarify requirements and needed resources (M4)**, **Guidelines and standard estimates for tasks uniform across all teams (M7)**, **support and motivation by scrum masters and agile coaches (M15)**, and **Tool/features to support the estimation process (M17)**. Proposition **use T-Shirt size as an estimation unit to avoid difficulties to estimate in story points and trust the team more (M20)** performed the worst, with only three experts agreeing and four disagreeing or strongly disagreeing. Two out of the three experts who agreed noted that the mitigation proposition is more suitable for high-level estimates and less effective for estimations at a lower level, as they tend to be less precise. Based on the feedback and improvement suggestions from the experts, adjustments were made to seven mitigation propositions. In one out of the eight cases, adjustments were made to the description and the name. Additionally, further allocations were made because, based on the feedback, it was observed that additional challenges could be mitigated by certain propositions that were not considered before. In total, five challenges were newly assigned to four mitigation propositions. The refinements and adjustments can be looked up in detail in the final version of the artifact in Appendix D.

8.2. Future Work

In this thesis, an artifact consisting of a collection of challenges and mitigation propositions in the context of effort estimation in large-scale agile development was developed by implementing one cycle of the Action Design Research approach. In conjunction with the Reflection and Learning Phase, the improvement suggestions made during the evaluation were incorporated into the artifact (final version in Appendix D). Due to time constraints, it was beyond the scope of this bachelor's thesis to apply the identified propositions in practice to mitigate the emerging challenges. Therefore, further cycles of the Action Design Research approach should be implemented in future work to refine the mitigation propositions. In addition, the application of these mitigation propositions in practice should be investigated, and the results should be confirmed or refuted by practical applications in other case organizations. Through future investigation of the challenges and mitigation propositions in effort estimation in large-scale agile development, it is possible to generalize our results and not limit them to the case study conducted to collect the primary data in this thesis.

A. Appendix A

A.1. Key Values and Principles behind the Agile Manifesto

	Key Values
V1	Individuals and interactions over processes and tools
V2	Working software over comprehensive documentation
V3	Customer collaboration over contract negotiation
V4	Responding to change over following a plan.

Note: The items on the left are valued more than those on the right

Table A.1.: Key Values of the Agile Manifesto [35]

	Principles
P1	Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
P2	Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
P3	Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
P4	Business people and developers must work together daily throughout the project.
P5	Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
P6	The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
P7	Working software is the primary measure of progress.
P8	Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
P9	Continuous attention to technical excellence and good design enhances agility.
P10	Simplicity–the art of maximizing the amount of work not done–is essential.
P11	The best architectures, requirements, and designs emerge from self-organizing teams.
P12	At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Table A.2.: Principles behind the Agile Manifesto [35]

B. Appendix B

B.1. Questionnaire of Case Study Interviews

First Part: General Information

1.1. What is your role in large-scale agile IT projects (stakeholder)?
1.2. How long have you been working in agile IT projects?
1.3. How long have you been working in large-scale agile IT projects?
1.4. How long has your company been working with agile IT projects?
1.5. How long has your company been working with large-scale agile IT projects?
1.6. In which working unit(s) of the program are you involved? (e.g., work-stream)
1.7. Are we allowed to contact you again for further research? (e.g., for questions regarding the given answers)

Second Part: Effort Estimation within the Program
Time and Scope

2.1. When is effort estimation performed?
  a) In which planning events?
  b) How often is effort estimation performed?
  c) For which time scope (sprint, iteration aka. wave)?

Involved Persons

2.2. Which stakeholders are involved in the effort estimation and how?
2.3. What is your role in the effort estimation?

Estimated Objects

2.4. How are the objects (requirements, work packages, work items, or tasks) broken down?
2.5. Which objects (requirements, work packages, work items, or tasks) are estimated?
2.6 Which units are used to estimate which object? And why are different units used? (e.g., Story Points)
2.7. Is the classification of the objects relevant for their estimation? (e.g., Gap, Fit)

Techniques/Methods

2.8. Do you use specific techniques/methods for the effort estimation? (e.g., planning poker)

Changes/Adaptions

2.9. Do you adjust the effort estimation?
  a) Why/ When?
  b) How often?

Actual Effort

2.10. Which role does the actual effort spent play?

Documentation and Monitoring

2.11. How are the effort estimations documented and communicated?
2.12. Are you using any tools for the documentation and monitoring of effort estimations?
2.13. Are these tools helpful in supporting the estimation process?
2.14. Are you using any metrics for monitoring efforts and the accuracy of estimations?

Challenges

2.15. Which challenges do you face when estimating?
2.16. How do you address these challenges?
2.17. Which actions can be (ideally) implemented to avoid these challenges?
2.18. Are there any other challenges you see regarding the overall estimation process of your program?
2.19. How do you think these challenges could be approached?

Third Part: Discussion

3.1. How can research support effort estimation in the industry?
3.2. Do you want to add any further information, comment, or a topic that we missed?

B.2. Questionnaire of Evaluation Survey

First Part: General Information

1.1. What is your role in large-scale agile IT projects (stakeholder)?
1.2. How long have you been working in large-scale agile IT projects?

Second Part: The Challenges

Program Setting

All questions from 2.1 to 2.8 using a Likert scale asking respondents to choose between Strongly Agree, Agree, Neither Agree nor Disagree, Disagree, and Strongly Disagree. Question 2.9 allowed respondents to provide an answer in their own words.

2.1. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

> **C1: Project setting**
> A fixed, large time frame requiring initial estimates and the tight linkage of estimates to the budget limit the flexibility of the estimations and lead to inaccurate initial estimates

2.2. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

> **C2: Time restrictions**
> The process of estimating effort and identifying dependencies is very time-consuming and

anyway, time is limited

2.3. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C3: Unclear responsibilities
Lack of knowledge about the responsibility of a person, team, or workstream for specific requirements

2.4. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C4: Pressure and control by management
Pressure and control by management, lead to inaccurate estimates to meet management's expectations, and the feeling of being pressured to be faster than management's original estimates

2.5. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C5: Lack of measures to improve estimations
Lack of measures to measure planned and actual efforts, resulting in less potential for improvement in future estimates

2.6. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C6: Inappropriate tool support
The complexity and inflexibility of SolMan, as well as the lack of functions that could support the estimation process, make it difficult to perform, plan, and document estimates

2.7. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C7: Monitoring of estimations and actual efforts
Monitoring estimated and actual effort is difficult due to (a) the complexity and scale of the program, (b) the lack of commitment to transparency, (c) the lack of backlog updates, and (d) the incorrect granularity of the estimation objects

2.8. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C8: Difficulty to estimate in story points unit
Even when companies define a standard for story points, the definition changes from project to project, resulting in teams struggling to understand them. In addition, teams in general often find it difficult to estimate effort using relative quantities

2.9. Do you have any improvement suggestions or want to add anything regarding the identified challenges? e.g., if you have not agreed to a challenge, briefly explain why.

Collaboration
All questions from 2.10 to 2.15 using a Likert scale asking respondents to choose between Strongly Agree, Agree, Neither Agree nor Disagree, Disagree, and Strongly Disagree. Question 2.16 allowed respondents to provide an answer in their own words.

2.10. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

> **C9: Lack of team commitment**
> Lack of team commitment in terms of (a) general resistance to the program's way of working, (b) need to convince new members that the program's way of working is a good approach, (c) team members not participating in estimation due to lack of knowledge/experience, and (d) teams viewing the estimation process as overhead.

2.11. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

> **C10: Lack of knowledge about contact partners in the beginning**
> In the beginning, there is a lack of knowledge about the contacts (e.g., experts) needed for effort estimation

2.12. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

> **C11: Efficient communication despite spatial distribution and language barriers**
> Language barriers make it difficult to correctly understand requirements and conduct the estimation process in virtual meetings complicates the estimation process

2.13. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

> **C12: Having a correct and common understanding of requirements needed for estimations**
> The lack of a common understanding of the quality criteria and the scope of the requirements, as well as these themselves e.g., through language barriers between individuals, teams, and working groups complicate the estimation

2.14. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

> **C13: Difficulty to estimate additional overhead such as meetings and explanations**
> Difficulty in estimating the additional effort (e.g., for meetings, explanations) due to the size and complexity of the program

2.15. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

> **C14: Considering dependencies to other teams, workstreams, and systems in the estimation**

Dependencies on other teams, workflows, and systems make estimation difficult, especially when there is a lack of knowledge about the dependencies

2.16. Do you have any improvement suggestions or want to add anything regarding the identified challenges? e.g., if you have not agreed to a challenge, briefly explain why.

Expertise
All questions from 2.17 to 2.22 using a Likert scale asking respondents to choose between Strongly Agree, Agree, Neither Agree nor Disagree, Disagree, and Strongly Disagree. Question 2.23 allowed respondents to provide an answer in their own words.

2.17. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C15: Subjectivity of estimates
The estimates are based on subjective criteria such as the individual knowledge and experience of the estimators

2.18. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C16: Lack of experts involved in estimates
During the estimation process, the participation of experts is required, but the unavailability or absence of experts in the team complicates the estimation process

2.19. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C17: Lack of involvement of experts in top-level estimations and roadmap planning
Lack of involvement of experts with knowledge and experience in estimation at the highest level Level (e.g., roadmap planning) leads to inaccurate initial estimates

2.20. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C18: Lack of knowledge and experience regarding effort estimation
Teams often lack experience and knowledge in effort estimation, making accurate estimation difficult

2.21. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C19: Neglection of relevant factors when estimating
Management neglects relevant factors (e.g., dependencies in initial estimates and non-functional requirements) and budget planning only takes the company code into account, but not the complexity of the requirement and the associated effort

2.22. To what extent do you agree that the challenge below is a challenge in the context of effort

estimation:

C20: Adjustment of effort estimates
Unfinished tasks must be moved as a whole to the next iteration (e.g., Wave, Sprint), i.e., the effort already spent on the task is missing from the backlog of the previous iteration and the estimate in the next iteration must be adjusted accordingly

2.23. Do you have any improvement suggestions or want to add anything regarding the identified challenges? e.g., if you have not agreed to a challenge, briefly explain why.

Uncertainties and Information Deficit
All questions from 2.24 to 2.28 using a Likert scale asking respondents to choose between Strongly Agree, Agree, Neither Agree nor Disagree, Disagree, and Strongly Disagree. Question 2.29 allowed respondents to provide an answer in their own words.

2.24. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C21: Information deficit in the initial estimation of large, complex requirements
Information deficits about the requirement (especially at the beginning) due to size, dependencies, and uncertainties

2.25. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C22: Information deficit regarding new requirements
Information deficit regarding new requirements ("greenfield requirements"), the implementation of which is uncertain and for which no comparable requirements are available

2.26. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C23: Unclear or incomplete specification of the requirements
An unclear specification and a lack of information in the functional description of requirements make it difficult to estimate

2.27. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C24: Missing knowledge about resources in terms of people involved in the implementation
Lack of knowledge about the people working on a requirement, their availability, and their skills in estimating

2.28. To what extent do you agree that the challenge below is a challenge in the context of effort estimation:

C25: Unforeseen changes
Unforeseen changes such as: (a) system or process related issues during implementation, (b) ad

hoc requirements, (c) changes in timelines, or (d) individuals leaving the program

2.29. Do you have any improvement suggestions or want to add anything regarding the identified challenges? e.g., if you have not agreed to a challenge, briefly explain why.

Third Part: The Mitigation Propositions

All questions in Part Three use a Likert scale in which respondents were asked to choose between Strongly Agree, Agree, Neither Agree nor Disagree, Disagree, and Strongly Disagree. In addition to each question, there was an opportunity to add possible improvement suggestions in their own words using a comment box.

3.1. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M1: Adding a buffer to estimations in case of uncertainty
Add a buffer for uncertainty to the estimation

3.2. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M2: Additional phase before the implementation phase to check requirements in detail (feasibility and quality)
Additional pre-implementation design phase to discuss requirements at a detailed level; Literature also recommends a pre-implementation phase with the customer to increase the success of the ERP implementation project [3].

3.3. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M3: Cultivation of a dependency list
Dependencies between teams, work streams, and systems, as well as the understanding of requirements, should be listed in a dependency list.

3.4. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M4: Deep dive sessions and workshops to clarify requirements and needed resources
Additional meetings/workshops/deep-dive sessions to clarify understanding of requirements and discuss resources needed (e.g., people involved in implementation). Literature advises customers to participate in additional sessions to resolve ambiguities [6].

3.5. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M5: Early and continuous communication between all levels: teams, workstreams, and program management
Early communication should take place between all levels to clarify understanding and actively

ask program management for support/experts. This is also recommended by the literature [3, 7].

3.6. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M6: Events to recap on learning and document those
Events (e.g., Sprint Review) should be used to discuss what has been learned and to document findings in order to improve in the next iteration(e.g., Sprint). This is also recommended in the literature in the form of feedback sessions [15].

3.7. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M7: Guidelines and standard estimates for tasks uniform across all teams
Provide guidelines and set a standard for components/tasks that are common to all teams (e.g., documentation, testing). The literature suggests defining standards for e.g., the complexity or scope of a story to improve estimates [30].

3.8. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M8: Consider additional effort regarding organizational and process factors during estimation
Organizational and process factors relevant to implementation should also be considered. Overall, research recommends considering factors that influence the effort estimation process, as well as non-functional requirements or testing effort [6, 30, 36, 52, 68].

3.9. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M9: Include people with experience in estimating effort and reacting to unforeseen in the estimation process
Experienced individuals who know how to estimate, respond to unforeseen, and facilitate the process efficiently should be part of the team and included in high-level estimation (e.g., roadmap planning). Similar advice is recommended in the literature [3, 7, 68, 88, 89, 90].

3.10. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M10: Measures to convince the team that effort estimation is a group/team activity
Measures to convince the team that effort estimation is a group activity, regardless of individual skill levels.

3.11. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M11: Normalization of story points (to person days)
Normalize the Story Points size metric to person days to ensure that everyone has a reference

and understanding of the size metric. Ensure that there is a clear and standardized definition of Story Points within the project [96].

3.12. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M12: Plan requirements for an appropriate time frame to improve the accuracy of estimates
Planning requirements in a reasonable time frame (e.g., not too far in advance). Nevertheless, research recommends performing planning activities, such as effort estimates, with sufficient lead time to identify certain factors, such as dependencies, at an early stage [7, 95].

3.13. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M13: Platforms and meetings to identify and discuss dependencies
Meetings and communication platforms to identify dependencies e.g., cross-team planning workshops or unit-wide retrospectives [7, 95].

3.14. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M14: Reduce time tracking pressure for employees
Scheduling more time for employee networking so they are more receptive to honest time tracking.

3.15. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M15: Support and motivation by scrum masters and agile coaches
Involve Scrum Masters and Agile Coaches to create an understanding of the importance of the estimation process, keep the team motivated, and support the estimation process. Literature recommends training programs for the teams [90].

3.16. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M16: Tool support to automate the estimation process
Use available data and tools to automate the estimation process to reduce time, complexity, and incomparability and provide more transparency. Analogies with similar projects and providing an initial estimate could also improve the process [15, 30].

3.17. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M17: Tool/feature to support the estimation process
Adding features (e.g., post-its, scrum boards) to the tool to support the estimation process. Literature suggests features to visualize e.g., metrics or propose initial estimates [30, 52].

3.18. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M18: Tracking of actual efforts
Tracking actual effort to understand the reasons for moving requirements to the next iteration (e.g., sprint) and improve estimates in the future. Literature recommends using accuracy metrics such as actual effort [30].

3.19. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M19: Use of supporting techniques during the estimation process
Use techniques (e.g., planning poker, spike stories) to improve estimation accuracy. Literature recommends similar, as well as "silent grouping" [89] and emphasizes that techniques should be selected based on project size and team experience [96].

3.20. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M20: Use T-Shirt size as an estimation unit to avoid difficulties to estimate in story points and trust the team more
Use T-shirt size as the unit of measurement to avoid difficulty in estimation and to ensure the team understands the unit of measurement. Research [96] suggests using a clearly defined, objective size measurement to avoid story point misunderstandings and ensure everyone understands the unit of measurement.

3.21. To what extent do you agree that the mitigation proposition below is a proposition to mitigate challenges in the context of effort estimation:

M21: Use agile metrics and store estimates for improvement and future use
Several researchers [30, 52, 88, 89] recommend the use of agile metrics, e.g., velocity, to better predict uncertain estimates, and to store estimates to track and reflect estimation performance.

C. Appendix C

C.1. Detailed Results of the Case Study

Alias	Company	Role	Experience in agile software dev. (years)		Experience in large-scale agile software dev. (years)		Duration (h:m)
			Expert	Company	Expert	Company	
SM1	SoftwareCo	Agile Coach and Scrum Master	6 - 10	6 - 10	6 - 10	6 - 10	0:39
PJM1	SoftwareCo	Project Manager	6 - 10	16 - 20	6 - 10	16 - 20	0:50
SolAr1	SoftwareCo	Solution Architect	3 - 5	16 - 20	3 - 5	16 - 20	0:45
Dev1	NetInfrCo	Developer	1 - 2	6 - 10	1 - 2	3 - 5	0:45
PO1	NetInfrCo	Product Owner	1 - 2	3 - 5	1 - 2	3 - 5	0:44
PO2	NetInfrCo	Product Owner	3 - 5	6 - 10	3 - 5	6- 10	0:31
PM1	SoftwareCo	Product Manager	6 - 10	6 - 10	3 - 5	6 - 10	0:44
PMO1	ConsultCo	PMO	11 - 15	11 - 15	6 - 10	11 - 15	0:40
PO3	NetInfrCo	Product Owner	6 - 10	6 - 10	6 - 10	6 - 10	0:38
Dev2	NetInfrCo	Developer	3 - 5	3 - 5	3 - 5	3 - 5	0:27
SM2	NetInfrCo	Scrum Master	6 - 10	6 - 10	3 - 5	6 - 10	0:29
BC1	NetInfrSubCo	Business Contact	1 - 2	> 20	1 - 2	> 20	0:48
SolAr2	SoftwareCo	Solution Architect	3 - 5	> 20	3 - 5	> 20	0:43
PMO2	ConsultCo	PMO	6 - 10	> 20	6 - 10	11 - 15	0:38
SolAr3	SoftwareCo	Solution Architect	11 - 15	11 - 15	6 - 10	6 - 10	0:47
PM2	NetInfrCo	Product Manager and Product Owner	6 - 10	6 - 10	6 - 10	6 - 10	0:47
BPE1	NetInfrCo	Business Process Expert, Test Coordinator, and Training Coordinator	3 - 5	6 - 10	3 - 5	6 - 10	0:47
SolAr4	NetInfrCo	Solution Architect	6 - 10	3 - 5	6 - 10	3 - 5	0:45
PM3	NetInfrCo	Product Manager	3 - 5	6 - 10	3 - 5	6 - 10	0:45
SM3	NetInfrCo	Scrum Master	6 - 10	6 - 10	6 - 10	6 - 10	0:45

Note: The interview participants SolAr3, PM2, and BPE1 took part in a group interview and the interview participants SolAr4, PM3, and SM3 took part in a group interview

Table C.1.: Overview of the interview participants and the companies

C.2. Detailed Results of the Evaluation

The Challenges

Challenge	Strongly Agree (1)	Agree (2)	Neither Agree nor Diagree (3)	Diagree	Strongly Disagree (3)
C1	4	7	1	0	0
C2	2	5	1	4	0
C3	0	8	3	1	0
C4	6	5	1	0	0
C5	1	4	2	4	1
C6	3	1	3	3	2
C7	4	3	3	2	0
C8	1	3	3	4	1
C9	4	4	1	1	2
C10	0	5	5	1	1
C11	1	7	1	3	0
C12	4	7	0	1	0
C13	3	5	2	2	0
C14	5	6	1	0	0
C15	7	2	0	3	0
C16	1	6	2	2	1
C17	1	6	1	3	1
C18	1	5	1	5	0
C19	3	6	0	3	0
C20	5	7	0	0	0
C21	6	5	1	0	0
C22	4	6	1	1	0
C23	6	6	0	0	0
C24	0	5	2	4	1
C25	2	7	3	0	0

Table C.2.: Overview of the quantitative evaluation results (per answer option for each challenge)

The Mitigation Propositions

Mitigation Proposition	Strongly Agree (1)	Agree (2)	Neither Agree nor Diagree (3)	Diagree	Strongly Disagree (3)
M1	3	5	1	1	2
M2	2	4	2	1	3
M3	2	8	1	1	0
M4	4	5	3	0	0
M5	2	7	1	2	0
M6	3	7	0	1	1
M7	4	5	3	0	0
M8	2	6	4	0	0
M9	4	4	2	1	1
M10	1	5	4	1	1
M11	4	5	1	1	1
M12	2	7	2	1	0
M13	3	7	0	2	0
M14	4	3	2	3	0
M15	8	1	3	0	0
M16	3	3	4	1	1
M17	4	6	2	0	0
M18	0	6	2	3	1
M19	0	5	4	2	1
M20	0	3	5	2	2
M21	3	6	2	0	1

Table C.3.: Overview of the quantitative evaluation results (per answer option for each mitigation proposition)

D. Appendix D

D.1. Adjustments to the Challenges

ID	Name	Description	CL	Interviewees
		Program Setting		
C1	Project setting	A fixed, large time frame requiring initial estimates and the tight tying of estimates to the budget limit the flexibility of estimations and lead to inaccurate initial estimates	SS	PJM1, Dev1, Dev2, SM2, SolAr2, SM3
C2	Time restrictions	There is a lack of time for estimating requirements, but the process of estimating effort and identifying dependencies is very time-consuming	LSAD	PJM1, PM1, PO3, SM2, BC1, SolAr2, PM2
C3	Unclear responsibilities	Lack of knowledge about the responsibility in terms of person, team, or workstream for certain requirements makes effort estimation difficult		PO2
C4	Pressure and control by management	Pressure and control from management, resulting in inaccurate estimates to meet management expectations and feeling pressured to be faster than management's original estimates		SM1, SolAr1
C5	Lack of measures to improve estimations	Lack of measures/metrics that measure planned and actual efforts resulting in less potential for improvement in future estimates		Dev2
C6	Inappropriate tool support	The complexity and inflexibility of the tool, as well as the lack of features that could support the estimation process, makes the requirement breakdown process, planning, and documenting of estimates difficult	LSAD	PM1, PO3, SolAr2

C7	Monitoring of estimations and actual efforts	The monitoring of estimates and actual effort is difficult due to the complexity and size of the program, the lack of commitment to transparency, the lack of keeping the backlog updated, and the little time invested in the breakdown of the objects leading to wrong granularity	LSAD, SS	SM1, PMO1, PMO2
C8	Difficulty to estimate in story points unit	Even if companies define a standard for story points, the definition can change from project to project leading to teams having difficulties understanding them. Further, teams often find it difficult to use relative size metrics to estimate effort	LSAD	PJM1, SolAr1
		Collaboration		
C9	Lack of team commitment	Lack of team commitment in terms of the general resistance to the program's way of working and the necessity to convince new members that the program's way of working is a good approach. Further, team members not participating in the estimation due to the lack of knowledge and experience, as well as teams seeing the estimation process as overhead. Also, team members with experience mostly have different understandings of what agile is, often not matching the specific program's way of working	LSAD	SM1, PJM1, PO2, PO3, SM2
C10	Lack of knowledge about contact partners in the beginning	At the beginning, there was a lack of knowledge about contacts (e.g., experts) required for the effort estimation process		PM1
C11	Efficient communication despite spatial distribution and language barriers	Language barriers hindering the correct understanding of requirements and performing the estimation process in virtual meetings makes estimating difficult	LSAD	SM1, SolAr1, PO1

C12	Having a correct and common understanding of requirements needed for estimations	A correct and common understanding of the quality criteria and scope of the requirements, as well as the requirements themselves, is necessary for estimating the requirements. The lack of this understanding or language barriers between individuals, teams, and workstreams complicating the understanding makes estimating difficult		PO2, PM1, PO3, Dev2, PMO2
C13	Difficulty to estimate additional overhead such as meetings and explanations	Difficulty to estimate additional overhead (e.g., meetings, explanations) due to the size and complexity of the program	LSAD	Dev1
C14	Considering dependencies to other teams, workstreams, and systems in the estimation	Dependencies to other teams, workstreams, and systems make it difficult to estimate, especially when there is a lack of knowledge regarding the dependencies	LSAD	PJM1, SolAr1, Dev1, PO1, PO3, SolAr2, PM3
		Expertise		
C15	Subjectivity of estimates	Estimates are based on subjective criteria such as the individual knowledge of the estimators		SM1, PM1
C16	Lack of experts involved in estimates	There is a need for expert involvement during the estimation process, but the unavailability or absence of experts on the team makes estimation difficult		SolAr1, PO1, PM1, SolAr2
C17	Lack of involvement of experts in top-level estimations	Lack of involvement of experts with the knowledge and experience in top-level estimation (e.g., roadmap planning) increases the risk of inaccurate initial estimates		BC1
C18	Lack of knowledge and experience regarding effort estimation	Teams often lack experience and knowledge regarding effort estimation, making accurate estimation difficult		PJM1, PO1, PM1, BC1, SolAr2
C19	Neglection of relevant factors when estimating	Management neglects relevant factors such as dependencies in initial estimates or non-functional requirements. In addition, during budget planning, only the factor of the size of the organization's structure is considered, but not the complexity of the requirement and the associated effort	LSAD	SM1, Dev2, SolAr2

C20	Adjustment of effort estimates	Unfinished items have to be moved as a whole to the next iteration (e.g., Wave, Sprint), leading to the challenge that the backlog of the previous iteration is missing the effort already spent on the item and the estimate has to be adjusted in the next iteration	LSAD	SolAr2
Uncertainties and Information Deficit				
C21	Information deficit in the initial estimation of large, complex requirements	Information deficit regarding requirements (especially in the beginning) due to size, dependencies, and uncertainty make estimating difficult	LSAD, SS	Dev1, PO2, PMO2
C22	Information deficit regarding new requirements	Information deficit regarding new requirements ("Greenfield-Requirements") for which the implementation is uncertain and no comparable requirements are available		Dev1
C23	Unclear or incomplete specification of the requirements	Unclear specification and an information deficit in the functional description of a requirement makes estimating difficult		SM1, SolAr1, PO1, PMO1, SM2, SolAr2, PMO2, SolAr3, SolAr4, PM3
C24	Missing knowledge about resources in terms of people involved in the implementation	As long as the estimation unit is based on person-days, a lack of knowledge about the person(s) working on a requirement, their availability, as well as their skills in estimation makes estimating difficult	LSAD	PO3, SolAr4
C25	Unforeseen changes	Unforeseen changes in terms of system or process-related problems during implementation, ad-hoc requirements, changes in the timeline, or people leaving the program make estimating difficult	LSAD	PO1, PO3, SM2, BC1, SolAr3, BPE1

Note: Challenges that are typical of agile or large-scale agile due to agile values and principles are abbreviated with LSAD. Challenges that are typical of standard software implementation are abbreviated with SS. The column in which the classification, whether typical LSAD or SS, is abbreviated with CL

Table D.1.: Adjusted Collection of Challenges in Effort Estimation

D.2. Adjustments to the Mitigation Proposition

ID	Name	Related Challenge	Description
M1	Adding a buffer to estimations in case of uncertainty	C13, C21, C22, C23, C25	Add buffer for uncertainty to the estimation (*PMO2, PM2*). Additionally, it would be helpful to provide a standard or guideline that includes explanations of specific uncertainties, as well as their magnitudes to be considered in estimation, to avoid inconsistencies between different teams ($PJM1^{*}$).
M2	Additional phase before the implementation phase to check requirements in detail (feasibility and quality)	C1, C12, C14, C21, C22, C23	Add an additional design phase before implementation to check requirements on a detailed level and the feasibility (*SolAr1*); Similar to the propositions of adding a design phase, literature recommends conducting a pre-implementation phase with the customer to increase the success of the ERP implementation project [90]
M3	Cultivation of a dependency list	C12, C14	List all dependencies between teams, workstreams, and systems and understandings of the requirements in a dependency list (*Dev1, Dev2, PM3*);
M4	**Deep dive sessions and workshops to clarify requirements and needed resources**	C2, C3, C12, C14, C21, C22, C23, C24	Add additional meetings, workshops, or deep dive sessions to clarify the understanding of requirements and discuss resources needed in terms of people involved for implementation (*PO1, PO2, PO3, Dev2, PM2*); It would be advisable to have the customer present in such additional meetings to resolve ambiguities [96]. To reduce the number of additional meetings, research [52, 90] mentions that daily stand-up meetings could be sufficient to clarify an understanding of the requirements. Further, Tanveer et al. [88] find that developing a common understanding, for which such meetings can be constructive, is often more important than making estimates.
M5	**Early and continuous communication between all levels: teams, workstreams, and program management**	C1, C3, C10, C11, C12, C14, C16, C18, C21, C22, C23, C24	Foster communication in an early stage between all levels to clarify understandings and actively ask the program management for support and experts (*Dev1, PO1, Dev2, PM3*); Literature [6] mentions that communication and collaboration at all levels can help to address planning time frames across all hierarchical levels. To address the challenge of communication issues and misunderstanding, a skilled and qualified project manager could be helpful [90].

M6	**Events to recap on learning and document those**	C5, C8, C9, C18, C20	Use events (e.g., Sprint Review) to discuss lessons learned and document findings to improve in the next iteration (e.g., Sprint) (*BC1*). However, make sure that the number of meetings remains reasonable so that the teams have enough time to implement their work (*AC*2*); This is also recommended in the literature in terms of feedback sessions [3]
M7	Guidelines and standard estimates for tasks uniform across all teams	C2, C8, C12, C15, C18	Provide more guidance and establish a standard for estimation components that are uniform across all teams (e.g., documentation, testing) (*PM1, Dev2*); Tanveer et al. [88] propose to define standards, e.g., for the complexity or story size to improve estimates
M8	Consider additional effort regarding organizational and process factors during estimation	C12, C19	Consider not only effort directly associated with the implementation but also from an organizational and process perspective such as communication costs or resources needed for the implementation during the estimation (*PMO2*); In addition, the research [3, 88, 89, 96, 95] recommends considering factors that influence the effort estimation process, such as the implementation experience of the developer, candidates with a high degree of optimism, or the estimation knowledge of team members. Further, all development activities, including, e.g., non-functional requirements that require significant effort or testing effort, should be considered in the estimates [96]

M9	Include people with experience in estimating effort and reacting to unforeseen in the estimation process	C2, C8, C9, C11, C12, C13, C16, C17, C18, C22, C25	Include experienced people in the team who know how to estimate, react to unforeseen, and moderate the estimation efficiently and well and involve experts in top-level estimations (e.g., roadmap planning) (*Dev1, PM1, PO3, BC1, SolAr2*). Further, it is beneficial to involve experts not involved in the program to challenge the estimates and gain further insight from them ($PMO1^*$). In case of language barriers, make sure to include people with business/technological logic and language skills to avoid misunderstandings and difficulties during estimation ($PMO1^*$); literature suggests that all concerned stakeholders should be involved in the effort estimation process [15, 90, 95] and all planning activities should gather and incorporate feedback from experts [6]. Especially the Scrum Master should be involved in the effort estimation process moderating the team and providing the relevant information on new requirements [52]. In case experts are not available, then Hill [36] proposes to compare pending requirements with a collection of already estimated requirements
M10	Measures to convince the team that effort estimation is a group/team activity	C9, C18	Introduce measures that convince the team that effort estimation is a group activity regardless the individual knowledge (*PJM1*);
M11	Normalization of story points (to person days)	C8, C9	Normalize the size metric story points to person days to assure that everyone has a reference and understanding to the size metric (*PJM1*); Normalizing story points in person days is one option to overcome difficulties when estimating in story points, but generally it should be ensured that there is a clear and standardized definition of story points within the project [30]
M12	**Plan requirements for an appropriate time frame to improve the accuracy of estimates**	C1, C20, C24	Plan requirements for an appropriate time frame (e.g., not too far in advance) to improve the accuracy of estimates (*PO3, PM3*); However, research recommends to conduct planning activities, such as effort estimation, with sufficient lead time to identify certain factors, such as dependencies, at an early stage [6, 7].
M13	**Platforms and meetings to identify and discuss dependencies**	C14, C19	Include meetings and communication platforms to discuss and identify dependencies (*PJM1, SolAr1, Dev1, SolAr2*), e.g., cross-team planning workshops or unit-wide retrospectives [7, 6]

M14	**Reduce pressure for employees**	C4	Reduce pressure for employees, e.g., by allocating more time for networking so that they are more open to honest time recording (*Dev1*) or not setting management expectations too high for teams to implement a certain amount of requirements per iteration (e.g., Sprint) ($AC1^*$)
M15	**Support and motivation by scrum masters and agile coaches**	C3, C8, C9, C11, C13, C18, C20, C22, C23, C25	Scrum Masters and Agile Coaches to underline the importance of the estimation process, keep the team motivated and support the estimation process (*PJM1, PMO1*). Further, provide a description of each role so that it is clear who is responsible for what ($PJM1^*$); Mallidi et al. [52] recommend a training program for the team to familiarize the team with the agile way of working and the use of relative estimation metrics, e.g., story points. In addition, it could be helpful to empower the team in their self-organized nature, to be creative and innovative, and to acknowledge their expertise [52].
M16	Tool support to automate the estimation process	C2, C6, C7, C9, C13, C14, C15, C18, C23	Use available data and tool support to automate the estimation process to reduce time, complexity, and incomparability and provide more transparency (*SM1, SolAr1, PM1, PMO1*); In this case, it is essential that the tool is easy to use and data is integrated in a proper way [88]. Research [3, 88] mentions that analogies with similar projects and providing initial estimates can also improve the process and estimates. In the case of providing initial estimates, try to avoid that teams adopt the initial estimates without discussing the requirements ($PMO1^*$)
M17	**Tool/feature to support the estimation process**	C2, C6, C7, C8, C9, C11, C12, C14, C15, C20	Add features (e.g., Post-its, Scrum Boards) to the tool to support the estimation process (*PM1*). Further, make sure to use appropriate tools for agile activities ($SolAr2^*$, $AC2^*$); By adding features that visualize aspects from, e.g., impact analysis or proposed estimates, the challenge of subjectivity can be mitigated and the estimation process, as well as risk management, can be supported [88, 89]

M18	**Tracking of actual efforts**	C5, C7, C15	Track the actual effort to understand reasons for shifting requirements to the next iteration (e.g., Sprint) and improve the estimation performance in future estimates (PM1, PMO1); In addition, using accuracy metrics, such as actual effort, can improve the overall planning process, as well as the development of common understanding [88]. Tracking actuals on a higher level, e.g., management level, may increase the risk of teams feeling controlled or pressured and thus reject the tracking of actuals ($SolAr2^*$, $AC1^*$). Therefore consider tracking the actuals on a team level
M19	**Use of supporting techniques during the estimation process**	C2, C8, C9, C11, C12, C15, C18, C21, C22	Using techniques (e.g., Planning Poker, Spike Stories) to improve estimate accuracy and question their accuracy based on uncertainty and understanding (*SM1, Dev1, PMO2*); In this case, the techniques should be selected based on the project size and experience of the team [52]. Techniques like silent grouping will minimize lengthy discussions, which in turn reduces the time spent on estimating [68] or applying impact analysis helping to identify the impact of implementing a requirement on the existing system can lead to an improvement of the estimation process [88, 89]
M20	**Use T-Shirt size as an estimation unit to avoid difficulties to estimate in story points and trust the team more**	C4, C8, C9	Use the estimation unit T-Shirt size to prevent difficulties in estimation, to assure that everyone has a reference and understanding to the size metric, and to give trust to the team (*SolAr1*). This is most useful for high-level planning estimates ($PJM1^*$, $PMO1^*$); Research [30] proposes to use well defined, objective size measurement to avoid misunderstanding of story points and assure that everyone understanding to the size metric [30]
M21	**Use agile metrics and store estimates for improvement and future use**	C5, C7, C8, C9, C18, C20, C21, C22, C25	Several researchers [15, 52, 88, 89] recommend to use agile metrics, e.g., velocity, to record activities for further use and predict uncertain estimates better, as well as store estimates to track, learn, and reflect the estimation performance

Note: Mitigation propositions that are typical of agile or large-scale agile due to agile values and principles are highlighted in bold; ***The experts marked with * are the experts from the evaluation***

Table D.2.: Adjusted Collection of Mitigation Propositions to address the identified Challenges in Effort Estimation

Bibliography

[1] Abdullah Altaleb and Andrew Gravell. "Effort estimation across Mobile app platforms using agile processes: a systematic literature review". In: *Journal of Software* 13.4 (2018), p. 242.

[2] S Ambler and M Lines. "Scaling agile software development: Disciplined agility at scale". In: *Disciplined Agile Consortium White Paper Series* (2014).

[3] Dirk Basten and Ali Sunyaev. "Guidelines for software development effort estimation". In: *Computer* 44.10 (2011), pp. 88–90.

[4] Kent Beck. *Extreme programming explained: embrace change*. Addison-wesley professional, 2000.

[5] Izak Benbasat, David K Goldstein, and Melissa Mead. "The case research strategy in studies of information systems". In: *MIS quarterly* (1987), pp. 369–386.

[6] 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*. 2016, pp. 1–5.

[7] Saskia Bick, Kai Spohrer, Rashina Hoda, Alexander Scheerer, and Armin Heinzl. "Coordination challenges in large-scale software development: a case study of planning misalignment in hybrid settings". In: *IEEE Transactions on Software Engineering* 44.10 (2017), pp. 932–950.

[8] Michael Bloch, Sven Blumberg, and Jürgen Laartz. "Delivering large-scale IT projects on time, on budget, and on value". In: *Harvard Business Review* 5.1 (2012), pp. 2–7.

[9] Ricardo Britto, Emilia Mendes, and Jürgen Börstler. "An empirical investigation on effort estimation in agile global software development". In: *2015 IEEE 10th international conference on global software engineering*. IEEE. 2015, pp. 38–45.

[10] Ricardo Britto, Muhammad Usman, and Emilia Mendes. "Effort estimation in agile global software development context". In: *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation: XP 2014 International Workshops, Rome, Italy, May 26-30, 2014, Revised Selected Papers 15*. Springer. 2014, pp. 182–192.

[11] Alan W Brown, Scott Ambler, and Walker Royce. "Agility at scale: economic governance, measured improvement, and disciplined delivery". In: *2013 35th International Conference on Software Engineering (ICSE)*. IEEE. 2013, pp. 873–881.

[12] Edna Dias Canedo, Dandara Pereira Aranha, M De Oliveira Cardoso, Ruyther Parente Da Costa, and Leticia Lopes Leite. "Methods for estimating agile software projects: Systematic literature review". In: *Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE*. 2018, pp. 34–39.

[13] Subhas Chandra Misra, Vinod Kumar, and Uma Kumar. "Identifying some critical changes required in adopting agile practices in traditional software development projects". In: *International Journal of Quality & Reliability Management* 27.4 (2010), pp. 451–474.

[14] Tsun Chow and Dac-Buu Cao. "A survey study of critical success factors in agile software projects". In: *Journal of systems and software* 81.6 (2008), pp. 961–971.

[15] Evita Coelho and Anirban Basu. "Effort estimation in agile software development using story points". In: *International Journal of Applied Information Systems (IJAIS)* 3.7 (2012).

[16] Mike Cohn. *Agile estimating and planning*. Pearson Education, 2006.

[17] Samuel Daniel Conte, Hubert E Dunsmore, and YE Shen. *Software engineering metrics and models*. Benjamin-Cummings Publishing Co., Inc., 1986.

[18] Emanuel Dantas, Mirko Perkusich, Ednaldo Dilorenzo, Danilo FS Santos, Hyggo Almeida, and Angelo Perkusich. "Effort estimation in agile software development: an updated review". In: *International Journal of Software Engineering and Knowledge Engineering* 28.11n12 (2018), pp. 1811–1831.

[19] Digital.ai. *16th State of Agile Report*. 2022. URL: https://info.digital.ai/rs/981-LQX-968/images/AR-SA-2022-16th-Annual-State-Of-Agile-Report.pdf (visited on 01/20/2023).

[20] Kim Dikert, Maria Paasivaara, and Casper Lassenius. "Challenges and success factors for large-scale agile transformations: A systematic literature review". In: *Journal of Systems and Software* 119 (2016), pp. 87–108.

[21] Torgeir Dingsøyr, Tor Erlend Fægri, and Juha Itkonen. "What is large in large-scale? A taxonomy of scale for agile software development". In: *Product-Focused Software Process Improvement: 15th International Conference, PROFES 2014, Helsinki, Finland, December 10-12, 2014. Proceedings 15*. Springer. 2014, pp. 273–276.

[22] Torgeir Dingsøyr and Nils Brede Moe. "Research challenges in large-scale agile software development". In: *ACM SIGSOFT Software Engineering Notes* 38.5 (2013), pp. 38–39.

[23] Torgeir Dingsøyr, Nils Brede Moe, Tor Erlend Fægri, and Eva Amdahl Seim. "Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation". In: *Empirical Software Engineering* 23 (2018), pp. 490–520.

[24] Torgeir Dingsøyr, Sridhar Nerur, VenuGopal Balijepally, and Nils Brede Moe. *A decade of agile methodologies: Towards explaining agile software development*. 2012.

[25] Tore Dyba and Torgeir Dingsoyr. "What do we know about agile software development?" In: *IEEE software* 26.5 (2009), pp. 6–9.

[26] Tore Dybå, Dag IK Sjøberg, and Daniela S Cruzes. "What works for whom, where, when, and why? On the role of context in empirical software engineering". In: *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement*. 2012, pp. 19–28.

[27] Felix Evbota, Eric Knauss, and Anna Sandberg. "Scaling up the planning game: Collaboration challenges in large-scale agile product development". In: *Agile Processes, in Software Engineering, and Extreme Programming: 17th International Conference, XP 2016, Edinburgh, UK, May 24-27, 2016, Proceedings 17*. Springer International Publishing. 2016, pp. 28–38.

[28] Marta Fernández-Diego, Erwin R Méndez, Fernando González-Ladrón-De-Guevara, Silvia Abrahão, and Emilio Insfran. "An update on effort estimation in agile software development: A systematic literature review". In: *IEEE Access* 8 (2020), pp. 166768–166800.

[29] Lucas Gren, Alexander Wong, and Erik Kristoffersson. "Choosing agile or plan-driven enterprise resource planning (ERP) implementations–A study on 21 implementations from 20 companies". In: *arXiv preprint arXiv:1906.05220* (2019).

[30] Tuna Hacaloğlu and Onur Demirörs. "Challenges of using software size in agile software development: A systematic literature review". In: *Academic Papers at IWSM Mensura 2018* (2018).

[31] Amani Mahdi Mohammed Hamed and Hisham Abushama. "Popular agile approaches in software development: Review and analysis". In: *2013 International Conference on Computing, Electrical and Electronic Engineering (ICCEEE)*. IEEE. 2013, pp. 160–166.

[32] Fred J Heemstra. "Software cost estimation". In: *Information and software technology* 34.10 (1992), pp. 627–639.

[33] Alan R Hevner, Salvatore T March, Jinsoo Park, and Sudha Ram. "Design science in information systems research". In: *Management Information Systems Quarterly* 28.1 (2008), p. 6.

[34] James A Highsmith and Jim Highsmith. *Agile software development ecosystems*. Addison-Wesley Professional, 2002.

[35] Jim Highsmith. *History: The Agile Manifesto*. 2001. URL: `https://agilemanifesto.org/history.html` (visited on 01/19/2023).

[36] Peter R. Hill. *Practical Software Project Estimation: A Toolkit for Estimating Software Development Effort Duration*. International Software Benchmarking Standards Group (ISBSG), 2011.

[37] Magne Jorgensen. "Experience with the accuracy of software maintenance task effort prediction models". In: *IEEE Transactions on software engineering* 21.8 (1995), pp. 674–681.

[38] Magne Jørgensen. "A review of studies on expert estimation of software development effort". In: *Journal of Systems and Software* 70.1-2 (2004), pp. 37–60.

[39] Magne Jørgensen. "What we do and don't know about software development effort estimation". In: *IEEE software* 31.2 (2014), pp. 37–40.

[40] Thanh Tung Khuat and My Hanh Le. "An effort estimation approach for agile software development using fireworks algorithm optimized neural network". In: *Int J Comput Sci Inf Secur (IJCSIS)* 14.7 (2016), pp. 122–130.

[41] Barbara A Kitchenham, Lesley M Pickard, Stephen G. MacDonell, and Martin J. Shepperd. "What accuracy statistics really measure". In: *IEE Proceedings-Software* 148.3 (2001), pp. 81–85.

[42] Dina Koutsikouri, Sabine Madsen, and Nataliya Berbyuk Lindström. "Agile transformation: How employees experience and cope with transformative change". In: *Agile Processes in Software Engineering and Extreme Programming–Workshops: XP 2020 Workshops, Copenhagen, Denmark, June 8–12, 2020, Revised Selected Papers 21*. Springer International Publishing. 2020, pp. 155–163.

[43] Philippe Kruchten. "Contextualizing agile software development". In: *Journal of software: Evolution and Process* 25.4 (2013), pp. 351–361.

[44] Philippe Kruchten. "Scaling down large projects to meet the agile sweet spot". In: *IBM developerWorks* 13 (2004).

[45] Elvan Kula, Eric Greuter, Arie Van Deursen, and Georgios Gousios. "Factors affecting on-time delivery in large-scale agile software development". In: *IEEE Transactions on Software Engineering* 48.9 (2021), pp. 3573–3592.

[46] Lina Lagerberg, Tor Skude, Pär Emanuelsson, Kristian Sandahl, and Daniel Ståhl. "The impact of agile principles and practices on large-scale software development projects: A multiple-case study of two projects at ericsson". In: *2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*. IEEE. 2013, pp. 348–356.

[47] Valentina Lenarduzzi. "Could social factors influence the effort software estimation?" In: *Proceedings of the 7th international workshop on social software engineering*. 2015, pp. 21–24.

[48] Timothy C Lethbridge, Susan Elliott Sim, and Janice Singer. "Studying software engineers: Data collection techniques for software field studies". In: *Empirical software engineering* 10 (2005), pp. 311–341.

[49] Rensis Likert. "A technique for the measurement of attitudes." In: *Archives of psychology* (1932).

[50] Samaneh Madanian, Maduka Subasinghage, and Shingai Conrad Tachiona. "Critical Success Factors of Agile ERP Development and Implementation Projects: A Systematic Literature Review". In: (2021).

[51] Markus Mäkinen. "Enhancing the success of agile enterprise resource planning system implementation project". In: (2022).

[52] Ravi Kiran Mallidi and Manmohan Sharma. "Study on agile story point estimation techniques and challenges". In: *Int. J. Comput. Appl* 174.13 (2021), pp. 9–14.

[53] Matthew B. Miles, A. Michael Huberman, and Johnny Saldaña. "Qualitative data analysis: A methods sourcebook". In: *The coding manual for qualitative researchers* (2018).

[54] Subhas Chandra Misra, Vinod Kumar, and Uma Kumar. "Identifying some important success factors in adopting agile software development practices". In: *Journal of systems and software* 82.11 (2009), pp. 1869–1890.

[55] Kjetil Molokken and Magne Jorgensen. "A review of software surveys on software effort estimation". In: *2003 International Symposium on Empirical Software Engineering, 2003. ISESE 2003. Proceedings.* IEEE. 2003, pp. 223–230.

[56] Samson Wanjala Munialo and Geoffrey Muchiri Muketha. "A review ofagile software effort estimation methods". In: (2016).

[57] Sridhar Nerur, RadhaKanta Mahapatra, and George Mangalaraj. "Challenges of migrating to agile methodologies". In: *Communications of the ACM* 48.5 (2005), pp. 72–78.

[58] Marta Olszewska, Jeanette Heidenberg, Max Weijola, Kirsi Mikkonen, and Ivan Porres. "Quantitatively measuring a large-scale agile transformation". In: *Journal of Systems and Software* 117 (2016), pp. 258–273.

[59] Neslihan Küçükateş Ömüral and Onur Demirörs. "Effort Estimation for ERP Projects—A Systematic Review". In: *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE. 2017, pp. 96–103.

[60] Wanda J Orlikowski. "Improvising organizational transformation over time: A situated change perspective". In: *Information systems research* 7.1 (1996), pp. 63–92.

[61] Mohd Owais and R Ramakishore. "Effort, duration and cost estimation in agile software development". In: *2016 Ninth International Conference on Contemporary Computing (IC3)*. IEEE. 2016, pp. 1–5.

[62] Maria Paasivaara, Benjamin Behm, Casper Lassenius, and Minna Hallikainen. "Large-scale agile transformation at Ericsson: a case study". In: *Empirical Software Engineering* 23 (2018), pp. 2550–2596.

[63] Srilata Patnaik and Satyendra C Pandey. "Case study research". In: *Methodological issues in management research: Advances, challenges, and the way ahead*. Emerald Publishing Limited, 2019.

[64] Kai Petersen and Claes Wohlin. "A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case". In: *Journal of systems and software* 82.9 (2009), pp. 1479–1490.

[65] Kai Petersen and Claes Wohlin. "The effect of moving from a plan-driven to an incremental software development approach with agile practices: An industrial case study". In: *Empirical Software Engineering* 15 (2010), pp. 654–693.

[66] Rashmi Popli and Naresh Chauhan. "Agile estimation using people and project related factors". In: *2014 International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE. 2014, pp. 564–569.

[67] Dan Port and Marcel Korte. "Comparative studies of the model evaluation criterions MMRE and PRED in software cost estimation research". In: *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*. 2008, pp. 51–60.

[68] Ken Power. "Using silent grouping to size user stories". In: *Agile Processes in Software Engineering and Extreme Programming: 12th International Conference, XP 2011, Madrid, Spain, May 10-13, 2011. Proceedings 12*. Springer. 2011, pp. 60–72.

[69] Kenneth Preiss, Steven L Goldman, and Roger N Nagel. *Cooperate to compete: Building agile business relationships*. Van Nostrand Reinhold, 1996.

[70] Saja Al Qurashi and M. Rizwan Jameel Qureshi. "Scrum of scrums solution for large size teams using scrum methodology". In: *arXiv preprint arXiv:1408.6142* (2014).

[71] Per Runeson and Martin Höst. "Guidelines for conducting and reporting case study research in software engineering". In: *Empirical software engineering* 14 (2009), pp. 131–164.

[72] Johnny Saldaña. "The coding manual for qualitative researchers". In: *The coding manual for qualitative researchers* (2021), pp. 1–440.

[73] RC Sandeep, Mary Sánchez-Gordón, Ricardo Colomo-Palacios, and Monica Kristiansen. "Effort estimation in agile software development: a exploratory study of practitioners' perspective". In: *Lean and Agile Software Development: 6th International Conference, LASD 2022, Virtual Event, January 22, 2022, Proceedings*. Springer. 2022, pp. 136–149.

[74] Harjanto Prabowo Santo Fernandi Wijaya and Raymondus Raymond Kosala. "Development of an Agile Erp Framework for Implementation: A Systematic". In: *International Journal of Control and Automation* 12.5 (2019), pp. 1–12.

[75] Inc. Scaled Agile. *Achieving Business Agility with SAFe® 5.0. A Scaled Agile, Inc. White Paper*. 2019. URL: https://www.scaledagileframework.com/safe-5-0-white-paper/ (visited on 02/28/2023).

[76] Christoph Tobias Schmidt, Srinivasa Ganesha Venkatesha, and Juergen Heymann. "Empirical insights into the perceived benefits of agile software engineering practices: A case study from SAP". In: *Companion Proceedings of the 36th International Conference on Software Engineering*. 2014, pp. 84–92.

[77] Ken Schwaber and Mike Beedle. *Agile software development with scrum. Series in agile software development*. Vol. 1. Prentice Hall Upper Saddle River, 2002.

[78] Ken Schwaber and Jeff Sutherland. *The 2020 Scrum Guide*. 2020. URL: https://scrumguides.org/scrum-guide.html (visited on 02/23/2023).

[79] Ken Schwaber and Jeff Sutherland. "The scrum guide". In: *Scrum Alliance* 21.1 (2011), pp. 1–38.

[80] Tina Schweighofer, Andrej Kline, Luka Pavlic, and Marjan Hericko. "How is effort estimated in agile software development projects?" In: *SQAMIA*. 2016, pp. 73–80.

[81] Maung K Sein, Ola Henfridsson, Sandeep Purao, Matti Rossi, and Rikard Lindgren. "Action design research". In: *MIS quarterly* (2011), pp. 37–56.

[82] Martin Shepperd and Steve MacDonell. "Evaluating prediction systems in software project estimation". In: *Information and Software Technology* 54.8 (2012), pp. 820–827.

[83] Pantjawati Sudarmaningtyas and Rozlina Mohamed. "A Review Article on Software Effort Estimation in Agile Methodology." In: *Pertanika Journal of Science & Technology* 29.2 (2021).

[84] Jeff Sutherland and Scrum Inc. *The Scrum At Scale® Guide*. 2022. URL: https://www.scrumatscale.com/scrum-at-scale-guide-online/ (visited on 02/28/2023).

[85] David Talby and Yael Dubinsky. "Governance of an agile software project". In: *2009 ICSE Workshop on Software Development Governance*. IEEE. 2009, pp. 40–45.

[86] Binish Tanveer. "Guidelines for utilizing change impact analysis when estimating effort in agile software development". In: *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering*. 2017, pp. 252–257.

[87] Binish Tanveer, Liliana Guzmán, and Ulf Martin Engel. "Effort estimation in agile software development: Case study and improvement framework". In: *Journal of Software: Evolution and Process* 29.11 (2017), e1862.

[88] Binish Tanveer, Liliana Guzmán, and Ulf Martin Engel. "Understanding and improving effort estimation in agile software development: An industrial case study". In: *Proceedings of the international conference on software and systems process*. 2016, pp. 41–50.

[89] Binish Tanveer, Anna Maria Vollmer, and Ulf Martin Engel. "Utilizing change impact analysis for effort estimation in agile development". In: *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE. 2017, pp. 430–434.

[90] Qawaf Tareq. *Avoiding the Most Common ERP Challenges with Agile Methodologies*. 2016.

[91] Ömer Uludag, Martin Kleehaus, Christoph Caprano, and Florian Matthes. "Identifying and structuring challenges in large-scale agile development based on a structured literature review". In: *2018 IEEE 22nd International Enterprise Distributed Object Computing Conference (EDOC)*. IEEE. 2018, pp. 191–197.

[92] Ömer Uludağ, Pascal Philipp, Abheeshta Putta, Maria Paasivaara, Casper Lassenius, and Florian Matthes. "Revealing the state-of-the-art in large-scale agile development: A systematic mapping study". In: *arXiv preprint arXiv:2007.05578* (2020).

[93] Ömer Uludağ, Abheeshta Putta, Maria Paasivaara, and Florian Matthes. "Evolution of the agile scaling frameworks". In: *Agile Processes in Software Engineering and Extreme Programming: 22nd International Conference on Agile Software Development, XP 2021, Virtual Event, June 14–18, 2021, Proceedings*. Springer International Publishing Cham. 2021, pp. 123–139.

[94] Muhammad Usman, Jürgen Börstler, and Kai Petersen. "An effort estimation taxonomy for agile software development". In: *International Journal of Software Engineering and Knowledge Engineering* 27.04 (2017), pp. 641–674.

[95] Muhammad Usman, Ricardo Britto, Lars-Ola Damm, and Jürgen Börstler. "Effort estimation in large-scale software development: An industrial case study". In: *Information and Software technology* 99 (2018), pp. 21–40.

[96] Muhammad Usman, Emilia Mendes, and Jürgen Börstler. "Effort estimation in agile software development: a survey on the state of the practice". In: *Proceedings of the 19th international conference on Evaluation and Assessment in Software Engineering*. 2015, pp. 1–10.

[97] Muhammad Usman, Emilia Mendes, Francila Weidt, and Ricardo Britto. "Effort estimation in agile software development: a systematic literature review". In: *Proceedings of the 10th international conference on predictive models in software engineering*. 2014, pp. 82–91.

[98] Marcel Van Oosterhout, Eric Waarts, and Jos van Hillegersberg. "Change factors requiring agility and implications for IT". In: *European journal of information systems* 15.2 (2006), pp. 132–145.

[99] Xiaofeng Wang, Kieran Conboy, and Oisin Cawley. ""Leagile" software development: An experience report analysis of the application of lean approaches in agile software development". In: *Journal of Systems and Software* 85.6 (2012), pp. 1287–1299.

[100] Laurie Williams and Alistair Cockburn. "Agile software development: It's about feedback and change". In: *Computer* 36.6 (2003), pp. 39–43.

[101] Robert K Yin. *Case study research: Design and methods*. Vol. 5. sage, 2009.

[102] Shahid Kamal Tipu Ziauddin and Shahrukh Zia. "An effort estimation model for agile software development". In: *Advances in computer science and its applications (ACSA)* 2.1 (2012), pp. 314–324.