UNIVERSITY OF TWENTE.

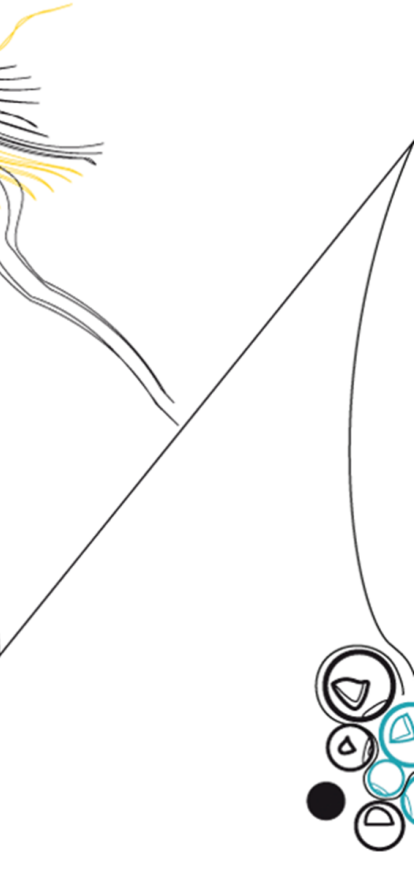
Faculty of Electrical Engineering, Mathematics & Computer Science

How Can (Large Scale) Agile be Effectively Adopted and Scaled Up in Dutch Public Sector Organisations

W.T.C. Bolhuis

M.Sc. Thesis

September 2021



Supervisors:
prof.dr. J. van Hillegersberg
dr. M. Daneva

Master Business Information Technology
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Summary

The goal of this paper is to aid Dutch public sector organisations in making more effective use of (large scale) agile. This research was executed in two steps inspired by the design science research methodology (DSRM). The first main step was a literature review of existing scientific and non-scientific literature on (large scale) agile. The second step was a multiple case study in which members of four Dutch public sector project, programmes, or organisations were interviewed.

The current scientific literature on (large scale) agile and (large scale) agile related subjects is very minimal as it is still a very young subject. Eight (large scale) agile approaches have been defined and discussed in the literature study of which two were encountered during the multiple case study.
A total of 20 success factors and 36 challenges of (large scale) agile were found split over eight and ten categories respectively.

A total of 22 employees split over four different cases were interviewed. 21 of these participants filled in a 17 question survey to gain a first insight into the current state of (large scale) agile in their (sub)organisation. Participants from all different roles within (large scale) agile project, programmes, or organisations were interviewed.
Based on the survey results, participants seemed to be motivated in their work and think (large scale) agile was fitting to their work. However they also felt the way of working in their (sub)organisation could be improved. As could management support, communication, and knowledge sharing.
Based on the interview results, short iterations, constant steering, customer involvement, and close communication were found as the strongest points of (large scale) agile. On the other hand within the cases a mismatch between the organisation and (large scale) agile, a distance to the user, and a lack of trust were the most commonly raised issues.

In general all participants of the multiple case study found (large scale) agile would be a fitting way of working to their (sub)organisation. This paper finds an effective (large scale) agile adoption is achieved when a Dutch public sector organisation has a willingness to change, makes use of (large scale) agile iterative evaluation tools, is well informed and trained, has executive sponsorship for a transition, and

is mindful about their transition. Eight core values of (large scale) agile are defined which should form a strong basis for a successful (large scale) agile adoption within Dutch public sector (sub)organisations.

Dutch public sector organisations seem to lack the right experience and knowledge of (large scale) agile to effectively adopt (large scale) agile in a broader sense. However, participants of the multiple case study indicate this could greatly benefit the Dutch public sector (sub)organisations. There seems to be a lacking urgency from the side of higher and middle management to adjust the way of working to more closely support the goals of current projects, programmes, or organisations. Especially the (large scale) agile principle to frequently adjust not only the product and the steps to get there, but also the utilised processes is seen as a major strength of (large scale) agile.

Even though utilisation of (large scale) agile would be beneficial to Dutch public sector project, programmes, and organisations, it should always be secondary to the goals of said project, programme, or organisation. The way of working chosen should be in line with the main goals of a project, programme, or organisations and should be able to add onto existing processes and roles within the project, programme, or organisation.

To achieve a more complete and effective way of working, only one (large scale) agile approach should be chosen as the basis of the way of working. However, this could be expanded upon by processes, tools, and/or roles from other (large scale) agile approaches.

Contents

Summary iii

List of acronyms ix

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Research context . . . 2
- 1.3 Goal of the research . . . 2
- 1.4 Structure of the paper . . . 2
- 1.5 Defining (large) scale . . . 3

2 Methodology 5

3 Background 7
- 3.1 Methodology . . . 7
- 3.2 Large scale agile . . . 8
- 3.3 Success factors and challenges of (large scale) agile . . . 11
- 3.4 Adoption of (large scale) agile . . . 14
  - 3.4.1 3.4-CS-1 . . . 15
  - 3.4.2 3.4-CS-2 . . . 16
  - 3.4.3 3.4-CS-3 . . . 17
  - 3.4.4 3.4-CS-4 . . . 18
  - 3.4.5 3.4-CS-5 & 3.4-CS-6 . . . 20
  - 3.4.6 Summarizing . . . 21
- 3.5 DevOps and (large scale) agile . . . 22
  - 3.5.1 scaled agile framework (SAFe) and DevOps . . . 23
  - 3.5.2 disciplined agile delivery (DAD) and DevOps . . . 23
  - 3.5.3 large scale scrum (LeSS) and DevOps . . . 24
  - 3.5.4 Comparison . . . 24
- 3.6 Comparing public and private sector IT projects . . . 24
- 3.7 Common (large scale) agile research approaches . . . 28

4 Case study **33**
4.1 Goal . . . 33
4.2 Methodology . . . 33
4.2.1 Interview analysis . . . 34
4.3 Design . . . 35
4.3.1 Survey . . . 35
4.3.2 Interviews . . . 35
4.3.3 Data compliance . . . 36
4.4 Cases . . . 36
4.4.1 Case 1 . . . 37
4.4.2 Case 2 . . . 38
4.4.3 Case 3 . . . 38
4.4.4 Case 4 . . . 39
4.5 Survey results . . . 39
4.6 Interview results . . . 41
4.6.1 Challenges and success factors within the cases . . . 41
4.6.2 Challenges and success factors within Dutch public sector . . . 42
4.6.3 Fitting (large scale) agile to Dutch public sector . . . 44
4.6.4 Core values of (large scale) agile . . . 45
4.6.5 Achieving a successful adoption . . . 47
4.6.6 Additional findings from the interviews . . . 50

5 Discussion and evaluation **55**
5.1 Initial conclusions . . . 55
5.1.1 How is (large scale) agile used in Dutch public sector . . . 55
5.1.2 What are the benefits of (large scale) agile compared to traditional project management . . . 56
5.1.3 Would (large scale) agile be a good addition to Dutch public sector organisations . . . 57
5.1.4 How can (large scale) agile be effectively adopted and scaled up in Dutch public sector organisations . . . 58
5.2 Evaluation . . . 61
5.2.1 Evaluation method . . . 61
5.2.2 Evaluation results . . . 63
5.3 Discussion . . . 63

6 Conclusion **69**
6.1 How can (large scale) agile be effectively adopted and scaled up in Dutch public sector organisations? . . . 69

6.2 What can Dutch public sector organisations do to more effectively adopt and scale up (large scale) agile? 72
6.2.1 Utilising (large scale) agile 72
6.2.2 Steering (large scale) agile 72
6.2.3 Training (large scale) agile 73
6.2.4 Managing (large scale) agile 73
6.2.5 Implementing (large scale) agile frameworks 73
6.2.6 Do not use (large scale) agile 74
6.3 Generalisability of findings 74
6.4 Limitations 76
6.5 Implications for research 77
6.5.1 Potential research questions 78
6.6 Implications for practice 79

References **81**

Appendices

A Search terms **89**

B Papers used **91**

C Glossary **93**

D Survey questions **95**

E Codebook **97**
E.1 Codebook demographics 97
E.2 Codebook Success Factors 99
E.3 Codebook Challenges 100
E.4 Codebook Additions 102

F Survey results **103**
F.1 Full survey results 103
F.2 Survey question correlation 104
F.3 Survey averages 104

G Success factor & challenge occurrences in case study **105**

List of acronyms

SAFe	scaled agile framework
DSRM	design science research methodology
XP	extreme programming
DAD	disciplined agile delivery
LeSS	large scale scrum
RAGE	recipes for agile governance in the enterprise
S@S	scrum@scale
PO	product owner
PI	program increment
WSJF	weighted shorted job first
LeSS Huge	large scale scrum huge
SoS	scrum of scrums
SoSoS	scrum of scrum of scrums
APO	area product owner
BPC	business process change

Chapter 1

Introduction

1.1 Motivation

After the introduction of agile in 2001 [8], many small IT development teams used these new principles to increase the efficiency and end-results of their projects. However, agile is only aimed towards small scale development teams. According to the Scrum Guide a development team should never contain less than three and especially not more than nine members [61]. Not every development project can be completed by a team of only nine people, therefore the idea of (large scale) agile was introduced. With (large scale) agile the main principles of agile could be applied on, as the name implies, a larger scale. Different (large scale) agile methodologies introduce various additions to existing agile methodologies to enable successfully scaling up agile.

In 2020 the Dutch government (Rijksoverheid) worked on 125 projects with a budget over five million euros, 62% of those projects ran over their initial planning, and there is a 20% increase in total costs over all these projects [41]. In 2018 I wrote a paper about the implementation of (large scale) agile within the Dutch public sector [10]. In this paper I talked with multiple experienced consultants who were currently working, or have previously worked, on big Dutch public sector IT projects. Among other things this paper concludes that the application of (large scale) agile would be beneficial to the Dutch public sector.
Slowly but steadily Dutch public sector organisations are adopting (large scale) agile methodologies [14, 69].
This paper is intended as a follow-up of the initial Bolhuis [10] paper to help Dutch public sector organisations more effectively adopt (large scale) agile.

1.2 Research context

This research was split up in two steps. First a literature study was executed at the University of Twente to gain an insight in the current scientific knowledge of (large scale) agile and IT-projects within Dutch public sector organisations.

The second main step of this research was a multiple case study executed with the support of Rijkswaterstaat, a Dutch government agency. Four Dutch public sector cases were gathered with the help of Rijkswaterstaat. The research was supported by a (large scale) agile working group within Rijkswaterstaat. This working group helped point the research towards what could potentially support them in their (large scale) agile adoption. Cases were gathered both inside and outside Rijkswaterstaat through the network of the members of the working group.

1.3 Goal of the research

In this paper I will attempt to provide Dutch public sector organisation with additional tools and knowledge to achieve an effective adoption of (large scale) agile. In their 2019 paper, Conboy and Carroll [12] find there is a lacking comparison between available (large scale) agile methodologies. Therefore, this paper will attempt to provide this comparison and aid (public sector) organisations with a tool to help them find the right methodology or combination of methodologies to apply in different situations. This tool will be in the form of an adoption model. This adoption model will allow (public sector) organisations to formulate a suitable (potentially hybrid) large scale agile methodology adjustable to different specific circumstances or situations.

The main question to be answered in this paper is the following: *How can (large scale) agile be effectively adopted and scaled up in Dutch public sector organisations?*

1.4 Structure of the paper

This research is executed in two main steps. First of all, Chapter 3 encompasses a literature study aimed to discover the current state of knowledge of (large scale) agile. In Chapter 4 four cases are discussed in a multiple case study to discover the current state of (large scale) agile within the Dutch public sector.
The findings of these two chapters will be discussed and evaluated in Chapter 5. Finally, in Chapter 6 the conclusions, limitations, and recommendations of this research are given.

1.5 Defining (large) scale

There is a wide variety of large scale agile frameworks or approaches each with their own definition of (large) scale. For example, SAFe starts using its scaling tools from five teams onward. On the other hand, scrum@scale (S@S) adds additional tools starting at two teams. As scale and large scale are difficult to define, this paper will formulate it as *(large scale) agile* . This way the term should encompass the main ideas behind (large scale) agile without raising a discussion on what would or would not be large scale.

Chapter 2

Methodology

To create the adoption model, the DSRM as designed by Peffers et al. [47] will be used. The DSRM consists of the following six activities:

1. Problem identification and motivation
2. Define the objectives for a solution
3. Design and development
4. Demonstration
5. Evaluation
6. Communication

During the first activity of the DSRM the problem is identified and a potential solution to this problem is motivated. To collect enough information to be able to identify the problem and motivate potential solutions I will execute a literature study. This literature study should provide the information to be able to identify general problems of large scale agile in the public sector. Besides, the literature study should provide an insight in the direction of the potential solution.
To expand on the theoretical knowledge from the scientific literature, practical knowledge will be gathered through a multiple case study. Four cases will be analysed to confirm the problem found through the literature study and as the main source of information to motivate the solution to the problem.

The second activity of the DSRM encompasses the definition of objectives of the solution. These objectives are derived rationally from the problem definition as determined during the first activity. The majority of these objectives will be derived from the case studies and where possible these will be added upon by scientific literature.

After the definition of objectives, enough information is gathered and the actual solution can be designed and created. The solution will be some form of artefact, Peffers et al. [47] mention for example models, constructs, or methods. As a start the functionality and architecture of the artefact should be created before the artefact itself is created.

The next two activities of the DSRM ensure the validation and verification of the artefact. First, during the fourth activity, the artefact is demonstrated in for example an experiment, simulation, or case study. The fifth activity involves the evaluation of the demonstration, this evaluation is performed in relation to the objectives as defined during the second activity. Based on the results from these two activities the artefact can be adjusted similarly to the third activity after which these activities will be performed again. Due to the time scope of this research it is uncertain to which extend these activities are possible.

The final activity of the DSRM is the communication of the artefact to relevant stakeholders. The artefact will be communicated through two channels. The first channel is this paper intended to communicate the artefact to relevant researchers. The second channel is communication of the artefact to the involved and relevant public sector organisations, including the organisations featured in the case study chapter.

The methodologies of the literature research and the multiple case study research will be discussed in chapter 3.1 and chapter 4.2 respectively.

Chapter 3

Background

3.1 Methodology

During the first activity of the DSRM the research problem should be defined and the value of the proposed solution should be justified [47]. To achieve this a literature study will be performed.

This chapter will feature the background literature study. The intention of this chapter is to identify the background and the context of this paper. This background chapter will attempt to answer the following research questions:

1. What is large scale agile?
2. What are the success factors and challenges of large scale agile implementations?
3. How is large scale agile adopted in large scale IT-projects?
4. How can large scale agile be adopted in combination with DevOps?
5. How do IT projects executed in the public sector compare to IT projects in the private sector?
6. How is the applicability of large scale agile researched in large scale IT-projects?

These questions will be answered through a systematic literature review. Papers are collected using *Scopus* and relevant citations in the papers collected from *Scopus*. Papers will be limited to the fields of *Computer Science* and *Business, Management and Accounting* and need to have been written in the last ten years to ensure relevancy. Only peer reviewed or conference papers will be utilised as these have a clear and demonstrable level trustworthiness. There is a preference for review papers as these present a more balanced view by combining multiple papers in one. Papers are sorted by citations and papers with less than ten citations are excluded from the search.

For each research question search terms have been defined. From all papers that passed the initial criteria papers were selected that seemed to actually answer

the research question. However, not all these papers turned out to contribute to the research questions they served, so not all of those selected papers were used in this paper. In appendix A the different search terms used are set out together with the search results, the amount of papers selected, and the amount of papers used. Secondly, a percentage is included to determine the percentage of papers selected from the total search results, and the percentage of papers used from the selection. The search terms used are included in order of usage. Multiple search terms could result in the same papers being found, papers found through multiple search terms are included in the metrics for all relevant search terms. Some search terms did not result in any new papers being selected, however it could have overlapped with previous search terms, these search terms are still included in the list for completeness, however the metrics are not tracked.
Papers found through the search terms are selected based on the title and the abstract. After initial papers are selected the introduction and conclusion is analysed to determine if they are applicable to the subject.

Not all information relating to (large scale) agile is accessible through existing scientific literature. Therefore, papers or white papers from the industry will also be utilised. These white papers are gathered through the websites of the major (large scale) agile methodologies or through websites of major consultancy organisations with expertise on (large scale) agile. The amount of white papers, or other sources, used to answer the research questions can also be seen in appendix A.

To answer the research questions some literature is simply used to answer the question, on the other hand some literature is used to provide more background information. Appendix B presents an overview of which references have been used to answer each research question in Chapter 3.

3.2 Large scale agile

As the name states (large scale) agile is a scaled up version of the agile methodology. The agile methodology was created in 2001 by Beck et al. [8] and consists of twelve principles. Agile software development mainly focuses on adaptability, flexibility, continuous delivery of working software, small development teams, and close contact between team members. The original principles of agile are aimed towards smaller teams which makes it very difficult to implement agile in large IT-projects as-is.

Major implementations of agile are Scrum, extreme programming (XP), Kanban, and Lean [71].
In their 2016 paper, Alqudah and Razali [1] identify and discuss seven approaches to scaling agile. In the case study in chapter 4 S@S was frequently used, therefore

this will be discussed as well:

- DAD
- SAFe
- LeSS
- large scale scrum huge (LeSS Huge)
- Spotify
- Nexus
- recipes for agile governance in the enterprise (RAGE)
- S@S

DAD, SAFe, Spotify, and RAGE are created based on a combination of a few different existing agile implementations. LeSS, LeSS Huge, Nexus, and S@S are all scaled up based on Scrum. All these methodologies feature a common approach to (large scale) agile, teams of teams, where scrum of scrums (SoS) teams are introduced in which multiple scrum teams are working together.

In 2012 the first full version of DAD was released together with the first DAD book *Disciplined agile Delivery* [3]. In the three years before this, DAD was developed by IBM before being released as a standalone agile toolkit. In 2020 the latest version of DAD was released, DAD 5, together with the new DAD book *Choose Your Wow!* [4]. DAD 5 consists of four different layers: Foundation, Disciplined DevOps, Value Streams, and Disciplined agile Enterprise, each layer adding to the layer below [2].

SAFe was originally based on the book *Scaling software agility* by Leffingwell [34] and its successor *agile software requirements*[36], the first version of SAFe was released in 2011 [27]. In 2019 the latest version of SAFe was released, SAFe 5.0, consisting of four different configurations: Essential, Portfolio, Large Solution, and Full, each configuration designed for different types of organisations or projects [58].

In 2005, Larman and Vodde [30] started development of LeSS based on their experiences in various (agile) projects. Larman and Vodde wrote multiple books about LeSS, with the latest book, *Large Scale Scrum*, being released in 2017 [31]. There is a significant difference in scaling between LeSS and LeSS Huge. LeSS has up to eight Scrum teams working together on the same product. Unlike DAD and SAFe, LeSS does not add any additional roles on top of the basic roles in Scrum [32]. In LeSS Huge these LeSS teams work together in so-called *Requirement Areas* with area based product owner (PO) called area product owner (APO), basically running multiple instances of LeSS alongside each other [33].

The Spotify agile approach is, as the name suggests, based on the working methods of the Swedish company Spotify. The approach was documented in the 2012 article *Scaling agile @ Spotify* by Kniberg and Ivarsson [28]. Within Spotify everything is split up in Squads. Each Squad autonomously works on their own part of the application. Squads working on related areas are grouped in Tribes and the

Squads in a Tribe are physically located close to each other. Tribes are not allowed to become bigger than 100 members to prevent bureaucracy. Tribe Members of different Squads with similar roles work together in Chapters to ensure the sharing of knowledge. Knowledge is also share outside of Tribes in Guilds which feature employees with similar roles throughout the entire organisation [28].

Nexus is a scaled version of Scrum, originally released in 2015 by Scrum creator Schwaber [60] and the Scrum organisation, last updated in 2018. Nexus allows for nine Scrum teams. Work is integrated by a special Nexus Integration Team, consisting of team members of the regular Scrum teams. On the level of the Nexus Integration team the main Product Backlog, Sprint planning, and coordination is handled before the individual Scrum teams translate this into their own planning [60].

RAGE was theorised in 2013 by Thompson [68] in the paper *Recipes for Agile Governance in the Enterprise*. As the name implies RAGE has a focus on the governance side of an organisation and splits the governance in three with project management, program management, and portfolio management. Thompson discusses different principles of agile and based on a case study Thompson defines *Anti-Patterns*, behavioural patterns that are counterproductive. RAGE is not so much a direct approach to scaling agile, but more of a guide on which problems can arise and how to resolve these [68].

Lastly, S@S was introduced in 2006 by Sutherland [67] to introduce linear scalability and business agility in the traditional ways of *Scrum*. S@S introduces a SoS which operates like a regular scrum team, however the team members of the scrum team are the 'normal' scrum teams. Each SoS team should consist of 4-5 scrum teams as this was discovered to be the ideal team size. S@S adds the role of SoS master and the chief product owner, both scaled roles on the original scrum master and product owner role. These are the scrum master and product owner of the SoS team. Once an organisation becomes too large *Scrum of Scrum of Scrums* teams are introduced, similarly to the SoS teams, these scrum of SoS teams would consist of 4-5 SoS teams. This way S@S could be scaled up infinitely [67].

All of these approaches have a multitude of things in common. Sometimes these features get different names, but the principles remain the same. In appendix C a glossary of the different features used by these approaches and their definition can be found, terms might differ between approaches but for the purpose of this paper the terminology in appendix C is used consistently to describe features.

3.3 Success factors and challenges of (large scale) agile

To answer this question I will take a look at various pieces of literature identifying success factors and challenges of (large scale) agile, and combine these to get a clear overview of the success factors and challenges of (large scale) agile. In table 3.1 the success factors of (large scale) agile can be found, sources that report the success factors are included in the description, each benefit receives its own code so they can be easily referenced to. Similarly in table 3.2 the challenges of (large scale) agile can be found.

In their 2019 paper Conboy and Carroll [12] identify nine challenges of (large scale) agile through a review of 13 case studies implementing various different (large scale) agile approaches spread over 15 years. Various different frameworks are used in the different case studies [12].

Putta et al. [54] performed a review on 54 case studies, from both peer-reviewed literature and grey literature, to identify both benefits and challenges of adopting SAFe. All grey literature used by Putta et al. originates from the SAFe website, the results from these case-studies might be biased or exaggerated, to ensure as much impartiality as possible I will not include any benefits or challenges that are solely presented by grey literature. There is a total of 14 challenges reported in grey literature, 11 of these challenges are reported in both grey and peer-reviewed literature, one additional challenge is only reported by the peer-reviewed literature [54].

Kalenda et al. [25] identify eight practices, nine challenges, and seven success factors of (large scale) agile based on a review of twelve pieces of literature. These are later applied to analyse a (large scale) agile application [25].

Shameem et al. [63] perform a systematic review on 20 articles to determine 15 success factors of global software development. Shameem et al. compare the frequency of these success factors between the client perspective and the vendor perspective [63].

In 2011 van Hillegersberg et al. [70] report seven challenges of global distributed (large scale) agile, based on two early pieces of literature. With most (large scale) agile frameworks being released after 2011 the challenges identified by van Hillegersberg et al. might not be as relevant with the current day frameworks anymore [70].

In May 2020 the fourteenth version of the *State of Agile report* was published by State of Agile [65], this survey reports, among others, on the challenges of scaling agile. The report is based on a survey filled in by 1121 individuals from a broad range of global software development industries [65].

Based on these six sources, 20 success factors and eight success categories of

Code	Category	Success Factor
SF1	Communication	**SF1.1** Communication, coordination, and control (3Cs) [63] **SF1.2** Close connections & constant communication between teams [25] **SF1.3** Effective customer involvement [63] **SF1.4** Encourage project visibility [63]
SF2	Acquire knowledge	**SF2.1** Train and coach people [25, 63] **SF2.2** Acquire knowledge on (large scale) agile [25] **SF2.3** Encourage knowledge sharing [63]
SF3	Infrastructure	**SF3.1** Utilise agile supporting tools & a strong technological infrastructure [25, 63] **SF3.2** Effective requirements analysis [63]
SF4	Practices	**SF4.1** Ensure strong engineering practices to achieve as little reduction of quality as possible [25] **SF4.2** Short iterations [63]
SF5	Team (members)	**SF5.1** Small team size [63] **SF5.2** Self-organising teams [63] **SF5.3** Experienced and motivated developers [63] **SF5.4** Conduct social events [63]
SF6	Transitioning	**SF6.1** Transitioning slowly and carefully [25] **SF6.2** Executive sponsorship [25]
SF7	Management support	**SF7.1** Management commitment [63] **SF7.2** Effective leadership [63]
SF8	Unification	**SF8.1** Define a common view on the change and terminology [25]

Table 3.1: Success factors of (large scale) agile

Code	Category	Challenge
C1	Team issues	**C1.1** Minimal collaboration & knowledge sharing between teams [25, 65, 70] **C1.2** Teams lacking autonomy [12, 54] **C1.3** Lack of trust [25, 70] **C1.4** Team sizes [70] **C1.5** Increased stress and pressure [25]
C2	Organisational issues	**C2.1** Difficulty scaling agile to non-IT departments [25, 54] **C2.2** Controversies between organisational and agile culture [12, 65] **C2.3** Mismatch between agile and client organisation [12, 65] **C2.4** Large scale agile approach not applicable due to regulations [65]
C3	Globalisation	**C3.1** Physical distance between team members [25, 70] **C3.2** Synchronising communications [70] **C3.3** Having multiple communication channels [70] **C3.4** Difficulty scaling agile to distributed organisations [54]
C4	Transitioning issues	**C4.1** Difficulty measuring the results of a transition [12, 25] **C4.2** Difficulty adopting agile mindset [54] **C4.3** Determining readiness of the organisation to transition [12] **C4.4** Difficulty initiating from both the top and bottom of the organisation [12] **C4.5** Struggles moving from fixed delivery to iterative delivery [54]
C5	Resistance towards change	**C5.1** Resistance towards change [25, 54, 65] **C5.2** Resistance towards new roles [54] **C5.3** Afraid of consequences of increased transparency [25]
C6	Framework issues	**C6.1** Inconsistent terminology used in frameworks with different ways to interpret [12, 65] **C6.2** Scaling agile feels like moving away from agile [54] **C6.3** Contradictions within the framework [54]
C7	Lack of information	**C7.1** Lack of training [25, 65] **C7.2** Lack of skills or experience with agile [25, 65] **C7.3** Little available literature resulting in unknown or unsolved issues [12, 65] **C7.4** Lack of comparison between frameworks [12]
C8	Practice specific issues	**C8.1** Issues with the first instance of a practice [54] **C8.2** Issues with integration [54] **C8.3** Difficulty automating testing [54] **C8.4** Difficulty scaling requirements management [25]
C9	Management issues	**C9.1** Lacking leadership participation [65] **C9.2** Lacking management support and sponsorship [65]
C10	Quality	**C10.1** Problems with quality assurance [25] **C10.2** Loss of quality [25]

Table 3.2: Challenges of (large scale) agile

(large scale) agile have been defined in table 3.1. In table 3.2 36 challenges and ten challenge categories of (large scale) agile have been defined. There is an overlap between the challenges and success factors, this is logical as success factors can be used to prevent certain challenges and avert any issues which might arise from the challenges. However, some challenge categories still remain unsolved by the success factors. Especially *Organisational issues (C2)*, *Globalisation (C3)*, *Transitioning issues (C4)*, and *Resistance towards change (C5)* go unsolved through the success factors. Both tables are ordered based on the total amount of different papers attributing to a certain group of success factors or challenges. A few challenges and success factors are logically related to each other or are direct opposites of one another. Below is a list of success factors and their directly opposing challenge.

- Training people: *Train and coach people - SF2.1* & *Lack of training - C7.1*
- Knowledge of agile: *Acquire knowledge on (large scale) agile - SF2.2* & *Lack of skills or experience with agile - C7.2*
- Knowledge sharing: *Encourage knowledge sharing - SF2.3* & *Minimal collaboration and knowledge sharing between teams - C1.1*
- Team size: *Small team size - SF5.1* & *Team sizes - C1.4*
- Team autonomy: *Self-organising teams - SF5.2* & *Teams lacking autonomy - C1.2*
- Executive sponsorship: *Executive sponsorship - SF6.2* & *Lacking management support and sponsorship - C9.2*
- Leadership participation: *Effective leadership - SF7.2* & *Lacking leadership participation - C9.1*
- Unification of terminology: *Define a common view on the change and terminology - SF8.1* & *Inconsistent terminology used in frameworks with different ways to interpret - C6.1*

Besides the papers identified before, Stojanov et al. [66] created a maturity model for the adoption of agile and SAFe practices. Due to the focus of this maturity model on SAFe it is not incorporated in the overview of general (large scale) agile success factors and challenges. However, this maturity model can be useful when reviewing a transition to SAFe. A (shortened) version of the model can be found in Table I of the Stojanov et al. [66] paper.

3.4 Adoption of (large scale) agile

To find out how (large scale) agile is adopted in different situations, I will review various case studies based on the identified success factors and challenges. These case studies will be discussed and practices within these case studies will be mapped to the identified success factors or challenges. In table 3.3 the different case studies

Code	Paper	Industry	Year	Approach	Note
3.4-CS-1	[44]	Global software development	2007	SoS	Distributed teams
3.4-CS-2	[70]	Global software development	2004	No framework	
3.4-CS-3	[29]	Telecommunications	2012	No framework	Meant to replace an existing waterfall project
3.4-CS-4	[43]	Telecommunications	2013	LeSS Huge	
3.4-CS-5	[42]	Software development	2015	SAFe	Compared in the same paper with 3.4-CS-6
3.4-CS-6	[42]	Software development	2016	SAFe	Compared in the same paper with 3.4-CS-5

Table 3.3: Case studies used in chapter 3.4

used are set out and given a code. The industry of the case organisation, the year the transition started, and the (large scale) agile approach used are included.

3.4.1 3.4-CS-1

Paasivaara et al. [44] seem to be the first to write about the implementation of agile in a large organisation. With a solution development organisation of about 40 people, 3.4-CS-1 is not a particularly large organisation. However most 3.4-CS-1 teams are distributed over two countries (*Physical distance between team members - C3.1*). The 40 members are split over 7 teams of each two to nine members (*Small team size - SF5.1* to avert *Team sizes - C1.4*), the sprints are synchronised between the various teams (preventing *Synchronising communications - C3.2*). 3.4-CS-1 uses Jira to keep track of their backlog and has a Centralised Version Control system (*Utilise agile supporting tools & a strong technological infrastructure - SF3.1*).

When the distributed office was founded two members of the original office moved to the distributed site to support the team members at the distributed office and more support was provided during the first iterations (preventing things like *Issues with the first instance of a practice - C8.1*). There are frequent visits between the different locations to be able to meet face-to-face (*Encourage knowledge sharing - SF2.3* averting *Minimal collaboration and knowledge sharing between teams - C1.1*, *Struggles moving from fixed delivery to iterative delivery - C4.5*, and *Lack of skills or experience with agile - C7.2*).
Regardless of frequent visits and sharing employees between the two locations, 3.4-

CS-1 is still struggling with the challenges of globalisation, not having video conferencing tools a silence caused by distance are reported (*Physical distance between team members - C3.1*).

Secondly, the issues with communication results in a misunderstanding of requirements, this challenge was not identified in chapter 3.3, 3.4-CS-1 resolves this challenge by having product owners ask follow-up questions to developers whenever explaining new user stories. 3.4-CS-1 reports a few benefits of adopting (large scale) agile, through the adoption of (large scale) agile the overall quality was increased (no *Loss of quality - C10.2*), there was better and more communication even with the reported issues, motivation of the employees has improved (*Experienced and motivated developers - SF5.3*).

3.4.2 3.4-CS-2

Besides presenting seven challenges of (large scale) agile adoption van Hillegersberg et al. [70] also go in depth on how these challenges are averted within 3.4-CS-2. 3.4-CS-2 reports on a globally distributed software development company scaling agile not using any of the aforementioned (large scale) agile frameworks.

To aid the transition to (large scale) agile, the initial iteration length of 3.4-CS-2 was six weeks, which was later brought back to four and later to two weeks (*Transitioning slowly and carefully - SF6.1* averting *Struggles moving from fixed delivery to iterative delivery - C4.5* resulting in *Short iterations - SF4.2*). 3.4-CS-2 did not have a direct client organisation and was therefore not able to have an actual client represented during development (*Controversies between organisational and agile culture - C2.2*), this role was adopted by product marketing and later product management. 3.4-CS-2 offered many different modes of communication (*Having multiple communication channels - C3.3*, but not reported as a problem), which allowed for easy communication between the teams (*Close connections & constant communication between teams - SF1.2*).

The solution of 3.4-CS-2 was split up in many products and as soon as a product became too big it was split in multiple products again to ensure a small team size (*Small team size - SF5.1* averting *Team sizes - C1.4*). 3.4-CS-2 does not force any exact processes upon their teams, but leaves them free to find their own flow with scrum and XP (*Self-organising teams - SF5.2*). 3.4-CS-2 does not compare teams with each other, as different teams work on different products which all need different technology (*Difficulty measuring the results of a transition - C4.1*). Many team members and teams had been working together for a long time within 3.4-CS-2 which made it easier for team members and teams to predict each other and improve collaboration (*Close connections & constant communication between teams*

- SF1.2, *Experienced and motivated developers - SF5.3*).

3.4-CS-2 provided many scrum trainings to all the different offices (*Train and coach people - SF2.1*) and the 3.4-CS-2 CEO supported the transition (*Executive sponsorship - SF6.2*). Many different tools were utilised within 3.4-CS-2 (*Utilise agile supporting tools & a strong technological infrastructure - SF3.1*) and rules were included when pushing builds to ensure quality (*Ensure strong engineering practices to achieve as little reduction of quality as possible - SF4.1* to avert *Loss of quality - C10.2*).

3.4.3 3.4-CS-3

Lagerberg et al. [29] perform a comparison between two projects, a traditional project and a new agile project (3.4-CS-3), started in 2012, meant to replace the traditional solution over time.
3.4-CS-3 consists of 120 people and 14 teams (*Small team size - SF5.1*). The teams work in 3 week iterations (*Short iterations - SF4.2*) and are cross-functional. 3.4-CS-3 does not use any particular (large scale) agile framework, but bases their way of working on scrum and feature driven development. Development teams are supported by supporting teams which take care of project management, continuity, and integration.

Lagerberg et al. report a full adoption of team empowerment in 3.4-CS-3 (preventing *Teams lacking autonomy - C1.2*), there is no on-site customer (no *Effective customer involvement - SF1.3*), and mature developers with 37% of developers having more than one year of experience with agile software development (*Experienced and motivated developers - SF5.3*).
There is more knowledge sharing within and between teams in 3.4-CS-3 than in its traditional counterpart, according to the team members of 3.4-CS-3 this is enabled and improved by the agile processes (*Close connections & constant communication between teams - SF1.2* averting *Minimal collaboration and knowledge sharing between teams - C1.1*). 3.4-CS-3 has a considerably larger visibility (*Encourage project visibility - SF1.4*), iteration planning and continuous integration were credited with this increased visibility. There was a slightly higher stress level reported in 3.4-CS-3 (*Increased stress and pressure - C1.5*), but the difference with the traditional project is not statistically significant, however there does seem to be a significant increase in stress due to the demos of 3.4-CS-3.

All team members of 3.4-CS-3 work in an open office space and are thus seated next to each other (no *Physical distance between team members - C3.1*). Interestingly, Lagerberg et al. find that a partial implementation of agile methods, in the 3.4-CS-3 traditional counterpart, results in positive results, but the full implementa-

tion, in 3.4-CS-3, presents stronger positive results.

3.4.4 3.4-CS-4

Paasivaara and Lassenius [43] report on a 2,5 year journey at a global telecommunications company adopting the LeSS framework with 20 teams spread over four locations.
The transition to (large scale) agile was promoted by management (*Executive sponsorship - SF6.2*) and started with 15 developers and two teams, as more teams were added they were coached by the existing teams (*Train and coach people - SF2.1*). When the first distributed team was added they travelled to the main site to get training (*Train and coach people - SF2.1*). Training was given by one of the creators of LeSS (*Train and coach people - SF2.1*, *Acquire knowledge on (large scale) agile - SF2.2*), and this LeSS creator led the first requirements workshop (*Train and coach people - SF2.1* to avert *Issues with the first instance of a practice - C8.1*). However, at the start of the project there were no dedicated coaches, these were added later on.

New teams were added to the project slowly and only when they were needed (*Transitioning slowly and carefully - SF6.1*). 3.4-CS-4 made use of two week iterations (*Short iterations - SF4.2*). Development only happened in two locations, the other two locations housed testing teams. There were 16 development teams and four testing teams, each team consisted of 6-10 members (*Small team size - SF5.1*). 3.4-CS-4 had 9-10 APOs, each APO actually consisted of two people, one system architect and one solution architect, presumably to increase the available knowledge (*Experienced and motivated developers - SF5.3*). Most APOs were located at the main site, which made it more difficult to communicate with the remote teams (*Physical distance between team members - C3.1*, *Difficulty scaling agile to distributed organisations - C3.4*). 3.4-CS-4 made use of a virtual maintenance team which featured 1-2 members of each team, this might make the virtual maintenance team a large team (*Team sizes - C1.4*), but it does ensure a broad spectrum of knowledge to resolve issues (*Close connections & constant communication between teams - SF1.2*). Almost all of the line managers in the organisation had a double role, so they were more involved with the work being done (*Effective leadership - SF7.2*). Paasivaara and Lassenius go more in-depth to the double APO role, this was decided by 3.4-CS-4 as features could not be mapped to one single product area, therefore it was decided to divert from the original idea of APO (*Controversies between organisational and agile culture - C2.2*).

The main backlog of 3.4-CS-4 was in the form of an excel sheet (no *Utilise agile supporting tools & a strong technological infrastructure - SF3.1*) and the division

was quite chaotic with different teams working concurrently on the same feature. The collaboration and communication between APOs and teams was challenging (weak *Communication, coordination, and control - SF1.1*). Another identified issue were user stories being too big to be able to be developed in one iteration (*Lack of skills or experience with agile - C7.2*).
A common sprint planning meeting was held at the start of each iteration. Team members of all teams could attend this meeting (*Encourage project visibility - SF1.4*), however most teams only sent one person and some team members saw it as a waste of time (could point towards *Resistance towards change - C5.1* or *Afraid of consequences of increased transparency - C5.3*). 3.4-CS-4 had daily SoS meetings (*Encourage project visibility - SF1.4*), however these meetings were split between the main site and the global sites (*Physical distance between team members - C3.1*, *Synchronising communications - C3.2*, *Difficulty scaling agile to distributed organisations - C3.4*). Many teams attending the SoS stated they had ”nothing to report” even though that might not have been true (might point towards *Afraid of consequences of increased transparency - C5.3*).
3.4-CS-4 implemented a common sprint demo, which did not look much like an actual demo, all teams got ten minutes to present their achievements. Many team members criticised this approach as it did not accurately portray achievements and results, management did believe this was a good approach (attempt at *Encourage project visibility - SF1.4*, but due to mismanagement resulting in *Minimal collaboration and knowledge sharing between teams - C1.1*, *Difficulty adopting agile mindset - C4.2*, *Struggles moving from fixed delivery to iterative delivery - C4.5*, *Lack of skills or experience with agile - C7.2*, *Lacking leadership participation - C9.1*). Initially, 3.4-CS-4 made use of a common retrospective for which all teams could suggest issues to discuss, three of those issues were discussed in an attempt to resolve them (*Encourage project visibility - SF1.4*). However, the issues discussed were too big to be resolve in one iteration and were rarely followed up upon. This was not as useful, so 3.4-CS-4 moved to a ”open space” where several discussion could take place simultaneously (*Encourage project visibility - SF1.4*), these discussions did not lead to any actual change either. Lastly, a new format was attempted with an internal coach (*Train and coach people - SF2.1*) which was executed more structured and resulted in an actual follow-up on issues.

Paasivaara and Lassenius report four pain points with the (large scale) agile adoption of 3.4-CS-4, a partially missing agile mindset (*Difficulty adopting agile mindset - C4.2*), issues with dividing the solution in requirement areas (*Practice specific issues - C8*), a lacking common view of the scrum implementation (no *Define a common view on the change and terminology - SF8.1* resulting in *Inconsistent terminology used in frameworks with different ways to interpret - C6.1*), and a constant

market pressure causing time pressure (*Increased stress and pressure - C1.5*). Interestingly, in 3.4-CS-4 many attempts are made to encourage the project visibility (*Encourage project visibility - SF1.4*) but these seem to have adverse results (*Difficulty scaling agile to distributed organisations - C3.4*, *Lack of comparison between frameworks - C7.4*). However, as can be seen in the example of the common retrospective, 3.4-CS-4 was actively working on improving things when they turned out not to work or have the results as expected.

3.4.5 3.4-CS-5 & 3.4-CS-6

Paasivaara [42] compares two case studies within the same organisation adopting SAFe. 3.4-CS-5 and 3.4-CS-6 are two different business lines within the globally distributed software development (sub)organisation of a telecommunications company.
3.4-CS-5 started their (large scale) agile journey about six months before 3.4-CS-6 did. 3.4-CS-5 consisted of 14 teams spanning four locations, 3.4-CS-6 consisted of 11 teams spanning three locations and an additional team split over two of those locations. Teams in both cases consist of five to ten members (*Small team size - SF5.1*). No solutions other than SAFe were considered and no comparison was made (*Lack of comparison between frameworks - C7.4*), 3.4-CS-5 and 3.4-CS-6 chose SAFe independently. 3.4-CS-5 and 3.4-CS-6 have increments of ten weeks and iterations of two weeks (*Short iterations - SF4.2*). Paasivaara report a closer collaboration and communication between the development teams, product managers, and product owners in both 3.4-CS-5 as 3.4-CS-6 (*Close connections & constant communication between teams - SF1.2*). Regular SoS meetings are scheduled by teams experiencing dependencies (*Effective customer involvement - SF1.3*).

3.4-CS-5 did not provide training or coaching before the transition (*Lack of training - C7.1*) and only initiated training when problems were faced about half a year after the (large scale) agile adoption. 3.4-CS-6 trained those with management roles before the transition with help of a SAFe consultancy company (*Acquire knowledge on (large scale) agile - SF2.2*), team members were trained before the first increment planning, and got a refresher training a few months after the first increment planning (*Train and coach people - SF2.1*, *Acquire knowledge on (large scale) agile - SF2.2*). Due to the lack of training in 3.4-CS-5 they also ran into issues executing the first increment planning (*Issues with the first instance of a practice - C8.1*).

Paasivaara report a large amount of change resistance (*Resistance towards change - C5.1*) in 3.4-CS-5, neither the top of the organisation nor the bottom of the organisation was extensively involved in the decision (*Difficulty initiating from both the top and bottom of the organisation - C4.4*). There was no feeling of importance

from management (*Lacking leadership participation - C9.1*, *Lacking management support and sponsorship - C9.2*) and teams felt a strong resistance to change due to little communication about the transition (lacking *Communication, coordination, and control - SF1.1*, *Lack of information- C7*) and lack of understanding of SAFe (*Lack of information- C7*). 3.4-CS-6 had little issues with resistance to change, even though they expected it based on 3.4-CS-5, this is accredited to accurate training (*Train and coach people - SF2.1*) and the quick resolving of issues.

3.4-CS-6 had full-time dedicated change agents from the management side (*Executive sponsorship - SF6.2*), whereas 3.4-CS-5 had fewer part-time change agents and those were thus less able to lead the charge (*Lacking management support and sponsorship - C9.2*). 3.4-CS-5 was reported to have difficulty improving upon issues that arose with SAFe (*Framework issues - C6*, *Practice specific issues - C8*). In 3.4-CS-5 the overall work satisfaction decreased with the adoption of SAFe, due to negative experiences with SAFe (*Team sizes - C1.4*), lack of autonomy (*Teams lacking autonomy - C1.2*), and feeling like increments were a move back to waterfall (*Scaling agile feels like moving away from agile - C6.2*). On the other hand, 3.4-CS-6 reported an increase in employee satisfaction.

Even though 3.4-CS-5 and 3.4-CS-6 are adopting SAFe, the Paasivaara [42] paper does not have enough information to accurately fill in the Stojanov et al. [66] SAFe maturity model. Regardless, this model would be beneficial when looking into the adoption of SAFe in organisations.

3.4.6 Summarizing

The case studies discussed above shed a light on potential additional relations between the success factors and challenges. Especially the success factors related to *Communication - SF1* and *Acquire knowledge - SF2* seem to be able to avert challenges within our identified case studies.
Improved *Communication - SF1* can avert *Lack of information- C7* and prevent *Minimal collaboration and knowledge sharing between teams - C1.1*.
There is a relation between the *Experienced and motivated developers - SF5.3* and the effectiveness of their *Communication - SF1*.
Acquire knowledge - SF2 can help prevent *Practice specific issues - C8* and can be improved upon through strong *Communication - SF1*.
Encourage knowledge sharing - SF2.3 should help resolving *Struggles moving from fixed delivery to iterative delivery - C4.5* and improve *Lack of skills or experience with agile - C7.2*. However, as 3.4-CS-4 shows, a mismatched communication implementation, or a too ambitious push for *Encourage knowledge sharing - SF2.3*, could also have adverse affects.

Ensure strong engineering practices to achieve as little reduction of quality as possible - SF4.1 can be utilised to prevent a *Loss of quality - C10.2*.
Transitioning slowly and carefully - SF6.1 should aid in *Struggles moving from fixed delivery to iterative delivery - C4.5* and can result in *Short iterations - SF4.2*.
Physical distance between team members - C3.1 can result in issues with *Synchronising communications - C3.2* and *Having multiple communication channels - C3.3*.
Lack of skills or experience with agile - C7.2 can especially result in *Difficulty scaling agile to distributed organisations - C3.4*.
Lastly, there is a relation between *Lacking leadership participation - C9.1* and *Lacking management support and sponsorship - C9.2*.

The most commonly reported success factors are:

- *Train and coach people - SF2.1* (six mentions)
- *Encourage project visibility - SF1.4* (six mentions)
- *Close connections & constant communication between teams - SF1.2* (five mentions)
- *Small team size - SF5.1* (five mentions)

The most commonly reported challenges are:

- *Physical distance between team members - C3.1* (four mentions)
- *Difficulty scaling agile to distributed organisations - C3.4* (three mentions)
- *Lack of skills or experience with agile - C7.2* (three mentions).

3.5 DevOps and (large scale) agile

According to Dyck et al. [15] the definition of DevOps is as follows:
"DevOps is an organisational approach that stresses empathy and cross-functional collaboration within and between teams – especially development and IT operations – in software development organisations, in order to operate resilient systems and accelerate delivery of changes."

As can be seen in appendix A there is no scientific literature to be found on DevOps and the scaling of DevOps. However, some frameworks do feature elements of DevOps or discuss how DevOps can be utilised within their framework. Out of the seven large scale agile approaches as discussed in section 3.2; SAFe, DAD, and RAGE mention DevOps in their models. However, RAGE does only mention the word DevOps and has no additional publicly available information on it, so for the sake of this chapter RAGE will be ignored. With regards to DevOps in LeSS a blog was written by explaining how DevOps is automatically integrated in the principles of LeSS, but no information is presented directly by the organisation behind LeSS.

3.5.1 SAFe and DevOps

SAFe simply states that DevOps is implemented by SAFe organisation to empower their agile release trains. SAFe claims that over time it will reduce the separation between development and operations. Many SAFe concepts and principles either support DevOps or directly support the business need that precedes the adoption of DevOps.
SAFe makes use of their CALMR approach to cover five main aspects of DevOps.
A Culture of shared responsibility should ensure collaboration, risk tolerance, self-service, knowledge sharing, and automation to enable DevOps success.
Automation of the continuous delivery pipeline should increase the reliability of the continuous delivery pipeline processes.
A Lean flow accelerating delivery should visualise and limit work in progress to achieve a reduction in batch sizes and manage queue lengths.
Measuring everything should help to quickly assess the quality of the product and should provide extra insight in the effectiveness of the continuous delivery pipeline.
Recovery enabling low risk releases should enhance recovery time in case releases result in operational failure, this should provide more freedom to push releases by reducing the risk of operational failure.

Lastly, SAFe deploys a DevOps Health Radar to assess maturity of their continuous delivery pipeline which should be influenced by DevOps. [59]

3.5.2 DAD and DevOps

DAD present Disciplined DevOps to aid in streamlining development and operations [50]. Disciplined DevOps is defined as the following:
"Disciplined DevOps is the streamlining of IT solution development and IT operations activities, along with supporting enterprise-IT activities such as Security and Data Management, to provide more effective outcomes to an organisation"
This usually means development teams make use of continuous delivery cycles to reach the goals of DevOps [51].
The model of Disciplined DevOps is based on the models of BizDevOps (including business operations), DevSecOps (focus on security), DataDevOps (improved data management), and making release management and support explicit. Disciplined DevOps presents 59 difference potential strategies to aid DevOps split over nine different categories [52]. Three success factors are identified in adopting Disciplined DevOps [53]:

- A collaborative and respectful culture should make the adoption more successful
- The focus should be on people, but processes and tools should not be forgotten about

• Make good use of the various choices available

The full Disciplined DevOps workflow can be found as figure 5 on the website of the Project Management Institute [51].

3.5.3 LeSS and DevOps

According to Lynn [38] DevOps is built into the LeSS approach. Lynn states this is due to LeSS focusing directly on team collaboration which is supported by LeSS leveraging fast feedback loops, frequent check-ins, and automated tests. LeSS achieves this through continuous integration & development, automated testing, and cross functional teams [38].

DevOps is not explicitly mentioned in any of the general LeSS literature, but the arguments made for the inclusion of DevOps are similar to the arguments made within SAFe and DAD.

3.5.4 Comparison

Hering [22] writes about the DevOps approaches in SAFe, DAD, and LeSS. Hering mentions a split between DevOps teams and system teams in SAFe. According to Hering this could still lead to a distance between development and operations. Within DAD, Hering states, there is too much focus on the processes of DevOps and not enough focus on the technical parts, even though DevOps is weaved into the fabric of DAD. Lastly, DevOps in LeSS is focused on supporting practices and principles.

Hering [22] is one of the few sources available that looks a little more broadly at DevOps in any scaled form, yet the information available is still lacking.

There is still a large gap in available (scientific) literature or knowledge on scaling DevOps, or the combination of DevOps in (large scale) agile approaches. It would be beneficial to the scientific community to expand on the knowledge of large scale DevOps.

3.6 Comparing public and private sector IT projects

As can be seen in section 3.3 and 3.4 a majority of available research is performed on IT projects within companies. However it is uncertain whether or not this research is applicable to IT projects within public organisations. In this chapter I will investigate the difference and similarities between IT projects in the public sector and the private sector. This should help understand how the results of the previous chapters apply to target organisations in the public sector.

Gasik [17] mentions three different models used in the research of organisation to model the differences between public and private organisations. The *generic model*, the *core model*, and the *dimension model*.
The generic model states that there are no differences between public and private organisations.
According to the core model there are substantial differences between public and private organisations, Sayre [57] states that public and private organisations are similar in all unimportant aspects. Gasik states that this can, for example, be seen in procurement management and stakeholder management.
Following the dimension model the *publicness* of an organisation is based on three dimension. Within those dimensions an organisation can be classified as a value between fully private and fully public. These dimensions as defined by Perry and Rainey [48] are *ownership*, *funding*, and *mode of social control*.

Based on a 512 respondent survey (2016), Gasik [17] identified three different groups of management areas with different levels of complexity in the comparison between the public and private sector. According to Gasik, the most complex management areas in the public sector are stakeholder management, procurement management, and communication management. The group of medium complex areas compared to the private sector are HR, scope, integrity, cost, time, and risk management. The smallest relatively complex management area is quality management.
In general project management of public projects is deemed to be significantly more complex than private project management. Based on this Gasik concludes that the generic model does not seem to be the correct model when comparing public and private organisations.
Gasik discusses potential reasons for the three relatively most complex management areas.

Stakeholder management in public project management is increasingly more complex due to a higher amount of external factors resulting in more stakeholders. Public projects are exposed to a higher amount of criticism and are at risk of affecting the public image of the government, therefore they must, for example, take into account politicians who might not focus on the project itself, but more on image. Public sector projects require more interdependence with other public agencies which increases the amount of stakeholders. Gasik states that the need to convince employees in the public sector to change their processes is greater than in the private sector. This would be due to frequent management changes and added restrictions from regulations.

Many public projects are executed by external organisations so there is a lot of procurement work to be performed. However there is a big difference between procurement of public projects compared to private projects. The main procurement

factor for private projects is usually the cost, however for public projects many more factors are to be considered. According to Gasik this is due to different success factors of public sectors, public sector projects must take into account some selection criteria which might be increasingly more difficult to measure.
Gasik reports that the *red tape*, formal rules and regulations of a sector, most commonly complicates procurement management and personnel management. Research indicates that *red tape* has a greater impact on the functioning of managers in the public sector compared to the private sector. This additional bureaucracy results in a complication of the procurement process.

Communication management is a tool to support stakeholder management and the key way to influence external stakeholders of the project. A higher complexity of stakeholder management makes communication management more complex as well. A higher amount of *red tape* is frequently associated with a lower efficiency of communication. Combining both these factors increases the complexity of communication management even more.

According to the PMBOK® guide[49], knowledge of project management can be split in ten knowledge areas:

Integration management - integrative, unifying, and/or coordinating project activities

Scope management - ensuring a project encompasses all and only the necessary work

Time management - ensuring a timely completion of projects

Cost management - ensuring a project stays inside its own budget

Quality management - processes which ensure satisfaction of the needs behind a project

Human resources management - managing all personnel matters in a project

Communication management - ensuring all involved parties and stakeholders have the right information at the right time

Risk management - decreasing potential threats and increasing positive impact

Procurement management - processes for acquiring goods and services outside of the organisation

Stakeholder management - interaction, engagement, and assuring satisfaction of all involved stakeholders

In a later paper, Gasik [18] reviews research comparing the public and private sector on each of the ten knowledge areas as defined by the PMBOK® guide[49]. The most significantly different knowledge areas according to Gasik are integration, HR, procurement, and stakeholder management. Communication management has a median significance. Less significant are cost, risk, quality, scope, and time management. Based on these findings of more and less significant findings over multiple

project management knowledge areas, Gasik concludes the dimension model is the only realistic model to compare public and private sector projects.
These findings are somewhat similar to the 2016 Gasik [17] paper. After concluding the generic model is not realistic [17] this new paper also signifies the core model is not the most realistic portrayal of the differences between public and private projects [18]. Similarly, Gasik mentions procurement, stakeholder, and communication management as significantly different in both papers, in the 2018 paper Gasik [18] adds integration and HR management to this list.

Jurisch et al. [24] discuss the key differences between business process change (BPC) in the public and private sector based on 128 case studies. Jurisch et al. conclude that both the private sector and the public sector could learn from one another when executing BPC projects. Similarly to the previous two papers by Gasik [17, 18], Jurisch et al. finds public projects have to put more effort in stakeholder management and building support for change, this is due to inconsistent interests and aims of both internal and external stakeholders. Public sector BPC projects require more resource forecasting and planning, due to stringent (annual) budgets. Lastly, the Jurisch et al. results show that public BPC projects invest more efforts in risk management, potentially this is due to frequent changes of governmental appointments (the frequent management changes as mentioned before[17]) and changing political agendas.

In 2010, Rosacker and Rosacker [56] present their findings on IT project management within the public sector. Rosacker and Rosacker state public and private sector organisations are similar in many important ways, but also provide some insight in the differences. An interesting point raised is the pressure on public sector organisations compared to private sector organisations. The public sector rarely has to deal with competitors and have no other organisations they need to stay ahead off because usually they are the only organisations providing their specific service. This results in public sector organisation not having the pressure to innovate in the short term to survive competition in the long term. However, the public sector does have to deal with political pressure, this political pressure is usually aimed at short term innovation with a more disruptive influence on long term innovation.
Rosacker and Rosacker discuss a larger accountability on public sector project managers, significantly restricting a public sector organisation from the ability to control or manage contemporary IT projects. Public sector projects not only get pressure from users and customers, but also from elected officials, media, special interest groups, and their own governmental management structure. This additional scrutiny is added on top of the inability of public sector organisations to keep strategical decisions a secret and thus be subject to scrutiny constantly instead of when they themselves decide to publish decisions.

Cats-Baril and Thompson [11] performed a case study on a 1995 American public sector IT project. The conclusions by Cats-Baril and Thompson are similar to the findings discussed before, indicating that these are not recent problems but issues that have been around for at least 25 years.
For example, Cats-Baril and Thompson find that a greater interdependence across organisational boundaries increases the need for clarity in public sector projects. Cats-Baril and Thompson mention the frequent turnover of upper management and the high amounts of red tape result in an increased difficulty to implement change and a higher need to convince employees of the need for change. Cats-Baril and Thompson add that criteria to justify technological innovations are more stringent and that relevant project managers do not have as much authority as their private sector counterparts.

The main differences between public sector and private sector projects seem to be in the management areas of stakeholder management, procurement management, and communication management to support the stakeholder management. More effort needs to be invested to convince employees and partners of a need to change within public sector projects. Due to a high volatility in public sector upper management, public sector projects are under an entirely different kind of scrutiny compared to their private sector counterparts. Additionally public sector projects have to deal with a considerably larger amount of red tape which slows down processes and complicates project management.

3.7 Common (large scale) agile research approaches

To answer this question I will perform a review on existing scientific case studies to determine the approach used. The papers used in this chapter can be found in table 3.4. From these sixteen papers the research methodology and approach, validation, review method of agile, and organisational focus level are extracted.

Fourteen out of the 16 papers (87.5%) make use of a form of case study research. The other two approaches are qualitative research (3.7-P-1) and simulation (3.7-P-4). Two papers (3.7-P-2 & 3.7-P-11) make use of a longitudinal case study, a case study executed over a longer period of time (seven years and three years respectively). Six papers (3.7-P-3, 5, 7, 8, 13, & 3.7-P-14) indicated the usage of a multiple case study approach, some of these are different projects in the same organisation, some papers look at multiple organisations. The remaining six papers (3.7-P-6, 9, 10, 12, 15, & 3.7-P-16) indicate the usage of a single case study approach. There is a different in the way these case studies are classified, some authors classify multiple projects within the same organisation as a singly case study, some authors classify multiple projects within the same organisation as a multiple

ID	Cite	Year	Goal of the paper
3.7-P-1	[9]	2011	Identify benefits and side-effects
3.7-P-2	[70]	2011	How is (large scale) agile adopted
3.7-P-3	[45]	2012	Investigate effectiveness of approach
3.7-P-4	[5]	2012	Simulate different approaches
3.7-P-5	[29]	2013	Compare waterfall to (large scale) agile
3.7-P-6	[20]	2013	How is (large scale) agile adopted
3.7-P-7	[6]	2013	Identify challenges of productivity or delay
3.7-P-8	[13]	2013	Team productivity in (large scale) agile
3.7-P-9	[16]	2013	How is (large scale) agile adopted
3.7-P-10	[46]	2014	How is (large scale) agile adopted
3.7-P-11	[21]	2015	How to apply the release iteration planning method
3.7-P-12	[66]	2015	Apply a maturity model
3.7-P-13	[43]	2016	How is (large scale) agile adopted
3.7-P-14	[26]	2017	Challenges with requirement engineering
3.7-P-15	[19]	2017	Challenges with requirement engineering and system testing
3.7-P-16	[42]	2017	How is (large scale) agile adopted

Table 3.4: Case studies in chapter 3.7

case study. Regardless, in general it can be concluded that a case study approach is the most commonly used research methodology. Naturally multiple case studies would yield better results as it removes a single case study bias. Secondly, the usage of a longitudinal case study can yield better results as it prevents the case study from becoming a snapshot of a situation instead of the entire project.
Even though 3.7-P-1 states the usage of a qualitative research they still make use of a case study, however this case study is only part of the bigger picture of their research. Within the simulation research (3.7-P-4) the case study is used as input for the simulation.

A vast majority of the papers identified make use of semi-structured interviews as can be seen in table 3.5. The total amount of interviews conducted various greatly, 3.7-P-1 interviews a total of nine persons, whereas 3.7-P-3 interviews a total of 58 persons. The interviews are conducted with a variety of different roles featured, for example 3.7-P-14 interviewed 15 different roles. The most common roles interviewed are managers, product owners, scrum masters, developers, and testers. Almost all case studies are executed within one company, except for 3.7-P-8, 3.7-P-14, and 3.7-P-15. Some case organisations work distributed over multiple locations, more than half (five out of eight) case studies feature personnel from multiple locations. None of the case studies feature personnel from all locations a

case organisation is active in.
A multiple case study can be performed if an organisation has multiple independent business lines in the same organisation by evaluating the individual transitions of these teams. It is important the product created in both projects have no direct influence on each other as otherwise these should be seen as one case.

Most other methods shown in table 3.5 are used in combination with semi-structured interviews, except for 3.7-P-4, 3.7-P-5, and 3.7-P-12. 3.7-P-4 makes use of a simulation, to compare two agile approaches, a simulation is made of a case organisation and the rules of each approach are applied to this simulation. 3.7-P-5 makes use of a survey, due to the size of this survey (86 answers out of 240 invitees) interviews are not utilised. 3.7-P-12 performs a assessment of their case organisation as a first test of their maturity model, this assessment is not done directly through (semi-structured) interviews, but members of the case organisation are directly involved to perform the assessment.

Method	Frequency of use
Semi-structured interviews	13
Observations	3
Focus groups	2
Surveys	2
Workshops	2
Assessment	1
Simulations	1

Table 3.5: The approaches used to research applicability of (large scale) agile in large scale IT-projects

Seven of the 16 papers explicitly mention their approach to ensuring validity, other papers do mention validity but these papers ensured validity through frequently checking their own process. Five cases (3.7-P-3, 5, 10, 13, & 3.7-P-14) validate their findings through an open feedback session for which all interviewees or survey respondents are invited, in some cases any other interested personnel is also invited. Two papers (3.7-P-7 & 3.7-P-9) ensure validity through a prolonged involvement of the researcher(s) in the project, neither of these papers are longitudinal case studies thus the prolonged involvement is an actual improvement upon their case study approach.

The 16 papers each use a different method to assess whether (the transition to) (large scale) agile is suitably supportive to the organisation. The most common method (69%) relies on identifying challenges and/or success factors. 3.7-P-8 assesses the transition to (large scale) agile based on their own conceptual framework which is created in the same paper, similarly 3.7-P-12 uses the case study to test

their own maturity model created in the same paper. 3.7-P-5 assesses the usage of (large scale) agile by comparing the (large scale) agile case to a similar case executed in a traditional way and compares the two based on various principles of agile. 3.7-P-3 bases their (large scale) agile assessment on the opinions of their interviewees, whether or not the interviewee deemed the transition to (large scale) agile to be successful.
3.7-P-4 is the only paper which makes use of a numerically measurable method, namely the amount of user stories completed, this is due to 3.7-P-4 being based on a simulation and it is simply not possible to generate more abstract results, like challenges and success factors, from a simulation.

Fifteen out of the 16 papers (94%) look at the effect of (large scale) agile on the organisational level, 3.7-P-8 is the only paper which looks at the effect of (large scale) agile on a team level, no papers look at the individual level. None of the papers mentioned look at multiple levels. To be able to get a clear picture of the different organisational levels all papers focus on the individual level, except for 3.7-P-4 which judges the results based on a team level.
Based on this, there seems to be a good opportunity to add to existing knowledge by assessing the effects of (large scale) agile on the individual or team level.

Chapter 4

Case study

4.1 Goal

The goal of this multiple case study research is to acquire information and knowledge of the current practical applications of (large scale) agile within multiple Dutch public sector organisations. This knowledge should help with the problem identification activity of the DSRM, but mostly with the second stage of the DSRM when defining the objectives of the artefact.

The required knowledge for these activities is first of all simply the current (large scale) agile methods used and the reasoning behind these methods and ways of working. The reasoning and considerations behind certain methods should give an insight in important factors for (large scale) agile adoption in these Dutch public sector organisations. Secondly, the perceived current and historical challenges and strengths of (large scale) agile as utilised should give an insight in the potential risks of (large scale) agile in the Dutch public sector. A final focus in these case studies will be on the perceived cultural change required in the Dutch government to be able to adopt (large scale) agile [10].

4.2 Methodology

Participants of the case studies will be asked to fill in a survey and take part in an semi-structured interview. Case studies are collected through the network of the researcher and the network of the supporting Rijkswaterstaat (large scale) agile working group. The contacts gathered through the network are utilised in a snowball fashion to get in contact with relevant organisational managers of the different case study (sub)organisations.

The target time of the interviews is approximately one hour, the target time of the survey is approximately ten minutes.

All data acquired through the case study will be processed anonymously, neither the participants nor the cases will be traceable for outside parties. Participants and cases might be traceable for colleagues and parties directly involved with these cases. When requested (partial) results of participants will be completely anonymised and made untraceable for colleagues and parties involved. All collected data and recorded interviews are deleted after publication of this paper.
Participants are able to withdraw from the research at any time without given reason.

The results of the case studies will be twofold. First of all the survey should provide quantitative, but limited, results. The importance of the surveys is the ability to draw measurable conclusions from the case studies and perform a broader analysis and comparisons between cases or roles within cases. These measurable results could also be used when executing the evaluation of the DSRM artefact, comparing the results of the survey before and after demonstration or usage of the artefact.

The interviews should provide the more qualitative results of the case studies. First of all, the interviews should give more insight in reasoning behind the survey results and allow for explanation of certain beliefs. Secondly, the interviews should allow for more in-depth insights in the usage of (large scale) agile within public sector organisations. The interviews will be recorded to thoroughly analyse the interviews and score remarks and answer given by participants. Frequency of (similar) statements will be analysed to determine the relevance and importance of issues raised during the interviews.

4.2.1 Interview analysis

All interviews were recorded after receiving permission from the participant. During the interview manual notes were created as well. Qualitative data analysis software ATLAS.ti was used to analyse the recordings. Relevant sections of the recordings were analysed following the concept of *in vivo coding* [39]. In vivo coding was chosen as a first step of the interview analysis to ensure all relevant sections of the interviews would be taken in account and there would be no bias based on the existing codes.
Fragments of the recordings were selected and turned into quotations to be used for the next steps of coding.

An initial list of codes was created based on the success factors and challenges as defined in section 3.3. These codes are applied to the various relevant quotations (*deductive coding* [40]). Through deductive coding not all quotations could be coded, therefore new codes were introduced which could encompass these quotations (*inductive coding* [40]). Once all quotations were coded the total list of codes were combined and redundant or overlapping codes were removed. As a last step

the coded quotations were evaluated based on the full codebook to ensure all quotations were coded correctly and completely.

The codes based on the success factors and challenges were split over those respective categories.
The newly created codes are split over three categories; positives, negatives, and notes. Naturally, the positive and negative codes reflect positive and negative opinions on either (large scale) agile or the current agile situation of the interviewees. All codes categorised under notes are observations or general opinions voiced by the interviewees.

The final codebook can be found in Appendix E.

4.3 Design

4.3.1 Survey

The survey consists of seventeen statements which can be graded on a 5-point likert-scale [37] from strongly disagree to strongly agree. All statements are formulated positively so a high score converts to a positive attitude towards a statement. Statements in the survey are presented in a randomised order. To ensure the survey does not take too long and participants are willing to fill in the full survey it was chosen to make use of seventeen questions, this allowed for sufficient coverage of measurable topics without encroaching too much on the set target time.

In the introduction section of the survey all participants will fill in their personal identification code communicated by the researcher beforehand. On this page the participants will be informed about their rights as a participants, this includes the anonymisation of data and the right to withdraw from the research.

The statements are formulated based on the (categories of) challenges and success factors as defined in section 3.3 and on the goals of the case study as defined in section 4.1. Not all (categories of) challenges and success factors can be translated in easily grade-able statements, these will be taken into account during the interviews.
A list of statements, translated to English, can be found in Appendix D.

4.3.2 Interviews

For the interviews, a list of key direction subjects is created. Similarly to the survey, this list is based on the literature study in section 3 and the goals of the case studies. Besides these questions, the most noteworthy results of a participants survey are discussed. The full survey results are not discussed.

Before the start of the interviews the participants are informed of the formalities surrounding their participation. All participants are asked for their permission to record the interview. Participants are informed about the way collected data is anonymised and their right to withdraw from the research at all times.

The subjects of the interviews are defined as the following:

- Current role within the case organisation and the work this involves
- Opinion about the current way of working
  - Biggest challenges with current way of working
  - Biggest strengths with current way of working
  - What would the participant change
- Opinion about the Dutch public sector government (culture) and (large scale) agile
- Survey results

The predetermined subjects in the interview is a non-exhaustive list, based on the subjects discussed with the participants additional questions or subjects will be explored.

The survey and interviews will be performed in Dutch as this is the primary language of the researcher and the participants.

4.3.3 Data compliance

The survey is executed through Google Forms, data collected through Google Forms is anonymised based on a personal identification code. The data collected through the Google Forms is not stored on Dutch public sector government servers and therefore requires anonymisation through the personal identification code. The data is processed and analysed through a Excel file stored on Dutch public sector government servers to ensure data compliance.
The interviews are performed digitally through Microsoft Teams and recorded through Microsoft Teams. Similarly to the survey results, these recordings are stored on Dutch public sector government servers to ensure data compliance.
All data will be deleted once it is processed and implemented in this paper.

4.4 Cases

This multiple case study research consists of four case organisations. All case organisations are sub-organisations, projects, or programs within Dutch government agencies or ministries. In this section, a short description of the case organisations and the different actors included in the research will be given. A short overview of the four cases can be found in table 4.1.

Case	# Employees	# Teams	Approach
Case 1	100	13	S@S
Case 2	20	3	S@S
Case 3	200	18	SAFe
Case 4	150	14	S@S & SAFe

Table 4.1: Overview of the four cases

The participants of the different case studies are split in five generalised role categories. Generalised role categories are used to be more easily able to compare roles in different cases. Roles are only grouped together in a category if their responsibilities or background are somewhat similar. The categories are listed below, added are examples of roles included in those categories. Product owners are not grouped in a category due to their unique (large scale) agile nature.

- **Agile process coach** (scrum master, scrum of scrums master, agile coach)
- **Developer** (developer, tester, analyst)
- **Manager** (program manager, product manager, architect)
- **Product owner** (product owner)
- **Stakeholder** (stakeholder, client, user)

A short overview of the division of these roles over the various cases can be found in table 4.2.

Case	APC	DEV	MGR	PO	SH
Case 1	2	0	0	1	0
Case 2	1	2	2	1	2
Case 3	1	4	2	1	1
Case 4	1	0	0	1	0

Table 4.2: Overview of the roles interviewed within each case

4.4.1 Case 1

The first case is a shared project of a ministry, and an executive agency. The project is executed with the use of S@S which was introduced by the executive agency. The project team consists of approximately 100 team members working in teams of 7-8 people. Case 1 works with three week sprints, however external testing takes six weeks so releases are only possible every six weeks. This case has a long list of stakeholders spread over multiple organisations which means a lot of focus has to be put on stakeholder management.

The epics of the product are set out in a program of requirements to provide certainty to the business side of the project. The budget and time of the project are

restricted. Regardless of the restrictions on scope, budget, and time, Case 1 works according to agile principle as the specification of the epics is still flexible.
For the first case two agile process coaches and one product owner were interviewed.

4.4.2 Case 2

The second case study consists of three teams working according to S@S. Two teams work on the main product with a third team focusing on the integration of external applications and systems with the main product. Two PO's support the teams, one with a focus on the main product teams and one with a focus on the integration team. As an addition to S@S, *Story Owners* are introduced in the main product teams to ensure story refinement. Testing is performed in-house with sprints and releases every three weeks. Users of the product are represented in expert teams as the user base of the product is widely spread out.
For Case 2 one agile process coach, two developers, two managers, one product owner, and two stakeholders were interviewed.

4.4.3 Case 3

The third case is a project working according to the SAFe principles. The project is split in 13 different subprograms each with their own team(s). Interviews were held within one subprogram which was focused on development of new IT solutions. A part of the work was executed by a external party acquired through a tender process. Case 3 has program increments of three months, which are split in sprints of one month which some teams split even further in two week sprints. At the time of interviews the third increment had just started. Before adopting SAFe, due to the increased size of the project, Case 3 worked according to continuous development principles. Besides a daily standup within each team, daily SoS meetings are held between the teams for a continuous synchronisation between the teams.
Within Case 3 one agile process coach, four developers, two managers, one product owner, and one stakeholder were interviewed.
The stakeholder interviewed did not fill in the survey as they were a former stakeholder and has not been involved with the project with the current way of working. However they were added as an interviewee as they were currently being trained as a SAFe practitioner.

4.4.4 Case 4

The final case is a supporting project organisation within a government agency. The project organisation bears responsibility for the infrastructural IT necessities of the government agency. Case 4 has developed their own (large scale) agile way of working following principles of both S@S and SAFe, with in total 14 teams. Case 4 has increments of 3 months with two weekly sprints and two weekly PO synchronisation. The teams in Case 4 work on multiple different projects and products. New teams, projects, or products are added when deemed necessary or in case they indicate the wish to be included in Case 4. Newly introduced teams initially do not actively partake in the SAFe proceedings but observe the proceedings to get acquainted with it.
Within Case 4 one agile process coach and one product owner were interviewed.

4.5 Survey results

Out of the 22 total interviewees 21 participants filled in the survey. A stakeholder of case 3 was no longer involved in the project since the introduction of a new way of working and therefore has not filled in the survey. Relevant subjects of the survey have still been discussed in the interview with the case 3 stakeholder.

In general the survey results are leaning towards the positive side. The average score given by all participants, excluding empty results, is a 3.83, with a median of 4. Only on five occasion a score of 1 was given to a question.
The seventeenth question, "*I think my organisation made a well-informed decision for the current structure of the organisation.*", has not been answered three times, for all three times the participant indicated this was due to them simply not being aware if this would be the case. In table 4.3 the average of the total results of the survey questions can be found. The seventh, eighth, and thirteenth question, respectively "*The current way of working in my organisation is good*", "*The support for large scale agile working from the different management layers is good*", and "*The communication and knowledge sharing with parties I do not work with on a daily basis is good*", were the lowest scoring questions.
The highest scoring question is the sixth question, "*I am motivated in my work*", followed by question twelve, "*I think (large scale) agile is fitting to the work I do*".
Based on the links between the survey questions and the challenges and success factors, as discussed in section 3.4, a few of the main challenges and success factors within the Dutch public sector can be determined. Based on a low score in question eight *Lacking leadership participation - C9.1* and *Lacking management support and sponsorship - C9.2* come to light as potential areas of improvement.

Q1: The communication and knowledge sharing with my direct colleagues and supervisor is good.	4.14
Q2: I have enough knowledge of (large scale) agile.	3.86
Q3: The technical environment (tools and infrastructure) which support my work are good.	3.81
Q4: There is sufficient attention to quality assurance of the final product I am working on.	3.86
Q5: I have enough autonomy in the work I do.	4.38
Q6: I am motivated in my work.	4.57
Q7: The current way of working in my organisation is good.	3.00
Q8: The support for large scale agile working from the different management layers is good.	3.00
Q9: My colleagues understand me when I talk about the work I do.	4.15
Q10: My stress levels are low.	3.67
Q11: I think (large scale) agile is fitting to my organisation.	3.81
Q12: I think (large scale) agile is fitting to the work I do.	4.48
Q13: The communication and knowledge sharing with parties I do not work with on a daily basis is good.	3.10
Q14: I feel involved in the current way of working of my organisation.	4.05
Q15: I do my working according to the agile principles.	3.70
Q16: I am aware of the work my direct colleagues are doing.	4.19
Q17: I think my organisation made a well-informed decision for the current structure of the organisation.	3.33

Table 4.3: Average survey results per question

Secondly, the low score in question 13 points towards issues with *Minimal collaboration and knowledge sharing between teams - C1.1* and *Synchronising communications - C3.2* or *Having multiple communication channels - C3.3*. On the other hand a higher score in question six points, partially, towards *Experienced and motivated developers - SF5.3*.

Interestingly, the combination of a low scoring question seven and a high scoring question twelve indicates that the current way of working could be improved and that employees think (large scale) agile could play a role in this improvement. This is in line with the conclusion of Bolhuis [10].

A correlation matrix was created to analyse the correlation between the individual survey questions. This correlation matrix can be found in appendix F.2. No significant correlation has been found between any of the questions. In appendix F.3 a full list of survey averages can be found. Appendix F.3 is split up in three different

sections, initially the averages per question (as presented in table 4.3). Besides the general averages of all the questions, the averages per case per question and per role per question are presented as well. These extra lists of averages are correlated with the general average.
No significant correlation has been found between the different roles or cases.

4.6 Interview results

In this section I will attempt to provide clarity on the current ways of working at the Dutch public sector, and on the vision of the various Dutch public sector interviewees on (large scale) agile working. This will be achieved through answering the following questions:

1. Which challenges and success factors can be found within the cases?
2. What are the biggest challenges and success factors of (large scale) agile within Dutch public sector?
3. Can (large scale) agile fit to the Dutch public sector?
4. What core values of (large scale) agile are essential to (large scale) agile at the Dutch public sector?
5. How can a successful adoption of (large scale) agile be achieved at the Dutch public sector?

4.6.1 Challenges and success factors within the cases

In section 3.3 the challenges and success factors of (large scale) agile were defined. Within this case study six success factors:

- *Close connections & constant communication between teams - SF1.2*
- *Effective customer involvement - SF1.3*
- *Train and coach people - SF2.1*
- *Encourage knowledge sharing - SF2.3*
- *Short iterations - SF4.2*
- *Small team size - SF5.1*

and four challenges:

- *Teams lacking autonomy - C1.2*
- *Lack of trust - C1.3*
- *Resistance towards change - C5.1*
- *Lacking leadership participation - C9.1*

were found in all four cases.
Nine challenges:

- *Team sizes - C1.4*
- *Large scale agile approach not applicable due to regulations - C2.4*
- *Having multiple communication channels - C3.3*
- *Resistance towards new roles - C5.2*
- *Afraid of consequences of increased transparency - C5.3*
- *Little available literature resulting in unknown or unsolved issues - C7.3*
- *Issues with the first instance of a practice - C8.1*
- *Issues with integration - C8.2*
- *Loss of quality - C10.2*

and one success factor:

- *Management commitment - SF7.1*

did not occur in any of the cases.

Most commonly, the remaining challenges and success factors occurred in two cases. In case 2 and case 3 most success factors and challenges were identified, however this could be due to the larger amount of interviewees in case 2 and case 3 compared to case 1 and case 4.
In Appendix G a table of the occurrence of success factors and challenges in each case can be found.
The frequency of occurrence of challenges and success factors is similar to those found in chapter 3.3.

Six of the additional codes are also mentioned in all four cases, these are:

- *NEG: Distance to user*
- *NOTE: Change always occur in a project*
- *NOTE: Management give direction, leave decisions & responsibility lower in organisation*
- *POS: Agile allows constant steering*
- *POS: Agile fits government*
- *POS: Agile works well*.

4.6.2 Challenges and success factors within Dutch public sector

The most commonly reported challenge and success factor of (large scale) agile within Dutch public sector is *Controversies between organisational and agile culture - C2.2*, it was reported 38 times in 14 out of 22 interviews and reported by interviewees in all different roles.
However, C2.2 was only reported in about 60% of the interviews. C2.2 was on average reported 2.7 times per interview. One interviewee reported C2.2 at seven different times during their interview. More than 50% of all codes were mentioned in

three or less interviews. The six codes mentioned in more than half of the interviews are the following:

- SF4.2 - Short iterations (18 interviews)
- POS - Agile allows constant steering (15 interviews)
- C2.2 - Mismatch organisation & agile (14 interviews)
- SF1.3 - Customer involvement (13 interviews)
- NEG - Distance to user (12 interviews)
- POS - Agile works well (12 interviews)

Similarly to the frequency of reported codes per interview, over 50% of all codes was reported a maximum of four times. The most commonly reported codes are the following:

- C2.2 - Mismatch organisation & agile (38)
- POS - Agile allows constant steering (29)
- SF1.3 - Customer involvement (25)
- SF4.2 - Short iterations (22)
- POS - Agile works well (17)
- SF1.2 - Close & constant communication (15)
- C1.3 - Lack of trust (15)

Combining the two lists above results in a list of eight codes, which seem to be the most occurring codes within the interviews, these can be found in table 4.4. All eight codes are mentioned by interviewees in all different roles. All codes, except for C2.2, occur in all four cases.

Code	**#I**	**#M**
SF4.2 - Short iterations	18	22
POS - Agile allows constant steering	15	29
C2.2 - Mismatch organisation & agile	14	38
SF1.3 - Customer involvement	13	25
NEG - Distance to user	12	14
POS - Agile works well	12	12
SF1.2 - Close & constant communication	10	15
C1.3 - Lack of trust	7	15

Table 4.4: Most commonly occurring codes in the case study research. #I is the amount of interviews the code was mentioned in, #M is the amount of mentions

Interestingly both *Effective customer involvement - SF1.3* and *NEG: Distance to user* are commonly occurring codes. At first sight it seems these two codes should

be mutually exclusive as you can not have both close customer involvement and a distance to your customer.
However, interviewees indicated that management and product owners were well able to involve the customer in the general project processes, but it was indicated that developers were less able to involve customers and users in the development processes. This mostly resulted in customers being involved with the epics of a project, but not able to be involved for individual stories.

4.6.3 Fitting (large scale) agile to Dutch public sector

Overall the interviewees seem to have a positive attitude towards their own use of (large scale) agile. However, many interviewees indicate that it might work well for their situation, but outside of their own situation they see *Controversies between organisational and agile culture - C2.2* (38 mentions in 14 interviews). Many interviewees indicate that the business and management side of the Dutch public sector wants certainty and strong deadlines within projects which would not be fitting to (large scale) agile.

A common co-occurrence with *Controversies between organisational and agile culture - C2.2* were the codes *NEG: Focus on money not business value* and *POS: Agile allows constant steering*.
The first code goes more in depth with reasoning why (large scale) agile would not fit Dutch public sector, which is a focus on the financial frames surrounding the project. Interviewees state that new features would be pushed by stakeholders not based on which features were most vital, but which stakeholder was willing to invest most to have their features added. Interviewees from all different roles indicate it would be better to have a central budget for the developing team so the focus can shift towards value driven development.

On the other hand, the second common co-occurrence with *Controversies between organisational and agile culture - C2.2* focuses on a commonly perceived strength of (large scale) agile : it's flexibility. Due to the short iterations (also a highly occurring code) it is possible to frequently readjust the direction of a project to better suit the needs and the goals of the project or the end product.
Many interviewees mention this in combination with the codes *NOTE: Change always occurs in a project* and *NOTE: Waterfall does not allow steering during a project*; indicating that before the start of a project you can never predict what happens over the coming years and thus need a form of flexibility which traditional project management methods can not offer as well as (large scale) agile can.

Some interviewees (six spread over all four cases) do indicate (large scale) agile would be fitting to Dutch public sector. However, to achieve a successful adoption

of (large scale) agile within Dutch public sector, the current challenges surrounding (large scale) agile within Dutch public sector need to be resolved.
As one interviewee put it:
"*Are they able to adopt (large scale) agile, probably.*
Do they want to adopt (large scale) agile, probably not (yet).
Should they adopt (large scale) agile, absolutely."
In section 4.6.4 and 4.6.5 I will discuss what is needed to achieve this successful adoption according to the interviewees.

4.6.4 Core values of (large scale) agile

To get a clear picture of the way (large scale) agile is, and should be, used within Dutch public sector, this section will discuss the most important values and features of (large scale) agile found within the cases or raised by interviewees.

Short iterations - SF4.2 and *Small team size - SF5.1* are the two main measurable success factors defined in chapter 3.3. According to the Scrum Guide [61] iterations should not be longer than one month and teams should not exceed ten members. Within all four cases iterations of one month or shorter were used and all teams consisted of less than ten members. Multiple interviewees mentioned the importance of both short iterations and small team sizes and many interviewees saw this as standard practice without which (large scale) agile would simply not be possible.

As discussed in section 4.6.3 there seemed to be *Controversies between organisational and agile culture - C2.2*. Interviewees indicated this was due to an untrained or unwilling business side of the organisation and a big distance between the developers and users or stakeholders. Due to the high amount of layers between the developers and end-users, or stakeholders, communication between the two layers is hindered. *Effective customer involvement - SF1.3* is frequently mentioned by interviewees as an important part of (large scale) agile. Especially being able to frequently gather feedback from users is deemed important to allow the development team to steer production when necessary.

An interesting finding in case 1 was the fact that even though sprints were three weeks long, releases only happened every six weeks. This was due to their six week testing process. Interviewees of case 1 indicated that agile was embedded in a larger Prince2 process, a strict program of requirements in which specification can be executed by the case organisation. However, to get quicker feedback interviewees in case 1 indicate that automated testing might have been beneficial. Interviewees from case 2 and 3 indicate that automated testing is an important part of (large scale) agile.

A second point raised to resolve *Controversies between organisational and agile culture - C2.2* is an adjustment of management approach. Various interviewees indicate it would be better if management would focus on determining the broad directions or the epics of a project and leave it up to the specialists (developer, etc) lower in the organisation to determine how this direction can be achieved and develop the more precise stories of a project. This goes hand in hand with indications of *Teams lacking autonomy - C1.2* and *Lack of trust - C1.3*.

Within case 3 and case 4 a program increment (PI) planning event is held quarterly. During the PI planning event plans are made for the coming four sprints and allows teams to beforehand deal with potential impediments and dependencies. Interviewees from both cases indicate that the addition of the PI planning event has greatly improved collaboration between teams. It was stated by multiple interviewees that simply supplying a moment for the teams to come together and communicate adds a lot of value to a project organisation (*Communication, coordination, and control - SF1.1*). However, some developers stated that they didn't feel like the PI planning event was the right place for them as they felt it was too high-over or too much management based. So even though a program increment planning event adds a lot of value it is important to, where possible, keep the subjects discussed relevant to all attendees.

Lastly, interviewees (mostly from the side of managers and stakeholders) indicate that, regardless of working according to (large scale) agile principles, deadlines and budgets are inescapable. Due to the size of the organisations and projects it is important to have some form of commitment and control over the directions of the project. As found in chapter 3.6 public organisation projects are under an entirely different level of scrutiny compared to private sector organisations. Interviewees mention that (large scale) agile teams should not simply be selling sprints but should be able to commit to targets and a budget. When working (large scale) agile it is possible to commit, but all those involved in the project need to keep in the back of their mind that a 100% commitment is not always realistic.

To summarise, the following eight values and features of (large scale) agile are deemed most vital to working with (large scale) agile in the Dutch public sector according to the interviewees:

- *Short iterations - SF4.2*
- *Small team size - SF5.1*
- *Effective customer involvement - SF1.3*
- *Automated testing (prevent Difficulty automating testing - C8.3*)
- *Management should give direction (prevent Lack of trust - C1.3*)
- *Leave responsibility & execution to the specialists (prevent Teams lacking autonomy - C1.2* & *Lack of trust - C1.3*)

- *Facilitate Communication, coordination, and control - SF1.1*
- *Commit to targets and a budget*

Besides these values and features a few clear benefits of (large scale) agile have been discussed by the various interviewees. Most interviewees mention the increased transparency, visibility, and predictability (*Encourage project visibility - SF1.4*), the focus on quality instead of money or time, a decrease in stress, and increased flexibility.

4.6.5 Achieving a successful adoption

Many interviewees indicate (large scale) agile works well and is a good fit to their own (project)organisation. Especially when looking at the survey results in section 4.5 it is seen that the participants think improvement of the current way of working is necessary. In this section I will look at the way (large scale) agile could be more widely adopted within Dutch public sector.

As found in the Bolhuis [10] paper a cultural change would be required within Dutch public sector to be able to start a transition towards (large scale) agile. Based on the interviews it seems the IT executive parts of Dutch public sector organisations are ready and able to adopt (large scale) agile, as can be seen in the cases discussed here. According to the interviewees, the most major cultural change will be needed in the management and business layers currently surrounding the IT executive teams. A better match needs to be found in the business processes used to provide direction to the IT executive teams. This goes hand in hand with *Determining readiness of the organisation to transition - C4.3* and *Resistance towards change - C5.1*. This required cultural change can also be seen in the fourth point in section 4.6.4 which would require a new kind of mentality from management.

To achieve this required cultural change within Dutch public sector training and coaching of relevant personnel is required. Naturally POs or Scrum Masters need to be trained for their new role, but others involved within a project or supportive personnel of projects also need to be well informed and trained on what (large scale) agile would entice and how it would be used within their organisation and role (*Train and coach people - SF2.1*).

It was very noticeable in the different interviews and conversation held within the case organisation that interviewees and employees made use of different terminology when talking about the same thing or attached a different definition to the same terms (*Inconsistent terminology used in frameworks with different ways to interpret - C6.1*).

SAFe offers an interesting implementation roadmap [35] in which through multiple steps the relevant stakeholders and players within the transition get trained. The

SAFe implementation roadmap starts with training those responsible for the transition, followed by management. This would form the basis to determine the direction of (large scale) agile necessary and provide vital sponsorship of a (large scale) agile transition.
Once the direction of (large scale) agile transition is determined training of the product owners, scrum masters, and architects can start. When those trainings are complete the individual teams can be trained, when expanding the amount of teams the last two steps can simply be executed again to train the newly added teams. Once it becomes necessary to start including additional roles, for example chief product owner, these can be trained as soon as this expansion is introduced.

Another change suggested by the interviewees is the way features are prioritised. Currently the different stakeholders invest in the project and in turn get to submit feature requests. However, prioritisation of these features is not always done based on business value, multiple interviewees indicate that the features of the highest paying stakeholders would be prioritised over stakeholders with a lower budget. Dutch public sector should investigate whether and how it would be possible to let projects control their own budgets and thus make prioritisation of features independent of budget.
A suggestion raised by an interviewee was letting stakeholders award points to all features, but awarding points to features from a different stakeholder would yield more points, this way features which have a value for multiple stakeholders are prioritised.
Another suggestion raised was making use of the weighted shorted job first (WSJF) prioritisation, which is based on the cost of delay of a feature and the time it would take for the feature to be developed [55].
As a way to guide stakeholders in the (large scale) agile process and within prioritisation, interviewees suggest ensuring there is a shared goal between the stakeholders. By having all relevant parties have the same dot on the horizon in their mind it will become easier to find a common direction and having everyone on one line will most likely make communication more efficient.

To be able to broadly apply (large scale) agile within Dutch public sector, it could be useful to attempt to standardise as much as possible. Potentially standardisation could be achieve within separate ministries and government agencies (*Define a common view on the change and terminology - SF8.1*). By standardising these processes it should become easier for teams to work together and should allow any project supportive organisations to more easily support multiple teams and multiple projects.
It could be considered to standardise processes broadly within the Dutch public sector and design standardised processes for all ministries and agencies. This could

allow for a broad Dutch public sector supportive organisation and could make it more easily possible to execute projects that over-arch multiple Dutch public sector organisations.

Interviewees have mentioned it is important to keep the human factor of transition and (large scale) agile in mind. Several interviewees indicate that (large scale) agile frameworks not always keep the human side of the organisation in mind, but simply focus on organisational change and processes.
An agile process coach from case 3 explained, when talking about their transition from continuous deployment to SAFe, how they took three months to share their intended steps with the various teams and team members. They presented the current situation, the issues employees experienced, and the solutions the new (large scale) agile way of working would bring them. By actively engaging all teams and team members in their change process as well as the management and business side of the organisation the transition was widely accepted and embraced. In part this points back towards training and coaching employees within a transition, informing them early on and engaging them in the process helps them gain knowledge and experience in the (large scale) agile way of working to be adopted.

Interviewees stated that regardless of the situation the starting position for a (large scale) agile transition should be one singular framework. The main benefit of frameworks is that they have been extensively thought out to ensure the combinations of roles and processes made fit together well. Combining roles and processes from multiple frameworks runs the risk of combinations which might raise issues. Therefore there should always be a singular framework at the basis of a (large scale) agile transition, where necessary this basis could be expanded upon by adding roles or processes from a different framework where necessary.

The suggestions for (large scale) agile adoption discussed above seem to give a few quite clear direction to achieve a successful adoption of (large scale) agile. However, many interviewees have also stated that the adoption of (large scale) agile can not be defined with such a clear direction.
First of all, seven interviewees (mostly agile process coaches and managers) indicate that the way of working in a project should be secondary to the goals of a project. A (large scale) agile approach should not be adopted simply to be able to work agile, (large scale) agile should fit the goals and environment around the project. Interviewees have also indicate that (large scale) agile should be fit to the working method of the rest of the organisation, this seems contrary to the cultural change indicated before.
Secondly, six interviewees have indicated (large scale) agile should not be followed to the letter and should not be seen as an absolute truth. The (large scale) agile frameworks give some clear directions and have clearly designed processes to

achieve a successful (large scale) agile way of working. These processes might not always fit to the organisation and might simply not be possible, if this is the case there is no shame in adjusting the processes to more closely fit the organisational needs.

Interviewees indicated that (large scale) agile inherently is able to adjust its processes and roles to better fit the needs of a project and its members. The scrum master which keeps an eye on the processes should be able to spot issues early on and is directly tasked with attempting to resolve these issues.
Secondly, the retrospective at the end of each sprint allows for the entire team to reflect on the last few weeks and discuss what could be improved.

A final point raised by multiple interviewees is not to address the way of working used as (large scale) agile or directly use framework names. In multiple situations during my research when talking to either interviewees or other personnel within Dutch public sector organisation it was noticeable that employees where prejudiced against certain (large scale) agile frameworks with no clear reasoning behind it. There seems to be a resistance to change based on the names and terminology used. For example, within case 4 some processes of SAFe are in use, but the names of these processes are changed to prevent the association with SAFe. Case 4 simply calls their way of working *[Case 4] way of working*. Interviewees from case 4 indicate that this has greatly helped the adoption of (large scale) agile as they put a focus on their own organisational processes and how those could be adjusted. Besides, this prevents strict believers of (large scale) agile complaining about the way of working used.

Based on the results from the interviews it seems a successful adoption of (large scale) agile is achieved by finding the right balance between changing processes where necessary and maintaining current processes where possible. Finding this balance seems to be the major challenge of (large scale) agile adoption, how this balance can most successfully achieved is still uncertain and highly dependent of the situation. However, this section should shed some light on what points to focus on.

4.6.6 Additional findings from the interviews

All interviewees have shared their personal and professional vision of (large scale) agile and shared many suggestions or thoughts on how (large scale) agile could come into its own as well as possible within Dutch public sector.
Some of those visions and thoughts could not be encompassed within the questions as raised at the start of section 4.6. In this section I will discuss some of these additional findings and what this could mean for (large scale) agile adoption within

Dutch public sector.

Multiple interviewees have indicated that the transition to (large scale) agile is inevitable. It was indicated that if initiative was not taking top-down the initiative will be taken bottom-up. Within some of the cases you could see this happening as surrounding parts of the organisation were not yet willing to work according to (large scale) agile, but the IT executive teams did want to work according to (large scale) agile. Both approaches have their own benefits. When scaling up bottom-up you ensure the processes fit the teams as accurately as possible (*Self-organising teams - SF5.2*). On the other hand when scaling up top-down it is possible to ensure a more shared vision and way of working between all teams (*Define a common view on the change and terminology - SF8.1*). The most important thing to realise is that if the top of the organisation wants a hand in a transition to (large scale) agile they need to get involved sooner rather than later because otherwise the transition will start bottom-up if it has not already started.

As mentioned in section 4.6.3, according to the interviewees (large scale) agile allows for better steering during a project compared to more traditional methodologies. However, it was also indicated by an interviewee that even though (large scale) agile allows for more steering, steering does become more difficult with (large scale) agile. Within (large scale) agile projects there are a lot more levels which are available for steering. In traditional projects, steering would mostly occur on budget, time, and scope before the project would start. During the project some minor steering could happen on those three fronts, but would usually result in contractual disputes. Within (large scale) agile projects, steering can occur every sprint and to a much larger extend compared to traditional projects. This does mean that those in charge of steering the project are actively steering the project during its run. This requires more from the management of a project organisation.

Some interviewees have indicated that some (large scale) agile processes are already used in traditional projects, but are not seen as necessarily (large scale) agile. A clear example given are daystarts during which a team would at the start of the day shortly discuss what everyone would be working on each day, similar to the daily standup.
Finding these existing processes and moving them closer to a (large scale) agile way of working could help achieve (large scale) agile adoption quicker.

The bigger a project becomes the more dependencies start occurring between teams and the (sub)products they deliver. The way to deal with dependencies in traditional projects was waiting for team A to complete their work before team B would start working on their (sub)product. However within (large scale) agile projects, interviewees indicated that you would want both teams and tasks to be performed simultaneously where possible.

Interviewees stated, when working with dependencies, it is possible to base the initial work on assumptions as determined between the two teams or (sub)products. This would usually allow for a majority of the work to be performed already. As soon as the independent (sub)product is finalised the final details of the dependent (sub)product can be realised. This allows (large scale) agile teams to be able to continue working even though a dependency would be blocking their progress.

Some of the interviewees have indicated that the tools that are currently available to support (large scale) agile are sometimes lacking. None of the cases seem to have been able to integrate all necessary functionalities into one tool. The combination of tools used is different in each case. Interviewees indicated that the multiple tools do sometimes make it more difficult as information is spread over multiple sources.
Secondly it was mentioned that when scaling up agile different teams would be using different tools and this lack of standardisation raised some issues.
Lastly it was indicated that some teams started using tools which were not generally available in the organisation, yet they still needed the functionality. This raises some security and centralisation issues. It is wise for the broader Dutch public sector organisations to look into the tools currently used and the functionalities required by teams to determine which applications need to be centralised and made more widely available.

The last issue raised in the interviews is the ability to tender (large scale) agile projects. Dutch public sector organisations do not always have all necessary skills and knowledge in-house to complete a project. Dutch public sector organisations have to go through tendering processes to determine which market party is allowed to perform the project. A major issue within the tendering process is the level of detail needed when creating the initial tender offer. The contracts involved are worth millions if not billions. Occasionally, when the details were not clear enough in the initial tender offer, a losing party sues the government organisation in an attempt to win the contract anyway. A recent example is the cancellation of a $10 billion contract for cloud computing at the Pentagon after which the tender process needed to be restarted [7].
Interviewees indicate that this is a known issue within Dutch public sector IT projects. Dutch public sector organisations are tempted to take a more traditional approach for tendered projects, however case 3 has quite successfully tendered a part of their project to a market party.
Interviewees from case 3 indicate that even though at first sight tendered projects seem more fitting to a traditional approach, (large scale) agile can be a great addition to a tendered project. It should be clear from the start that the intention of the project is to use (large scale) agile, and allow for enough room in the tender offer to make

use of (large scale) agile. Suggestions given to help keep control of a tendered project with (large scale) agile were to provide a bonus if work is delivered on time and put a fine on work delivered a certain threshold after the initial deadline.
Secondly, contracts and progress could be evaluated on a shorter basis, for example after one or two increments. This allows for Dutch public sector organisations to tender external market parties on a project yet with the flexibility of (large scale) agile, and the necessary control for tendering.

Chapter 5

Discussion and evaluation

5.1 Initial conclusions

The initial goal of this paper was the creation of a adoption model for (large scale) agile. The main idea behind this model was determining which processes and roles of the various (large scale) agile frameworks would result in the most fitting (large scale) agile adoption for Dutch public sector IT projects.
However, during the case study research it was discovered that the goals of this research did not match the necessity of current (large scale) agile adoption in Dutch public sector IT projects. Interviewees indicated that a (large scale) agile adoption should not consist of multiple (large scale) agile frameworks and processes, but should instead be based on one central (large scale) agile framework.

As a replacement for the adoption model in this section I will attempt to answer the following question:
How (large scale) agile can be effectively adopted and/or scaled up within Dutch public sector organisations?
To do this, three sub-questions have been defined:

- How is (large scale) agile used in Dutch public sector organisations?
- What are the benefits of (large scale) agile compared to traditional project management?
- Would (large scale) agile be a good addition to Dutch public sector organisations?

5.1.1 How is (large scale) agile used in Dutch public sector

To discover how (large scale) agile is used within Dutch public sector, a case study has been performed encompassing four cases spread over two Dutch public sector organisations. Within the four cases (parts of) S@S or SAFe were used as the

(large scale) agile framework the way of working was based upon.
These four cases are all (large scale) agile organisations on a project basis or a portfolio basis. In all four cases the use of (large scale) agile is applied to improve collaboration between multiple teams working alongside on the same project or portfolio. The cases all have a different environment and therefore each have a different (large scale) agile approach.

All cases based their way of working on one (large scale) agile framework, each with different levels of detail and with different interpretations of the various roles and processes. Some cases added a few processes and roles from another (large scale) agile framework to their existing (large scale) agile framework to better fit their needs.

Within the (large scale) agile approaches found in the cases, a majority of the, in section 3.3, discussed challenges and success factors could be found. Additional codes with both positive and negative connotations on the achieved adoption of (large scale) agile were added during the analysis of the case study results. The eight most commonly occurred codes within the case study can be found in table 4.4.

5.1.2 What are the benefits of (large scale) agile compared to traditional project management

The main benefits of (large scale) agile according to the case study is the short iterative nature of (large scale) agile. Many interviewees indicated (large scale) agile was better able to steer a project compared to traditional project management. Interviewees indicate that, during a project, change always occurs and (large scale) agile would be better suited to dealing with this change than traditional project management methods would be.
This short iterative nature does not only allow steering on scope, budget, or planning, but also allows the teams to, where necessary, adjust processes and roles to better fit their needs within a project. This is seen as a great benefit of (large scale) agile as it allows the teams to adjust the way of working in such a way they can focus more on the goals of their project.

Interviewees indicate that within traditional projects many aspects of (large scale) agile are already used. For example, work is often split into smaller separate tasks and teams start their days with a central meeting. Some interviewees indicate traditional projects are already moving a little bit towards (large scale) agile, but (large scale) agile offers a better adjustable structure due to the short iterative nature.

Besides an increased possibility to steer (large scale) agile projects, interviewees mention an increased transparency, visibility, and predictability, a focus on quality

instead of budget or time, a decrease in employee stress, and increased flexibility when adopting (large scale) agile.

5.1.3 Would (large scale) agile be a good addition to Dutch public sector organisations

In the case study survey participants indicated that the current way of working in their organisation is not optimal. Participants have also indicated they think (large scale) agile would be a fitting way of working to their organisation.
All of the participants were already working in a (large scale) agile project or (sub) organisation, they indicated (large scale) agile works well in their own project or (sub)organisation, but think (large scale) agile should also be used more broadly outside their own projects or (sub)organisations .

Interviewees indicate that a current aversion to (large scale) agile could be due to the business- or management side of the organisation wanting more control over the project than they think (large scale) agile is able to offer. However, the interviewees in turn indicate that in their experience (large scale) agile adds more control over the project than the business- or management side of the organisation would have over traditional projects. A downside of the additional control, is the fact that being able to steer on more levels also makes it more difficult to accurately steer the project and more knowledge would be required to steer a project.

Several interviewees mention Dutch public sector IT projects frequently not achieving the goals they are supposed to achieve, simultaneously Dutch public sector IT projects would frequently be running over budget or out of time.
According to the *Rijks ICT Dashboard* [41], large IT projects of the Dutch public sector went 20% over budget in the first half year of 2021, and over half of the projects are projected to need more time. A total of 43% of all large IT projects is expected to have a delay of more than one year.
Interviewees indicate they think (large scale) agile would be better able to keep a project within it's budget and deliver on time. Some interviewees mention this would be in line with advices given by the *Adviescollege ICT-toetsing* [23].

According to interviewees, the transition to (large scale) agile on an IT-level is inevitable, this can also be seen within the case study. Some of the discussed cases have started making use of (large scale) agile as they themselves felt it would be the better way of working to more accurately reach their goals. Most commercial parties are also thinking about or have already adopted (large scale) agile. To be able to keep up with the market an affiliation with (large scale) agile is the absolute necessary minimum.
Implementing (large scale) agile more broadly in the organisation and offering top-

down support for it, will most likely results in a more standardised form of (large scale) agile. This should make it easier to implement and scale (large scale) agile in the future and make it easier for non-IT departments to collaborate with the standardised (large scale) agile projects or (sub)organisations.

5.1.4 How can (large scale) agile be effectively adopted and scaled up in Dutch public sector organisations

The results of the three sub-questions paint a positive picture of (large scale) agile within Dutch public sector organisations and IT projects. However, it is important to note that the participants of the case study are all mainly involved in IT projects or IT (sub)organisations. Therefore, the results of this study might not be applicable to Dutch public sector organisations as a whole, but will be relevant to Dutch public sector IT projects or IT (sub)organisations.

In section 4.6.4 eight core values and features of (large scale) agile were defined. According to the interviewees those core values are the most vital to adopt (large scale) agile in Dutch public sector organisations. Some of the core values follow logically from most (large scale) agile frameworks (short iteration, small teams, user feedback, automated testing), however some core values seem to be more directly affiliated with the Dutch public sector organisation culture within the cases.

To effectively apply (large scale) agile in relevant Dutch public sector IT projects or (sub)organisations a cultural change would be needed, as current Dutch public sector organisations do not seem entirely ready to embrace the (large scale) agile way of working. Participants indicate the main challenge comes with incorporating the business- and management side of the organisation in the (large scale) agile way of working. Incorporation of the business- and management side of the organisation (both non-IT) can not necessarily be achieved by applying (large scale) agile in all different (sub)organisations involved with a (large scale) agile project as it is unknown how well matched (large scale) agile would be to these departments.

The main issue within integration the business- and management side of the case organisations seems to be a lacking knowledge and experience with using (large scale) agile and integrating non-IT departments in a (large scale) agile way of working. Only a few of the interviewed case study participants have previously had a form of training to prepare them for their (large scale) agile role based on the used (large scale) agile framework. In general, it seemed as if not everyone involved with the (large scale) agile projects within the cases was aware of the way (large scale) agile worked and what that would mean for their role. This also resulted in those involved not always be aware how they could improve their (large scale) agile way of working or adjust their (large scale) agile way of working to better fit their needs.

Section 4.6.5 discusses potential steps to take to achieve a successful adoption of (large scale) agile, however the list of steps is non-exclusive and remains dependent on individual situations. The most surprising or influential suggestions will be shortly discussed here.
Management roles should become more leading roles, determining the high level direction of a project or (sub)organisation. Creating detailed specifications of functionalities, and the responsibility for those specifications and creation of those specified stories should be placed as low in the organisation as possible. The experts (developers) creating the solutions should also bear the responsibility to form the details of the solution.
Within all cases it was mentioned prioritisation was performed with a financial bias. The stakeholder with the highest budget had the most to say and would be more likely to get their functionalities produced, instead of focusing on the functionalities with the highest amount of business value. This is most likely something that is difficult to change as it is understandable a high paying stakeholder wants their money's worth of produced functionalities. However, a balance should be found which would allow for prioritisation on business value whilst still maintaining a decent level of stakeholder influence.
The two suggestions discussed above paint a clear picture of potential necessary cultural adjustment within Dutch public sector projects or (sub)organisation before being able to make full use of all (large scale) agile benefits.

Even though section 4.6.5 discusses some tips on achieving a successful (large scale) agile adoption there is no single perfect way to implement (large scale) agile or even use (large scale) agile frameworks. Adoption of (large scale) agile and the form a (large scale) agile way of working should have are highly dependent on the individual situations of the project or the (sub)organisation (large scale) agile is being implemented in.
First of all, (large scale) agile is not a goal on its own. The way of working used should always be secondary to the goals of a product, project, or (sub)organisation. It is important to create a way of working based on the processes and roles which are most needed to achieve those goals. It is very likely a (large scale) agile way of working will fit and support those goals, however (large scale) agile will not always be the perfect solution.
Secondly, a (large scale) agile implementation could work, but should always be update and adjusted when necessary. The (large scale) agile solution chosen could fit a project or (sub)organisation at one point in time, but it could later have different demands or have for example scaled up in a different way than initially expected. The benefit of (large scale) agile in this situation is the fact it also allows constant steering on the (large scale) agile solution used. Through, for example, retrospectives and

scrum master, which are inherent parts of (large scale) agile, the processes and roles deployed can frequently be evaluated and adjusted to better fit the potentially changing needs.

As stated before, any (large scale) agile implementation should be of added values to better support and achieve the goals of a project or (sub)organisation. The way of working is secondary to the goals and should be continuously adjusted to ensure the way of working will always optimally help support and achieve the goals of a project or (sub)organisation.
To be able to find this best fitting (large scale) agile implementation and to be able to continuously steer the (large scale) agile implementation it is important to have well informed and well trained (large scale) agile personnel. It is impossible to know how a certain (large scale) agile implementation could improve your way of working if you are not even aware of the existence of the (large scale) agile implementation. This can be achieved in two steps. First of all a centralised, potentially a nationwide, Dutch public sector (large scale) agile support organisation with specialists in various (large scale) agile implementations. This organisation could help guide projects or (sub)organisation wanting to change their way of working in finding the best fit and could do the same for projects or (sub)organisation that want to improve on their current way of working.
Secondly those involved in the individual (large scale) agile implementations should be trained in their respective roles. This could, for example, ensure a scrum master is fully capable of continuously improving the way of working or could give a product owner some useful tools to increase efficiency of prioritisation.

The results discussed in section 4.6 should provide an insight into the practicalities behind (large scale) agile adoption and usage. Especially section 4.6.5 and section 4.6.6 can provide some useful directions when considering a transition to (large scale) agile frameworks.

With regards to scaling up an (large scale) agile implementation it seems to be most vital to transition slowly and carefully (SF6.1). Within the case study only practical experience with S@S and SAFe has been collected, therefore it is uncertain whether these conclusions also apply to the other (large scale) agile frameworks as discussed in section 3.2.
For implementing (large scale) agile on a large scale directly the same applies as when scaling up an existing (large scale) agile implementation.
It is important to take a transition to (large scale) agile in small steps. Most (large scale) agile frameworks have documentation on the best way to implement their respective framework. These always take it in small steps.
For example, implementing or scaling (large scale) agile could look like the following steps:

1. Begin with one scrum team
2. Adjust processes and surroundings until the scrum team functions well
3. Add multiple scrum teams working alongside each other
4. Introduce a scrum-of-scrums team with the necessary additional support
5. Adjust processes and surroundings until the scrum-of-scrums team functions well
6. Add multiple scrum-of-scrums teams working alongside each other
7. Introduce a scrum-of-scrum-of-scrumts team with the necessary additional support
8. etc.

The above example is solely based on S@S or similar implementations, however when the size of a project or (sub)organisation increases it could also be considered to make a transition to a different (large scale) agile framework. In this example I would suggest considering introduction of SAFe at step 7, but like mentioned before this is dependent on the individual situation of the (large scale) agile implementation and the project or (sub)organisation it is being used in.
Even though the example is based on a S@S implementation this does not mean that S@S is seen as the perfect start point of a (large scale) agile adoption. The individual guidelines of (large scale) agile frameworks should be taken into account. For example, the most basic version of SAFe (SAFe Essential) starts with a minimum of five teams and would thus not be possible in our given example.

The combination of creating an environment that is able to change, well informed and trained individuals, and a careful transition should provide Dutch public sector projects and (sub)organisations with the right tools to be able to achieve a successful (large scale) agile adoption. In the end an effective (large scale) agile adoption comes down to finding the right balance between changing processes and roles where necessary and maintaining a well functioning status quo where possible. Each individual situation is different, but when projects or (sub)organisations are willing to change, well informed, and mindful, (large scale) agile could provide some great benefits over more traditional ways of working.

5.2 Evaluation

5.2.1 Evaluation method

As there is no product or design to demonstrate the fourth step of the DSRM can not be executed. However, the fifth activity in the DSRM is evaluating the results of the demonstration. In this paper the results have not been applied in a demonstration,

however the evaluation activity will be performed based on the findings.
The initial results, as discussed in section 5.1, will be evaluated in two sessions. The first sessions is with a (large scale) agile working group within a Dutch public sector agency, members from three of the four cases are represented in this working group. The goal of the working group is to be able to get (large scale) agile more widely adopted within Dutch public sector. Some of the members of this working group were interviewed as part of the multiple case study research.
For the second session all interviewees, additional stakeholders, and other interested individuals are invited.
Participants of both sessions will receive the, at that time, current version of this paper, an executive summary of the initial results and advice to the Dutch public sector, and a introductionary presentation. After the introduction the participants of the evaluation sessions are asked to share their thoughts on the results and advices, ask question, and discuss the results.
Based on the discussions and results of the evaluation sessions the initial result will be adjusted where necessary. After the first session the executive summary, initial results and advice, and presentation will be adjusted to incorporate the feedback from the first session in the second session.

The initial results as discussed in section 5.1 have been translated into six advices for the executive summary as sent to the two evaluation session participants. The advices are as follows:

1. Make use of (large scale) agile, it has significant benefits and could help Dutch public sector projects and (sub)organisations better achieve their goals.
2. Make use of the (large scale) agile ability to continuously steer and ensure adjustment of (large scale) agile processes and roles even after initial implementation.
3. Make sure all relevant parties are well trained and informed about the (large scale) agile implementation.
4. Give management a more leading role, providing high level directions and allowing responsibility to be with those executing a project or creating the product.
5. Implement and scale (large scale) agile slowly and carefully.
6. Consider changing terminology to prevent those with a aversion to (large scale) agile from not wanting to transition to an improved way of working. Truly make the way of working your own.

5.2.2 Evaluation results

In both evaluation sessions the participants agreed with the initial results, conclusions, and advices. Based on the discussions it seems the focus should be on making effective use of iterative improvement processes within agile, like retrospectives and scrum masters. Secondly a strong sponsorship and support from management seems to be important to achieve a successful (large scale) agile adoption.
Participants agreed with basing the way of working on a single (large scale) agile framework, however it was indicated that flexibility in the way of working was equally important. To adjust the way of working processes and roles from other (large scale) agile frameworks could be appended, but when many changes are made it needs to be reevaluated whether the right (large scale) agile framework was chosen to function as a basis.

Regardless of the positive attitude towards (large scale) agile, (large scale) agile is not the solution for every single (IT-based) situation. The main advice of usage of (large scale) agile or other ways of working is dependent on individual situations includes making use of traditional project management methods.
Secondly, participants of the evaluation sessions indicated that it will definitely happen during a project that processes and roles could vary even between sprints, this puts even more emphasis on the need of retrospectives and scrum masters.

Participants discussed that regardless of whichever (large scale) agile or other project management approach would be chosen sponsorship from management would be necessary. A (large scale) agile transition should be supported from both top-down as bottom-up. The details of the transition should be determined bottom-up, best fitting the needs of the individual teams. The general direction and especially facilitation of the (large scale) agile transition should be provided top-down.

A final thing discussed in the evaluation sessions was the principle of *Bricolage* project management [64]. The main idea behind of Bricolage project management is the fact that there is a wide variation of project management best practices, frameworks, or approaches. As projects will never run under perfect conditions, therefore Bricolage talks about using best-fit practices over best practices. Fitting various pieces from various best practices to best fit the situation at hand.
Bricolage project management seems to match a few of the results as found in this paper, however it is still important to main a central basis to your approach.

5.3 Discussion

The initial idea behind this paper was to create an adoption model for (large scale) agile in Dutch public sector project, programmes, or organisations. I expected this

adoption model to consist of various processes from the various (large scale) agile approaches which would be most fitting to Dutch public sector (sub)organisations as a whole. However, during this research I discovered that there is no such thing as one perfect solution. The solution to an effective adoption of (large scale) agile has little to nothing to do with the (large scale) agile approach chosen. The most important thing is having capable team members and managers and having a flexible enough organisation to adjust their way of working when necessary.

Initially I did not necessarily expect (large scale) agile to be the holy grail which could solve all problems within Dutch public sector IT-projects, but I did expect simply implementing it could resolve many issues. Contrary to this I discovered that (large scale) agile is already in use, but doesn't solve the issues mostly holding back development. As can be seen from both the literature research as from the multiple case study research (large scale) agile works and helps improve organisations, but organisations need to be ready and open to making use of (large scale) agile.

The lower levels of Dutch public sector (sub)organisations seem to be ready to adopt a (large scale) agile way of working, but the rest of their organisations does not seem to be ready for this yet. The benefits seen by the participants of the case study are not always seen outside of their (IT) organisation, or the benefits are not expected to outweigh the extra hassle (large scale) agile would bring.

Most challenges and success factors of (large scale) agile found in the literature research (section 3.3) can be somewhat logically derived from either the various (large scale) agile approaches or from general principles of project management. Therefore it was not very surprising almost all challenges and success factors were also found in at least one of the four cases. It could be hoped that challenges logically derived from general project management principles might not be as frequently present as newer challenges raised by the adoption of (large scale) agile, however this did not seem to be the case.

The line between traditional project management approaches and (large scale) agile approaches seems to be smaller than initially expected. Work frequently gets split up in smaller packages, teams utilise the start of the day to quickly discuss what needs doing, evaluation sessions are planned. However, the strong iterative nature of (large scale) agile is not as present in more traditional approaches and seems to be a great benefit of (large scale) agile approaches over traditional approaches. Due to frequent opportunities to steer and readjust a project, programme, or organisation within (large scale) agile, it makes it easier to (re)prioritise tasks, reach a minimal viable product, and ensure an effective way of working.
However, more frequent steering seems to also be more challenging, even though it should allow for smaller adjustments to be made at a time. I would expect smaller adjustments to be easier to make and to be more easily accepted by the various

actors and stakeholders in an organisation. The main downside of more frequent steering seems to be the fact that management needs to pay attention to more factors and smaller factors as they also need to be able to steer those smaller factors. With less frequent steering, steering could still happen on a more general level and thus fewer factors needed to be taken into account which would save time and resources.

The main problem with (large scale) agile adoption at Dutch public sector (sub)-organisations seems to be an unwillingness from the business and management sides of the (sub)organisations trying to adopt (large scale) agile. This unwillingness seems to stem from a lacking knowledge of and experience with (large scale) agile. Interviewees indicate management might want more control over a project, programme, or organisation than they think (large scale) agile can offer. Interestingly the interviewees also indicate (large scale) agile allows for more control over a project, programme, or organisation than traditional approaches offer. It could be interesting to discover where this difference in expectations of (large scale) agile comes from, why would the interviewees think (large scale) agile offer more control and management think it would not offer the right amount of control?

A common point of discussion is a potential cultural change needed within Dutch public sector (sub)organisations to be able to better adopt (large scale) agile. I also concluded a cultural change would be required in Bolhuis [10]. However, it is uncertain what this cultural change would exactly entice. Dutch public sector (sub)organisations do not seem entirely ready to start adopting (large scale) agile in a broader sense. Especially the business and management side of the cases seemed to not be ready or willing to adopt (large scale) agile. Dutch public sector (sub)organisations need to look into the possibilities of matching the way of working of the management and business side to the (large scale) agile way of working and to the iterative nature of the (large scale) agile way of working. Different departments will have different requirements for their way of working, so a middle ground needs to be found between the different departments on how they can best link up their ways of working and support each other.
The benefits and urgency of (large scale) agile are currently still lost on the management and business departments. However, if they are provided with the necessary information and knowledge (large scale) agile could benefit them as well.

It is difficult to pinpoint what should change for an effective adoption of (large scale) agile, both within the Dutch public sector (sub)organisations as within the Dutch public sector culture. I think the iterative and reflective nature of (large scale) agile are the two key features to achieve an effective adoption. Being able to freqeuntly make small adjustments should help a team or an organisation as a whole to continuously keep working to as close to a perfect way of working as possible. A

willingness to change or to let go of what was once thought was the perfect solution is a key part of a future Dutch public sector culture.

A second cultural change within the Dutch public sector would be moving away from the plethora of financial incentives. Contrary to a private sector organisation, if a department does not spend all the funds allocated to them they run the risk of receiving less funding the next year. Money seems to be the deciding factor for prioritisation in many situations. In all four cases it was mentioned that who pays decides. Prioritisation was not always happening based on the full value of a feature, but on how much money would be paid to realise a certain feature.
Development teams should be able to control prioritisation themselves and prioritise features based on their business value. This should allow the team and the project to focus on reaching their main goal and delivering a high quality product. A way should be found through which the various stakeholders still get their moneys worth, but are not in full control.

Initially, I expected to find the solution to the problems at the Dutch public sector by combining features from multiple (large scale) agile approaches into one perfectly fitting solution. I expected a mix of various (large scale) agile approaches would be able to provide all processes and roles necessary to better support Dutch public sector organisations developing IT products. However, many interviewees indicated it would be better to stick to a single (large scale) agile approach and expand upon it when necessary.
What I initially not took into account is the fact that the (large scale) agile approaches discussed in this paper have been created over a long period of time with the sole purpose of being a (somewhat) perfect solution for managing a large scale (IT) project, programme, or organisation. These (large scale) agile approaches are created in such a way that they should be a complete solution and should be able to take all important and relevant facets of project management into account.
Any way of working should be as complete as possible to ensure all important and relevant facets of project management are taken into account. Therefore, there should always be a central basis to the way of working chosen, one single project management approach which covers all those facets. Once that basis is in place it can be somewhat expanded upon by processes, tools, and roles from other project management approaches. Interviewees also indicated that there should be a limit on how many additional processes, tools, or roles are added to the basis, because if too many of those are necessary maybe a different basis needs to be picked.
Even though these results went against my initial thoughts on (large scale) agile within the Dutch public sector, it has become an important factor in the results of this paper. Adoption of (large scale) agile is only partially dependent on (large scale) agile itself, a more important factor is the way it is adopted. Especially the consid-

eration of Bricolage project management strengthens my thoughts that there is not one perfect solution. Finding, creating, the perfect solution is in the hands of those managing and controlling the project, programme, or organisation. A successful (large scale) agile adoption is just as much dependent on the (large scale) agile approach chosen as it is on those choosing and implementing the (large scale) agile approach.

In general, the results and conclusions in this paper feel like stating the obvious over and over again. There have rarely been any findings which were surprising or seemed illogical to me. However, even though the results and conclusions might feel entirely logical, utilising these findings in practice seems to be more challenging than it looks. On paper the adoption of (large scale) agile looks and sounds like an easy decision, but in reality a lot more is involved with a successful (large scale) agile adoption.

A successful and effective adoption of (large scale) agile within Dutch public sector organisations is heavily dependent on those managing and working on the transition. An organisation needs to be ready and willing to adopt (large scale) agile otherwise it has no use trying to make use of a (large scale) agile approach. So, even though (large scale) agile can be beneficial to Dutch public sector projects, programmes, or (sub)organisations they need to come to this conclusion by themselves as well.

Chapter 6

Conclusion

6.1 How can (large scale) agile be effectively adopted and scaled up in Dutch public sector organisations?

The initial goal of this paper was the creation of an adoption model to aid in the adoption of (large scale) agile within Dutch public sector or other public sector organisations. During the case study research it became apparent an adoption model might not be what is needed at the moment to help improve adoption of (large scale) agile within Dutch public sector. Therefore, based on the case study results the goal was readjusted towards attempting to answer the following question:
"*How can (large scale) agile be effectively adopted and scaled up in Dutch public sector organisations?*"

The needs for an effective adoption and scaling up (large scale) agile are quite similar to each other. The only difference being that when scaling (large scale) agile up the environment should already be better suited to (large scale) agile and would most likely need less adjustments.

Similarly to the findings in section 3.4, the interviewees indicate usage of (large scale) agile within a project has a positive effect on the execution and outcome of the project. However, the results of this paper are only based on participants from IT projects or (sub)organisations, so it can not be concluded whether (large scale) agile should be adopted outside IT projects or (sub)organisations.

In this paper eight different (large scale) agile approached are discussed, within the case study research only two of these approaches were represented. No conclusions can be drawn on which (large scale) agile approach would be a better fit to Dutch public sector IT projects or (sub)organisations. The usage of any kind of project management approach is heavily dependent on the individual situation of projects or (sub)organisations. Therefore, an effective adoption of (large scale) ag-

ile and scaling of (large scale) agile approaches is not dependent on the approach, but on the environment it is being applied in.
Usage of (large scale) agile is not a goal in itself, (large scale) agile or any other kind of project management approach should only be applied if it is beneficial to a project or (sub)organisation. Usage of (large scale) agile should be secondary to the goal of a project or (sub)organisation

However, a major benefit of (large scale) agile approaches is its ability to frequently (re)adjust. Readjustment within (large scale) agile encompasses both readjustment of the project details (scope, budget, planning) as project environment (processes, roles). This means that an (large scale) agile approach is able to frequently check whether the (large scale) agile approach used is still beneficial to (goal of) the project or (sub)organisation.

Therefore, a successful (large scale) agile adoption is dependent on how well able an organisation is in evaluating their needs and their way of working.
To be able to find the most fitting way of working to the goals of an organisation the right information and training is required for those tasked with adjusting the way of working to the goals. Knowledge of multiple (large scale) agile or any other kind of project management approaches is vital as it gives an insight into the possibilities for a fitting way of working.

Most (large scale) agile or other project management approaches and best practices are thoroughly thought out to ensure the approach or best practice is able to accurately take care of all relevant parts of managing a project or (sub)organisation. These approaches or best practices are designed with a certain background in mind, but also consider parts of project management that are less on the surface yet still important to consider.
Due to the thorough nature of most of these project management approaches it is important to let one approach form the basis of the way of working, this basis should (naturally) be the most fitting approach to the needs of the project or (sub)organisation. The features of this basis approach can be appended on by introducing additional processes or roles from other approaches where necessary. However, if the basis approach is altered too much it should be considered whether or not the right basis approach was chosen.

Even thought a successful (large scale) agile adoption is dependent on the situation and the environment it is being adopted in, some guidelines can be given. In section 3.3 20 success factors split over eight success factor categories (table 3.1) and 36 challenges split over ten challenge categories (table 3.2) have been defined. These success factors and challenges should provide some insight in what to look out for or what to utilise when adoption (large scale) agile.
Based on the multiple case study the following factors have been raised as important

features of (large scale) agile for adoption within Dutch public sector IT projects:

- *Train and coach people - SF2.1*
- *Acquire knowledge on (large scale) agile - SF2.2*
- *Management commitment - SF7.1*
- *Short iterations - SF4.2*
- *Small team size - SF5.1*
- *Effective customer involvement - SF1.3*
- *Facilitate Communication, coordination, and control - SF1.1*
- *Automated testing (prevent Difficulty automating testing - C8.3*)
- *Management should give direction (prevent Lack of trust - C1.3*)
- *Leave responsibility & execution to the specialists (prevent Teams lacking autonomy - C1.2* & *Lack of trust - C1.3*)
- *Commit to targets and a budget*

The challenges of (large scale) agile found within the multiple case study are the following:

- *Controversies between organisational and agile culture - C2.2*
- *Minimal collaboration and knowledge sharing between teams - C1.1*
- *Difficulty scaling agile to non-IT departments - C2.1*
- *Mismatch between agile and client organisation - C2.3*
- *Determining readiness of the organisation to transition - C4.3*
- *Lacking leadership participation - C9.1*
- *Lacking management support and sponsorship - C9.2*
- *Teams lacking autonomy - C1.2*
- *Lack of trust - C1.3*

All-in-all a transition to, or scaling up, (large scale) agile is heavily dependent on the specific situation and environment the transition is taking place. The way of working chosen within a project or (sub)organisation should at all times be secondary to the goals and needs of a project. A way of working needs to match the situation, but it is highly unlikely a single project management approach could satisfy all the needs of a project or (sub)organisation.
Even though a combination of approaches or best practices will be necessary it is important to pick one approach as the basis for the way of working. This basis approach can be expanded on or added onto by processes or roles from other approaches and best practices where necessary. However, the basis approach should be maintained as much as possible, if too many processes or roles from other approaches and best practices are added it needs to be considered if the basis approach is the best fitting approach.

This is not only applicable when transition to (large scale) agile, but is applicable to usage of any kind of project management approach or best practices.

Some guidelines can be given for a successful transition, but the most vital part of a successful transition is still being informed and experienced enough to find the best fitting approach to the project or (sub)organisation situation at hand. Once a transition is completed the way of working needs to be frequently evaluated to ensure it is still fitting to the goals of the project or (sub)organisation.
Here (large scale) agile has a benefit over other project management approaches as frequent evaluation of the utilised way of working is embedded in the (large scale) agile approach. The retrospectives process and scrum master role within (large scale) agile play a vital role in achieving this flexibility and the ability to readjust the way of working where necessary.

6.2 What can Dutch public sector organisations do to more effectively adopt and scale up (large scale) agile?

In this section I will shortly discuss some advices to Dutch public sector organisations on how they can improve their (large scale) agile adoptions.

6.2.1 Utilising (large scale) agile

It has become clear from the multiple case study that within the cases (large scale) agile is seen as a better way of working compared to more traditional approaches. As in most situations work is split up into smaller sections (large scale) agile seems to be a fitting approach to define the sections, ensure efficient execution, but especially to frequently adjust sections or processes where necessary.

It is advisable to look into the possibilities of adopting (large scale) agile in large IT projects, programmes, or organisations with the Dutch public sector. The core values of (large scale) agile as defined in section 4.6.4 are a good guideline for this (large scale) agile adoption. However, it should be kept in mind (large scale) agile is not a goal in itself, but should be secondary to the goals of the project, programme, or organisation.

6.2.2 Steering (large scale) agile

A transition to (large scale) agile is a complex process and it is important to find the right balance between changing processes where necessary and maintaining

the status quo where possible. The iterative nature of (large scale) agile should aid in finding this balance and adjusting this balance when necessary. Especially the retrospectives at the end of a sprint in combination with the scrum master could prove very valuable in ensuring a successful and efficient way of working is created and maintained.

6.2.3 Training (large scale) agile

To successfully adopt (large scale) agile and continuously adjust (large scale) agile where and when necessary an experienced and trained staff is vital. Staff will not be able to adjust the way of working if they are unaware of the the potential existing solutions. Secondly a strong supportive organisation should be in place to help answer question or resolve problems with the way of working when necessary.
This could take the form of a central organisational unit with the knowledge of existing frameworks and approaches and with the necessary skills to train and inform staff that would be working with any of these frameworks or approaches.

6.2.4 Managing (large scale) agile

To most effectively make use of the skills and knowledge of those in the organisation actually working on or developing a solution it is important the exact execution of tasks and the responsibility to do so is posted as low in the organisation as possible. Detailed specification of tasks, the responsibility over this specification, and the responsibility for executing the tasks should be with the team members tasked with the execution of the tasks. It could prove beneficial to not make a singular person responsible, but to make a team as a whole responsible, however in this case it becomes even more important team size remain small (maximum of ten team members).

By leaving specification and responsibility lower in the organisation management could function more from a leadership perspective. Management can thus focus on the big picture, and look ahead to determine the high level directions and goals of a project. Besides this, management could play a serving role to the teams and adjust processes where possible and ensure a proper facilitation of teams.

6.2.5 Implementing (large scale) agile frameworks

There are a lot of (large scale) agile frameworks and approaches to chose from. The core values of all these frameworks are largely overlapping with the core values as discussed in section 4.6.4. Through the case study only knowledge on SAFe and

S@S was gained, therefore I will limit my advices to these two frameworks as it is currently uncertain how the other (large scale) agile frameworks perform in practice.

It is most likely easiest to start a transition to (large scale) agile with a singular agile/scrum team. This first team should provide the first insights in (large scale) agile and how it could function within the organisation. As soon as this team functions satisfactorily additional teams can be added alongside and combined in a SoS team.
As soon as an SoS team is introduced it should be considered to make use of a shared increment planning where all teams can come together every 2-4 sprints to discuss potential impediments and/or collaboratory issues. As soon as the SoS team is functioning satisfactorily it could be expanded upon by adding more SoS teams and combining them in a scrum of scrum of scrums (SoSoS) team. However I would advice to look into the smallest variant of SAFe (Essential SAFe) instead. SAFe offers more tools and processes to support multiple teams working on the same project compared to S@S.

At all times only one (large scale) agile framework or approach should be at the basis of the way of working. However, the way of working could be expanded upon by processes, tools, or roles from other (large scale) agile frameworks.

6.2.6 Do not use (large scale) agile

Lastly, it should be kept in mind that the adoption of (large scale) agile is not a goal in itself. Any (large scale) agile framework or approach should be supportive of the goal of a project, programme, or organisation and should fit the current and future needs of the organisation. Secondly, it was found that at times there is an aversion to (large scale) agile, certain (large scale) agile frameworks, or certain (large scale) agile processes. This is most likely due to the high buzzword factor *agile* still has.

The main goal of readjusting a way of working is to for example improve efficiency, effectiveness, morale, or reduce costs. It is important to all together be able to speak the same language and use the same terminology when talking about the same things. However, if there is a strong aversion to (large scale) agile, the (large scale) agile processes and roles could simply be adopted as a more personalised way of working instead of calling it (for example) S@S or SAFe.

6.3 Generalisability of findings

The results and conclusions drawn in this paper are mostly based on the multiple case study executed. However, it is interesting to know how applicable these findings are outside of the four cases discussed. Following the model of Seddon and

Scheepers [62] I will briefly discuss the generalisability of the findings of this paper. As there are little clear quantitative conclusions to be drawn in this paper, the *ABC* path of Seddon and Scheepers will be used.

In section 6.1 and section 6.2 I lay out the main conclusions and advices of this research. These conclusions are based on the four cases as laid out in section 4.4 and the thoughts and experiences of the interviewees of these four cases.

The four cases discussed are a varied representation of Dutch public sector projects. The cases have been gathered from multiple Dutch public sector organisations and therefore do not portray the situation in one singular Dutch public sector organisation. The size of the cases vary greatly, as can be seen in table 4.1. A wide variety of team members of the cases was involved in the research, as can be seen in table 4.2. Two of the cases have outsourced the execution of their cases to an external party, one within the Dutch public sector and the other to a commercial party. One of the cases was developing and maintaining various supportive products for other projects running within the same Dutch public sector organisations, whereas the other cases were actively developing and maintaining one or multiple products meant for external use.
Even though all four cases took place in different environments and under different circumstances the general results of this paper were mentioned in all four cases.

The main issues surrounding the adoption of (large scale) agile in the four cases discussed are issues which will very likely arise in any Dutch public sector project, programme, or organisation. For example, the way projects are financed and budgeted is standardised over Dutch public sector organisations. IT-projects and the products created will all have a similar kind of user-base and will be held to similar standardised Dutch public sector standards.
All participants and all cases are directly related to IT projects, programmes, or organisations. Conclusions can not be drawn on non-IT departments and can therefore also not be generalised to non-IT Dutch public sector projects, programmes, or organisations.

The most relevant issues and the potential causes of these issues do not only occur in the four cases discussed. As described above the environment of the four cases will very likely be similar in other Dutch public sector IT projects, programmes, and potentially even organisations. Therefore the results of this paper are not only applicable to the four cases discussed, but are applicable in the broader Dutch public sector.

In Bolhuis [10] I discuss a few different cases within Dutch public sector. The conclusions and results of Bolhuis [10] seem to overlap with the results found here, further strengthening the generalisable applicability of these results.
As no other research has been found looking into the combination of (large scale)

agile and (Dutch) public sector organisations, the generalisability can not be further expanded upon by additional literature.

6.4 Limitations

The original goal of this paper was the creation of an adoption model to aid Dutch public sector organisations with an effective adoption of (large scale) agile. Therefore the DSRM was chosen to execute this research. Within the steps of the DSRM a multiple case study was executed. During this multiple case study it became apparent an adoption model would not be the right solution to aid Dutch public sector organisations with an effective adoption of (large scale) agile. Therefore it was decided no artefact would be created, instead advices were given on how to achieve an effective adoption of (large scale) agile.
Due to the focus of the DSRM on artefact creation it might not have been the best methodology to achieve the final conclusions as presented in section 6.1. The advices given and conclusions drawn could also be seen as an artefact and with the combination of DSRM and a multiple case study the conclusions of this paper are still relevant.
However, a multiple case study research approach aimed at similar results of this paper could present more concise conclusions and might be able to present more measurable results.

In section 3.2 eight different (large scale) agile approaches are discussed. Initially S@S was not included in this list as it was not included in the paper by Alqudah and Razali [1]. Besides S@S, SAFe was the only other (large scale) agile approach found within the multiple case study in section 4. With addition of the case studies in section 3.4, one more (large scale) agile approach was discussed. For the other (large scale) agile approaches discussed only theoretical information is available. However looking at the theoretical descriptions of the various (large scale) agile approaches, most of the results of this paper will be applicable to the (large scale) agile approaches not featured in either of the case studies. However, not all results in this paper will be applicable to all different (large scale) agile approaches.

Additional limitations arise within the multiple case study.
All participants of the multiple case study were from the IT side of their respective organisations. Therefore the results of this study might not be applicable outside of IT projects or IT (sub)organisations.
Three of the four cases are within the same Dutch public sector organisation. The remaining case study is in no way related to the other three cases. Therefore the generalised conclusions drawn in this paper might not be applicable for all Dutch public sector organisations.

Considerably more participants of the case study were from either case 2 or case 3. From case 1 three people were interviewed, from case 2 eight, from case 3 nine. and from case 4 two people were interviewed. Therefore, the results of the case study might be skewed towards case 2 and 3.

The main limitation of Chapter 3 is the available scientific literature. Large scale agile is still a very young subject and there is not a lot of scientific literature available. Similarly, IT projects within the public sector is also a subject with limited available scientific research. This can also be seen in appendix A. Due to this small amount of available scientific literature it is possible there exists an overlap between the sources used to determine success factors and challenges and the case studies used to evaluate these success factors and challenges. The case studies are not directly used when determining success factors and challenges, but it is possible the review papers used in section 3.3 are based on some of the case studies as used in section 3.4.
Not all of the results and conclusions as discussed in this paper have been explicitly proven, some conclusions are based on a combination of proven results in this paper and insights based on these results.
The results and conclusions of this paper are mostly based on scientific knowledge acquired from private sector IT projects, it is uncertain how well applicable this knowledge is to the intended public sector IT projects.

6.5 Implications for research

This paper gives a first insight into the possibilities of (large scale) agile within (Dutch) public sector organisations. However, a lot remains unknown or uncertain. As (large scale) agile is a *young* subject there is little available scientific knowledge and thus there is a wide variety of potential future research.

First of all it could provide some interesting and potentially substantial results when this research is executed in a larger format, potentially with more cases or participants. It could especially provide additional insights if at the start the goal of the research would not be on creation of an adoption model.
This larger format research could also gather more information on usage and effectiveness of (large scale) agile outside of IT-projects or IT (sub)organisations.

A major addition to knowledge of (large scale) agile would be a comparison between the various available (large scale) agile frameworks. A lack of comparison between the various frameworks was already identified as a challenge (*Lack of comparison between frameworks - C7.4*), but also noticeable during the case study. Due to the lacking current knowledge on the various (large scale) agile frameworks it is uncertain which would best fit the various needs of various (public sector) organisa-

tions or projects. Considering designing a effective way of working is dependent on having the right information on the available (large scale) agile or other project management approaches or best practices a comparison of these available approaches and best practices is an important addition.
This comparison could also give a better insight into where transition between different (large scale) agile approaches could be relevant, for example when it would be relevant to switch to a different (large scale) agile approach during scaling another (large scale) agile approach.

Lastly, based on the literature research in Chapter 3 two more interesting fields of potential study were found.
Little is know about the combination of DevOps and (large scale) agile. It seems to be incorporated in the various (large scale) agile approaches, but little is actually known. This could potentially be incorporated in the comparison research of the various (large scale) agile frameworks as mentioned before.
Most research into (large scale) agile is performed on a team or organisational level, but little is still known about the effects of (large scale) agile on individuals.

6.5.1 Potential research questions

Based on the implications for research a few research questions can be proposed for future researchers interested in this field.

How can (large scale) agile be effectively adopted and scaled up in Dutch public sector organisations?
A continuation or repetition of the research executed in this paper. A central adjustment necessary would be a change in approach to more closely fit the research to the intended goals. Secondly it could proof very useful to expand the size of the case study executed to get a larger and potentially more diverse insight in the use of (large scale) agile in Dutch public sector organisations. It is advisable to take a multiple case study approach, as this will give the most direct insight in the practicalities of the utilised (large scale) agile framework(s). Where possible a longitudinal multiple case study would also allow to get more insights in the long term developments and use of (large scale) agile.

How can (large scale) agile be effectively adopted and scaled up in non-IT organisations?
This paper has exclusively focused on the usage of (large scale) agile frameworks in IT organisations. However it could be of interest to discover how (large scale) agile frameworks could fit to non-IT or non-technical organisations. These are usually the kind of parties required to support the IT organisations discussed in this paper. Potentially adopting (large scale) agile in these non-IT organisations and their IT

counterparts could improve the efficiency of both organisations as a whole.

How do the various (large scale) agile frameworks compare to one another?
There is little scientific knowledge on the comparison between the various available (large scale) agile frameworks. This makes it more difficult for projects, programmes, or organisations to pick the most fitting framework for their situation. A comparison of the various available (large scale) agile frameworks could provide more insights in the vital factors of (large scale) agile and aid in a broader adoption of (large scale) agile.

How can (large scale) agile frameworks be combined with DevOps practices?
There is little to no knowledge available on how DevOps can be utilised within (large scale) agile frameworks, other than the claims of the various frameworks themselves.

What is the effect of (large scale) agile on individual team members (and their efficiency)?
Most research is performed on a team level or an organisational level. However for the strength of a team or organisation is dependent on the strength of the individuals forming the team or organisation. It could be interesting to know more about the effect (large scale) agile has on various kinds of individuals in different kind of roles and how (large scale) agile frameworks could (potentially) better tailor to the needs of those individuals.

How can (large scale) agile be effectively adopted and scaled up in private sector organisations?
This paper has a focus on Dutch public sector organisations. However little is known about the effect of (large scale) agile on private sector organisations.

6.6 Implications for practice

The main implications for practice for these paper are related to potential ways of working to be adopted by a project, programme, or organisation.

The use of (large scale) agile is found to have a positive effect on Dutch government IT projects, programmes, and organisations. Organisations can improve their existing way of working by adopting features of (large scale) agile or by adopting (large scale) agile as a whole. However, an adoption of (large scale) agile should not be done with the inherent goal of improving the way of working. The way of working should at all times be secondary to the goals of the project, programme, or organisation. When considering a readjustment to the way of working it is important to consider what goals or what processes this way of working should support or should improve upon.

Reconsidering a way of working should always be an ongoing process, the demands for a way of working differ over time. Organisations need to be flexible, evaluative, and have a willingness to change and to keep changing (small) details of their way of working to ensure an effective way of working over time.
Some principles of (large scale) agile can play a role in this. Especially the iterative nature of (large scale) agile allows for frequent adjustments to the way of working and through retrospectives the way of working is frequently evaluated. Roles like scrum masters and agile coaches allow a sideline view on the processes and can aid in finding and relieving potential pain points in the way of working.

However, it is important to remember that (large scale) agile or any other way of working is not a magical cure to existing problems. Every single situation requires different measures and every project will require a (slightly) different way of working to be most effective. Therefore to successfully improve on a way of working it could be increasingly beneficial to have knowledge of multiple potential frameworks or approaches as this could help gain inspiration on how to improve the existing way of working.

When readjusting a way of working this should mostly be based on a singular framework or approach. Most frameworks are created in such a way they take all important aspects of a way of working into account. When combining multiple frameworks the risk arises that some important aspects are forgotten. Therefore there should always be one framework at the basis on a way of working which could be expanded upon by other frameworks or approaches.

Bibliography

[1] M. Alqudah and R. Razali. A review of scaling agile methods in large software development. *International Journal on Advanced Science, Engineering and Information Technology*, 6(6):828, Dec 2016. doi: 10.18517/ijaseit.6.6.1374.

[2] S. Ambler and M. Lines. Intro to disciplined agile, Oct 2020. URL `https://www.pmi.org/disciplined-agile/introduction-to-disciplined-agile`. Accessed 03-11-2020.

[3] S. W. Ambler and M. Lines. *Disciplined Agile Delivery: a Practitioner's Guide to Agile Software Delivery in the Enterprise*. Ibm Press, Jun 2012. ISBN 9780132810135.

[4] S. W. Ambler and M. Lines. *Choose your WoW!: a disciplined agile delivery handbook for optimizing your way of working*. Project Management Institute, Inc, 2020. ISBN 9781628256505.

[5] D. J. Anderson, G. Concas, M. I. Lunesu, M. Marchesi, and H. Zhang. A comparative study of scrum and kanban approaches on a real case study using simulation. *Lecture Notes in Business Information Processing*, page 123–137, 2012. doi: 10.1007/978-3-642-30350-0_9.

[6] D. Badampudi, S. A. Fricker, and A. M. Moreno. Perspectives on productivity and delays in large-scale agile projects. *Lecture Notes in Business Information Processing*, page 180–194, 2013. doi: 10.1007/978-3-642-38314-4_13.

[7] D. Bartz and D. Shepardson. Pentagon hits reset on trump's $10 bln cloud deal, welcoming new players, Jul 2021. URL `https://www.reuters.com/technology/pentagon-scraps-jedi-award-microsoft-will-rebid-2021-07-06/`. Accessed 13-07-2021.

[8] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, and et al. Principles behind the agile manifesto, 2019. URL `https://agilemanifesto.org/principles.html`.

[9] E. Bjarnason, K. Wnuk, and B. Regnell. A case study on benefits and side-effects of agile practices in large-scale requirements engineering. *Proceedings of the 1st Workshop on Agile Requirements Engineering - AREW '11*, 2011. doi: 10.1145/2068783.2068786. URL https://dl.acm.org/citation.cfm?id=2068786.

[10] W. T. C. Bolhuis. The application of scalable agile in dutch government. In *Twentse Student Conference on IT*, volume 28. University of Twente, 2018. URL http://essay.utwente.nl/75918/1/Bolhuis_BA_EEMCS.pdf.

[11] W. Cats-Baril and R. Thompson. Managing information technology projects in the public sector. *Public Administration Review*, 55(6):559, Nov 1995. doi: 10.2307/3110347.

[12] K. Conboy and N. Carroll. Implementing large-scale agile frameworks: Challenges and recommendations. *IEEE Software*, 36(2):44–50, Mar 2019. doi: 10.1109/ms.2018.2884865. URL https://arxiv.org/ftp/arxiv/papers/1901/1901.08130.pdf.

[13] C. de O. Melo, D. S. Cruzes, F. Kon, and R. Conradi. Interpretative case studies on agile team productivity and management. *Information and Software Technology*, 55(2):412–427, Feb 2013. doi: 10.1016/j.infsof.2012.09.004.

[14] Dutch Tax and Customs Administration. Case study - dutch tax and customs administration, 2021. URL https://www.scaledagileframework.com/case-study-dutch-tax-and-customs-administration/. Accessed 15-03-2021.

[15] A. Dyck, R. Penners, and H. Lichter. Towards definitions for release engineering and devops. *2015 IEEE/ACM 3rd International Workshop on Release Engineering*, May 2015. doi: 10.1109/releng.2015.10.

[16] B. Fitzgerald, K.-J. Stol, R. O'Sullivan, and D. O'Brien. Scaling agile methods to regulated environments: An industry case study. *2013 35th International Conference on Software Engineering (ICSE)*, May 2013. doi: 10.1109/icse.2013.6606635.

[17] S. Gasik. Are public projects different than projects in other sectors? preliminary results of empirical research. *Procedia Computer Science*, 100:399–406, 2016. doi: 10.1016/j.procs.2016.09.175.

[18] S. Gasik. A framework for analysing differences between public-sector and other-sector projects. *Zarzadzanie Publiczne*, 45(3(45)/2018):73–88, 2018. doi: 10.15678/zp.2018.45.3.05.

[19] F. Gomes De Oliveira Neto, J. Horkoff, E. Knauss, R. Kasauli, and G. Liebel. Challenges of aligning requirements engineering and system testing in large-scale agile: A multiple case study, Sep 2017. URL https://ieeexplore.ieee.org/abstract/document/8054871.

[20] V. T. Heikkilä, M. Paasivaara, C. Lassenius, and C. Engblom. Continuous release planning in a large-scale scrum development organization at ericsson. *Lecture Notes in Business Information Processing*, page 195–209, 2013. doi: 10.1007/978-3-642-38314-4_14. URL https://link.springer.com/chapter/10.1007%2F978-3-642-38314-4_14.

[21] V. T. Heikkilä, M. Paasivaara, K. Rautiainen, C. Lassenius, T. Toivola, and J. Järvinen. Operational release planning in large-scale scrum with multiple stakeholders – a longitudinal case study at f-secure corporation. *Information and Software Technology*, 57:116–140, Jan 2015. doi: 10.1016/j.infsof.2014.09.005.

[22] M. Hering. Devops in scaled agile models – which one is best?, Feb 2015. URL https://notafactoryanymore.com/2015/02/10/devops-in-scaled-agile-models-which-one-is-best/. Accessed 08-01-2021.

[23] A. ICT-toetsing. Home — adviescollege ict-toetsing, 2020. URL https://www.adviescollegeicttoetsing.nl/.

[24] M. C. Jurisch, C. Ikas, P. Wolf, and H. Krcmar. Key differences of private and public sector business process change. *e-Service Journal*, 9(1):3–27, 2013. doi: 10.2979/eservicej.9.1.3.

[25] M. Kalenda, P. Hyna, and B. Rossi. Scaling agile in large organizations: Practices, challenges, and success factors. *Journal of Software: Evolution and Process*, 30(10):e1954, May 2018. doi: 10.1002/smr.1954.

[26] R. Kasauli, G. Liebel, E. Knauss, S. Gopakumar, and B. Kanagwa. Requirements engineering challenges in large-scale agile system development. *2017 IEEE 25th International Requirements Engineering Conference (RE)*, Sep 2017. doi: 10.1109/re.2017.60. URL https://ieeexplore.ieee.org/abstract/document/8049141/.

[27] R. Knaster. About - scaled agile framework, Oct 2020. URL https://www.scaledagileframework.com/about/. Accessed 03-11-2020.

[28] H. Kniberg and A. Ivarsson. *Scaling Agile @ Spotify with Tribes, Squads, Chapters & Guilds*. crisp.se, 2012. URL https://blog.crisp.se/wp-content/uploads/2012/11/SpotifyScaling.pdf.

[29] L. Lagerberg, T. Skude, P. Emanuelsson, K. Sandahl, and D. Stahl. The impact of agile principles and practices on large-scale software development projects: A multiple-case study of two projects at ericsson. *2013 ACM / IEEE International Symposium on Empirical Software Engineering and Measurement*, Oct 2013. doi: 10.1109/esem.2013.53. URL http://users.jyu.fi/~mieijala/kandimateriaali/The%20impact%20of%20agile%20principles%20and%20practices%20on%20largescale.pdf.

[30] C. Larman and B. Vodde. *Scaling lean & agile development: thinking and organizational tools for large-scale Scrum*. Upper Saddle River, Nj Addison-Wesley, 2009. ISBN 9780321480965.

[31] C. Larman and B. Vodde. *Large-Scale Scrum: more with LeSS*. Boston Addison-Wesley, 2017. ISBN 9780321985712.

[32] C. Larman and B. Vodde. Why less? - large scale scrum (less), 2020. URL https://less.works/less/framework/why-less. Accessed 04-11-2020.

[33] C. Larman and B. Vodde. Less huge - large scale scrum (less), 2020. URL https://less.works/less/less-huge/index. Accessed 04-11-2020.

[34] D. Leffingwell. *Scaling software agility: best practices for large enterprises*. Addison-Wesley, 2007. ISBN 9780321458193.

[35] D. Leffingwell. Implementation - scaled agile framework, Feb 2021. URL https://www.scaledagileframework.com/implementation-roadmap/.

[36] D. Leffingwell and D. G. Reinertsen. *Agile software requirements: lean requirements practices for teams, programs, and the enterprise*. Addison-Wesley, , Cop, 2012. ISBN 9780321635846.

[37] R. Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 22(140):5–55, 1932. URL ATechniquefortheMeasurementofAttitudes.

[38] D. Lynn. Less and devops, Sep 2017. URL https://www.agile42.com/en/blog/2017/09/22/less-and-devops/. Accessed 06-01-2021.

[39] J. Manning. In vivo coding. *The International Encyclopedia of Communication Research Methods*, page 1–2, Nov 2017. doi: 10.1002/9781118901731.iecrm0270.

[40] M. B. Miles and M. A. Huberman. *Qualitative data analysis : an expanded sourcebook*. Sage Publ., 20[09, 1994. ISBN 9780803946538.

[41] Ministerie van Binnenlandse Zaken en Koninkrijksrelaties. Rijks ict dashboard, 2020. URL `https://www.rijksictdashboard.nl`. Accessed 20-07-2021.

[42] M. Paasivaara. Adopting safe to scale agile in a globally distributed organization. *2017 IEEE 12th International Conference on Global Software Engineering (ICGSE)*, May 2017. doi: 10.1109/icgse.2017.15.

[43] M. Paasivaara and C. Lassenius. Scaling scrum in a large globally distributed organization: A case study. *2016 IEEE 11th International Conference on Global Software Engineering (ICGSE)*, Aug 2016. doi: 10.1109/icgse.2016.34.

[44] M. Paasivaara, S. Durasiewicz, and C. Lassenius. Distributed agile development: Using scrum in a large project. *2008 IEEE International Conference on Global Software Engineering*, Aug 2008. doi: 10.1109/icgse.2008.38.

[45] M. Paasivaara, C. Lassenius, and V. T. Heikkilä. Inter-team coordination in large-scale globally distributed scrum. *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement - ESEM '12*, 2012. doi: 10.1145/2372251.2372294.

[46] M. Paasivaara, B. Behm, C. Lassenius, and M. Hallikainen. Towards rapid releases in large-scale xaas development at ericsson: A case study. *2014 IEEE 9th International Conference on Global Software Engineering*, Aug 2014. doi: 10.1109/icgse.2014.22.

[47] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee. A design science research methodology for information systems research. *Journal of Management Information Systems*, 24(3):45–77, Dec 2007. doi: 10.2753/mis0742-1222240302. URL `https://dl.acm.org/citation.cfm?id=1481768`.

[48] J. L. Perry and H. G. Rainey. The public-private distinction in organization theory: A critique and research strategy. *The Academy of Management Review*, 13(2):182, Apr 1988. doi: 10.2307/258571.

[49] Project Management Institute. *A guide to the project management body of knowledge : (PMBOK® guide)*. Project Management Institute, 6 edition, 2017. ISBN 9781628251845.

[50] Project Management Institute. Disciplined devops – disciplined agile (da), Jan 2021. URL `https://www.pmi.org/disciplined-agile/process/disciplined-devops`. Accessed 07-01-2021.

[51] Project Management Institute. Defining devops, Jan 2021. URL https://www.pmi.org/disciplined-agile/process/disciplined-devops/defining-devops. Accessed 07-01-2021.

[52] Project Management Institute. Devops strategies throughout disciplined agile, Jan 2021. URL https://www.pmi.org/disciplined-agile/process/disciplined-devops/devops-strategies-throughout-disciplined-agile. Accessed 07-01-2021.

[53] Project Management Institute. Critical success factors, Jan 2021. URL https://www.pmi.org/disciplined-agile/process/disciplined-devops/critical-success-factors. Accessed 07-01-2021.

[54] A. Putta, M. Paasivaara, and C. Lassenius. Benefits and challenges of adopting the scaled agile framework (safe): Preliminary results from a multivocal literature review. *Product-Focused Software Process Improvement*, 19:334–351, 2018. doi: 10.1007/978-3-030-03673-7_24.

[55] D. G. Reinertsen. *The principles of product development flow : second generation lean product development*. Celeritas, 2009. ISBN 9781935401001.

[56] K. M. Rosacker and R. E. Rosacker. Information technology project management within public sector organizations. *Journal of Enterprise Information Management*, 23(5):587–594, Sep 2010. doi: 10.1108/17410391011083047.

[57] W. S. Sayre. Premises of public administration: Past and emerging. *Public Administration Review*, 18(2):102, 1958. doi: 10.2307/973789.

[58] Scaled Agile Inc. Achieving business agility with safe 5.0, 2019. URL https://www.scaledagile.com/resources/safe-whitepaper/. Accessed 03-11-2020.

[59] Scaled Agile Inc. Devops - scaled agile framework, Nov 2020. URL https://www.scaledagileframework.com/devops/. Accessed 07-01-2021.

[60] K. Schwaber. *Nexus TM Guide The Definitive Guide to scaling Scrum with Nexus: The Rules of the Game*. Scrum.org, 2018. URL https://scrumorg-website-prod.s3.amazonaws.com/drupal/2018-01/2018-Nexus-Guide-English_0.pdf?nexus-file=https%3A%2F%2Fscrumorg-website-prod.s3.amazonaws.com%2Fdrupal%2F2018-01%2F2018-Nexus-Guide-English_0.pdf.

[61] K. Schwaber and J. Sutherland. *The Scrum GuideTM The Definitive Guide to Scrum: The Rules of the Game*. Scrum.org, 2017. URL https:

//www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf#zoom=100.

[62] P. B. Seddon and R. Scheepers. Towards the improved treatment of generalization of knowledge claims in is research: Drawing general conclusions from samples. *European Journal of Information Systems*, 21(1):6–21, Jan 2012. doi: 10.1057/ejis.2011.9.

[63] M. Shameem, C. Kumar, B. Chandra, and A. A. Khan. Systematic review of success factors for scaling agile methods in global software development environment: A client-vendor perspective. *2017 24th Asia-Pacific Software Engineering Conference Workshops (APSECW)*, 24, Dec 2017. doi: 10.1109/apsecw.2017.22.

[64] M. Silva. Project management bricolage: the whatever works way, Nov 2018. URL https://wellingtone.co.uk/project-management-bricolage-whatever-works-way/. Accessed 26-07-2021.

[65] State of Agile. *14th Annual State of Agile Report*. Digital.ai, May 2020. URL https://stateofagile.com/#ufh-i-615706098-14th-annual-state-of-agile-report/7027494.

[66] I. Stojanov, O. Turetken, and J. J. Trienekens. A maturity model for scaling agile development. *2015 41st Euromicro Conference on Software Engineering and Advanced Applications*, Aug 2015. doi: 10.1109/seaa.2015.29.

[67] J. Sutherland. Scrum@scale guide, 2006. URL https://www.scrumatscale.com/scrum-at-scale-guide-online/. Accessed 24-06-2021.

[68] K. Thompson. *Recipes for Agile Governance in the Enterprise THE ENTERPRISE WEB*. cPrime Inc, 2013. URL https://www.cprime.com/wp-content/uploads/woocommerce_uploads/2013/07/RAGE-Final-cPrime1.pdf.

[69] A. van Ast. Safe werken: wendbaar en resultaatgericht, 2021. URL https://www.werkenvoornederland.nl/organisaties/ministerie-van-binnenlandse-zaken-en-koninkrijksrelaties/logius/safe-werken-wendbaar-en-resultaatgericht. Accessed 15-03-2021.

[70] J. van Hillegersberg, G. Ligtenberg, and M. N. Aydin. Getting agile methods to work for cordys global software product development. In *International Workshop on Global Sourcing of Information Technology and Business Processes*, page 133–152. Springer, 2011.

[71] M. A. Waheed, S. Muhammed, I. Sherjeel, A. Muhammed, and K. Adnan. A review of popular agile software development technologies. *Journal of Information Technology & Software Engineering*, 08(04), 2018. doi: 10.4172/2165-7866.1000245.

Appendix A

Search terms

Research question	Search terms	Total papers found	Initial papers selected	Final papers used	Other sources used
1.	”Large Scale Agile”	67	8 (12%)	0 (0%)	8
2.	”Large Scale Agile AND Challenges”	29	6 (21%)	3 (50%)	3
	”Large Scrum AND Challenges”	10	3 (30%)	2 (67%)	
	”Disciplined Agile Delivery AND Challenges”	1	0	0	
	”Large Scale Agile AND Benefits”	11	3 (27%)	1 (33%)	
	”Large Scrum AND Benefits”	3	0	0	
	”Large Scale Agile”	69	8 (12%)	2 (25%)	
3.	”Large Scale Agile AND Application OR Case Study”	23	3 (13%)	2 (67%)	1
	”Large Scrum AND Application OR Case Study”	72	13 (18%)	3 (23%)	
	”Scaled Agile Framework AND Application OR Case Study”	5	1 (20%)	1 (100%)	
	”Disciplined Agile Delivery AND Application OR Case Study”	0	0	0	
	”Large Scale Agile”	69	9 (13%)	3 (33%)	
4.	”Large Scale Agile AND DevOps”	1	0 (0%)	0 (0%)	8
	”Large Scrum AND DevOps”	0	0 (0%)	0 (0%)	
	”Scaled Agile Framework AND DevOps”	0	0 (0%)	0 (0%)	
	”Large Scale DevOps”	0	0 (0%)	0 (0%)	
5.	”Compare Projects AND Public Sector AND Private Sector”	0	0 (0%)	0 (0%)	7
	”Compare AND Public Sector AND Private Sector”	3	0 (0%)	0 (0%)	
6.	”Large Scale Agile AND Application OR Case Study”	23	13 (57%)	11 (85%)	1
	”Large Scrum AND Application OR Case Study”	72	9 (13%)	8 (89%)	
	”Scaled Agile Framework AND Application OR Case Study”	5	2 (40%)	1 (50%)	
	”Large Scale Agile”	69	10 (14%)	8 (80%)	

Table A.1: Search terms used and usage of papers

Appendix B

Papers used

Research Question	Papers used
1.	[1, 2, 3, 4, 8, 27, 28, 30, 31, 32, 33, 34, 36, 58, 60, 68, 71]
2.	[12, 25, 54, 63, 65, 66, 70]
3.	[29, 42, 43, 44, 70]
4.	[15, 22, 38, 50, 51, 52, 53, 59]
5.	[11, 17, 18, 24, 48, 49, 56, 57]
6.	[5, 6, 9, 13, 16, 19, 20, 21, 26, 29, 42, 43, 45, 46, 66, 70]

Table B.1: Papers utilised to answer each research question

Appendix C

Glossary

Solution The overall final product that is being worked towards.

Product A (unique) part of the Solution, with its own release of functional components at the end of a Iteration.

Development Team A (agile) team designing, building, testing, and releasing (parts of) a Product.

Product Owner (PO) Person with the final responsibility for the Backlog and the requirements of a Product.

Solution Owner (SO) Person with the final responsibility for the Backlog and the requirements of the Solution.

Area Product Owner (APO) Between a SO and a PO, overarching for multiple Products, but not the Solution.

Scrum Master Supporter of one or multiple Development Teams, they ensure the agile process is being followed.

Story A short description of requested functionality or a feature.

Solution Backlog The list of all Stories for the entire Solution.

Product Backlog The list of all Stories for each Product.

Team Backlog The list of all Stories a Development Team is supposed to work on.

Iteration A fixed-length period of team during which Stories from a Backlog are developed.

Stand-up A, usually daily, meeting during which the previous day and the coming day are discussed.

Demo At the end of each Iteration the Team presents their work and shows what it can do in a Demo.

Review A meeting at the end of an Iteration during which the progress made is reflected upon.

Retrospective A meeting in which is reflected on the process of the Iteration.

Definition of Done A agreed upon list of criteria the final Product or Solution will have to meet.

Increment A fixed-length period of time spanning multiple Iterations, with similar goals on a larger scale.

Scrum of Scrums (SoS) A Stand-up, but the meeting is held between multiple Teams instead of individuals.

Scrum of Scrums master (SOSm) A upscaled version of the Scrum Master

Chief Product Owner (CPO) A upscaled version of the Product Owner

Appendix D

Survey questions

Below the full list of survey statements can be found. These are translated versions of the original dutch statements. The translations are not literal translations but translations which most closely match the intended direction of the survey questions.

1. The communication and knowledge sharing with my direct colleagues and supervisor is good.
2. I have enough knowledge of (large scale) agile.
3. The technical environment (tools and infrastructure) which support my work are good.
4. There is sufficient attention to quality assurance of the final product I am working on.
5. I have enough autonomy in the work I do.
6. I am motivated in my work.
7. The current way of working in my organisation is good.
8. The support for large scale agile working from the different management layers is good.
9. My colleagues understand me when I talk about the work I do.
10. My stress levels are low.
11. I think (large scale) agile is fitting to my organisation.
12. I think (large scale) agile is fitting to the work I do.
13. The communication and knowledge sharing with parties I do not work with on a daily basis is good.
14. I feel involved in the current way of working of my organisation.
15. I do my working according to the agile principles.
16. I am aware of the work my direct colleagues are doing.
17. I think my organisation made a well-informed decision for the current structure of the organisation.

Appendix E

Codebook

E.1 Codebook demographics

Code	Code Group
# case: 1	Demographics
# case: 2	Demographics
# case: 3	Demographics
# case: 4	Demographics
# role: Agile Process Coach	Demographics
# role: Developer	Demographics
# role: Manager	Demographics
# role: Product Owner	Demographics
# role: Stakeholder	Demographics
#interview 01	Demographics
#interview 02	Demographics
#interview 03	Demographics
#interview 04	Demographics
#interview 05	Demographics
#interview 06	Demographics
#interview 07	Demographics
#interview 08	Demographics
#interview 09	Demographics
#interview 10	Demographics
#interview 11	Demographics
#interview 12	Demographics
#interview 13	Demographics
#interview 14	Demographics
#interview 15	Demographics
#interview 16	Demographics
#interview 17	Demographics
#interview 18	Demographics
#interview 19	Demographics
#interview 20	Demographics
#interview 21	Demographics
#interview 22	Demographics

Table E.1: Codebook - Demographics

E.2 Codebook Success Factors

Code	Code Group
COMMUNICATION SF1	Success Factors
Communication: Communication, coordination, control SF1.1	Success Factors
Communication: Close & constant communications SF1.2	Success Factors
Communication: Customer involvement SF1.3	Success Factors
Communication: Encourage visibility SF1.4	Success Factors
KNOWLEDGE SF2	Success Factors
Knowledge: Train & coach people SF2.1	Success Factors
Knowledge: Acquire knowledge SF2.2	Success Factors
Knowledge: Knowledge sharing SF2.3	Success Factors
INFRASTRUCTURE SF3	Success Factors
Infrastructure: Tools & technology SF3.1	Success Factors
Infrastructure: Requirements analysis SF3.2	Success Factors
PRACTICES SF4	Success Factors
Practices: Engineering practices =>less quality reduction SF4.1	Success Factors
Practices: Short iterations SF4.2	Success Factors
TEAMS SF5	Success Factors
Teams: Small team size SF5.1	Success Factors
Teams: Self-organising teams SF5.2	Success Factors
Teams: Experienced & motivated members SF5.3	Success Factors
Teams: Conduct social events SF5.4	Success Factors
TRANSITIONING SF6	Success Factors
Transitioning: Slowly & carefully SF6.1	Success Factors
Transitioning: Executive sponsorship SF6.2	Success Factors
MANAGEMENT SUPPORT SF7	Success Factors
Management support: Management commitment SF7.1	Success Factors
Management support: Effective leadership SF7.2	Success Factors
UNIFICATION SF8	Success Factors
Unification: Define common view SF8.1	Success Factors

Table E.2: Codebook - Success Factors

E.3 Codebook Challenges

Code	Code Group
TEAM ISSUES C1	Challenges
Team issues: Minimal collaboration & knowledge sharing C1.1	Challenges
Team issues: Lacking autonomy C1.2	Challenges
Team issues: Lack of trust C1.3	Challenges
Team issues: Team size C1.4	Challenges
Team issues: Increased stress & pressure C1.5	Challenges
ORGANISATIONAL C2	Challenges
Organisational: Scaling to non-IT C2.1	Challenges
Organisational: Mismatch organisation & agile C2.2	Challenges
Organisational: Mismatch client & agile C2.3	Challenges
Organisational: Mismatch regulations & agile C2.4	Challenges
GLOBALISATION C3	Challenges
Globalisation: Physical distance C3.1	Challenges
Globalisation: Synchronised communication C3.2	Challenges
Globalisation: Multiple communication channels C3.3	Challenges
Globalisation: Scaling to distributed organisation C3.4	Challenges
TRANSITION ISSUES C4	Challenges
Transition issues: Measuring results C4.1	Challenges
Transition issues: Adopting agile mindset C4.2	Challenges
Transition issues: Organisation readiness C4.3	Challenges
Transition issues: Initiating top-down & bottom-up C4.4	Challenges
Transition issues: Moving from fixed to iterative delivery C4.5	Challenges
RESISTANCE C5	Challenges
Resistance: Resistance towards change C5.1	Challenges
Resistance: Resistance towards roles C5.2	Challenges
Resistance: Consequences of transparency C5.3	Challenges
FRAMEWORK C6	Challenges
Framework: Inconsistent terminology C6.1	Challenges
Framework: Scaling doesn't feel agile C6.2	Challenges
Framework: Framework contradictions C6.3	Challenges
INFORMATION C7	Challenges
Information: Lack of training C7.1	Challenges
Information: Lack of skills or experience C7.2	Challenges
Information: Little available literature C7.3	Challenges
Information: Lack of comparison C7.4	Challenges
PRACTICAL C8	Challenges
Practical: Issues with first iteration C8.1	Challenges
Practical: Issues with integration C8.2	Challenges
Practical: Automating testing C8.3	Challenges
Practical: Scaling requirements management C8.4	Challenges
MANAGEMENT ISSUES C9	Challenges
Management issues: Lacking leadership participation C9.1	Challenges
Management issues: Lacking management support & sponsorship C9.2	Challenges
QUALITY C10	Challenges
Quality: Problems with assurance C10.1	Challenges
Quality: Quality loss C10.2	Challenges

Table E.3: Codebook - Challenges

E.4 Codebook Additions

Code	Code Group
***NEG**	Additional
*neg: Agile does not fit government	Additional
*neg: Distance to user	Additional
*neg: Focus on money, not on customer value	Additional
*neg: Lack of time	Additional
*neg: Perverse financial stimulations	Additional
*neg: Through agile replacement of an existing product is more challenging	Additional
*neg: Using method because other teams do so as well	Additional
***NOTE**	Additional
*note: Change always occurs in a project	Additional
*note: Cultural change needed	Additional
*note: Don't follow agile to the letter	Additional
*note: Don't use agile for everything	Additional
*note: Entire industry uses Agile, you need to fit in	Additional
*note: Let Agile fit to the working method of the rest of the organisation	Additional
*note: Management give direction, leave decisions & responsibility lower in organisation	Additional
*note: Waterfall didn't really give you the product you really wanted	Additional
*note: Waterfall does not allow steering during the project	Additional
*note: Waterfall offers false security	Additional
*note: Way of working is secondary to achieving your goals	Additional
*note: You always split work in smaller packages	Additional
***POS**	Additional
*pos: Agile allows constant steering	Additional
*pos: Agile brought everyone together again	Additional
*pos: Agile fits government	Additional
*pos: Agile has less overrun	Additional
*pos: Agile is better than Traditional	Additional
*pos: Agile is cheaper	Additional
*pos: Agile is transparent	Additional
*pos: Agile produces subproducts	Additional
*pos: Agile works well	Additional
*pos: Less stress compared to other projects	Additional
*pos: Low stress	Additional

Table E.4: Codebook - Additions

Appendix F

Survey results

F.1 Full survey results

C	Role	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17
	AVG	4,14	3,86	3,81	3,86	4,38	4,57	3,00	3,00	4,15	3,67	3,81	4,48	3,10	4,05	3,70	4,19	3,33
1	APC	4	3	4	5	5	5	2	4	5	5	5	5	2	4	5	5	5
1	APC	5	5	4	4	5	5	3	5	4	2	5	5	4	5	4	5	3
1	PO	3	4	5	4	5	4	4	3	3	3	4	5	3	4	4	3	4
2	APC	5	4	3	3	3	4	2	3	4	4	2	4	3	4	4	5	3
2	DEV	4	4	5	5	5	5	4	4	5	5	4	5	2	5	3	3	3
2	DEV	5	4	5	5	5	5	4	1	5	2	3	5	3	5	4	5	4
2	MGR	4	3	4	4	5	5	3	3	4	3	4	4	3	3	3	4	
2	MGR	4	4	4	5	5	5	3	1	4	2	4	4	3	3	4	4	2
2	PO	2	3	2	2	3	4	2	2	4	4	2	4	3	4	3	3	3
2	SH	4	3	2	4	3	3	4	3	4	3	4	4	3	3	3	4	
2	SH	5	5	4	5	5	4	3	4		4	4	5	5	4		4	5
3	APC	4	4	3	4	4	4	2	4	4	4	4	5	3	3	4	4	
3	DEV	5	5	5	4	4	5	3	4	4	4	3	3	4	4	4	5	3
3	DEV	4	4	4	1	5	5	2	2	4	4	4	4	2	4	2	5	2
3	DEV	3	4	3	3	4	4	3	2	4	4	4	4	2	4	3	3	3
3	DEV	5	2	4	4	5	5	4	3	4	4	3	5	4	5	4	5	4
3	MGR	4	4	4	4	4	5	4	4	4	3	4	4	4	4	4	4	3
3	MGR	4	4	5	2	4	5	4	4	4	4	4	5	4	5	3	4	4
3	PO	4	3	4	4	4	5	3	3	4	4	4	4	3	4	4	4	3
3	SH																	
4	APC	5	5	4	5	5	5	3	3	5	5	4	5	3	4	5	5	3
4	PO	4	4	2	4	4	4	1	1	4	4	5	5	2	4	4	4	3

Table F.1: Anonymised survey results including average results.

Used abbreviations:

APC Agile Process Coach

DEV Developer

MGR Manager

PO Product Owner

SH Stakeholder

F.2 Survey question correlation

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17
Q1	X	0,35	0,36	0,42	0,33	0,35	0,14	0,28	0,37	-0,07	0,04	0,16	0,43	0,27	0,42	0,79	0,16
Q2	0,35	X	0,28	0,15	0,18	0,08	-0,07	0,22	0,08	-0,07	0,19	0,04	0,25	0,11	0,17	0,13	-0,15
Q3	0,36	0,28	X	0,25	0,66	0,71	0,57	0,31	0,17	-0,07	0,08	0,16	0,27	0,47	0,13	0,19	0,25
Q4	0,42	0,15	0,25	X	0,44	0,13	0,25	0,12	0,45	-0,10	0,30	0,33	0,12	-0,06	0,70	0,09	0,35
Q5	0,33	0,18	0,66	0,44	X	0,61	0,23	0,06	0,28	-0,10	0,46	0,47	0,02	0,26	0,20	0,22	0,23
Q6	0,35	0,08	0,71	0,13	0,61	X	0,19	0,15	0,40	0,00	0,13	0,04	0,09	0,43	0,19	0,41	-0,14
Q7	0,14	-0,07	0,57	0,25	0,23	0,19	X	0,25	0,00	-0,31	-0,07	0,09	0,40	0,33	-0,08	-0,15	0,23
Q8	0,28	0,22	0,31	0,12	0,06	0,15	0,25	X	0,01	0,24	0,22	0,15	0,47	0,20	0,17	0,12	0,37
Q9	0,37	0,08	0,17	0,45	0,28	0,40	0,00	0,01	X	0,35	0,08	0,29	-0,30	0,29	0,28	0,34	0,20
Q10	-0,07	-0,07	-0,07	-0,10	-0,10	0,00	-0,31	0,24	0,35	X	-0,02	0,12	-0,29	0,11	0,07	-0,05	0,19
Q11	0,04	0,19	0,08	0,30	0,46	0,13	-0,07	0,22	0,08	-0,02	X	0,40	-0,19	-0,07	0,16	-0,02	0,11
Q12	0,16	0,04	0,16	0,33	0,47	0,04	0,09	0,15	0,29	0,12	0,40	X	0,00	0,44	0,32	0,01	0,57
Q13	0,43	0,25	0,27	0,12	0,02	0,09	0,40	0,47	-0,30	-0,29	-0,19	0,00	X	0,17	0,20	0,21	0,34
Q14	0,27	0,11	0,47	-0,06	0,26	0,43	0,33	0,20	0,29	0,11	-0,07	0,44	0,17	X	0,03	0,18	0,34
Q15	0,42	0,17	0,13	0,70	0,20	0,19	-0,08	0,17	0,28	0,07	0,16	0,32	0,20	0,03	X	0,39	0,44
Q16	0,79	0,13	0,19	0,09	0,22	0,41	-0,15	0,12	0,34	-0,05	-0,02	0,01	0,21	0,18	0,39	X	0,06
Q17	0,16	-0,15	0,25	0,35	0,23	-0,14	0,23	0,37	0,20	0,19	0,11	0,57	0,34	0,34	0,44	0,06	X

Table F.2: Correlation between the 17 questions.

F.3 Survey averages

		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Correlation
GENERAL		4,14	3,86	3,81	3,86	4,38	4,57	3,00	3,00	4,15	3,67	3,81	4,48	3,10	4,05	3,70	4,19	3,33	X
CASES	#																		
1	3	4,00	4,00	4,33	4,33	5,00	4,67	3,00	4,00	4,00	3,33	4,67	5,00	3,00	4,33	4,33	4,33	4,00	0,76
2	8	4,13	3,75	3,63	4,13	4,25	4,38	3,13	2,63	4,29	3,38	3,38	4,38	3,13	3,88	3,43	4,00	3,33	0,93
3	9	4,13	3,75	4,00	3,25	4,25	4,75	3,13	3,25	4,00	3,88	3,75	4,25	3,25	4,13	3,50	4,25	3,14	0,90
4	2	4,50	4,50	3,00	4,50	4,50	4,50	2,00	2,00	4,50	4,50	4,50	5,00	2,50	4,00	4,50	4,50	3,00	0,85
ROLES	#																		
APC	5	4,60	4,20	3,60	4,20	4,40	4,60	2,40	3,80	4,40	4,00	4,00	4,80	3,00	4,00	4,40	4,80	3,50	0,84
DEV	6	4,33	3,83	4,33	3,67	4,67	4,83	3,33	2,67	4,33	3,83	3,50	4,33	2,83	4,50	3,33	4,33	3,17	0,91
MGR	4	4,00	3,75	4,25	3,75	4,50	5,00	3,50	3,00	4,00	3,00	4,00	4,25	3,50	3,75	3,50	4,00	3,00	0,81
PO	4	3,25	3,50	3,25	3,50	4,00	4,25	2,50	2,25	3,75	3,75	3,75	4,50	2,75	4,00	3,75	3,50	3,25	0,87
SH	2	4,50	4,00	3,00	4,50	4,00	3,50	3,50	3,50	4,00	3,50	4,00	4,50	4,00	3,50	3,00	4,00	5,00	0,13

Table F.3: Average survey scores.

Correlation coefficient between general average and specific averages.

Appendix G

Success factor & challenge occurrences in case study

	Case1	Case2	Case3	Case4
SF1.1		X	X	X
SF1.2	X	X	X	X
SF1.3	X	X	X	X
SF1.4			X	X
SF2.1	X	X	X	X
SF2.2		X	X	
SF2.3	X	X	X	X
SF3.1		X	X	
SF3.2		X		X
SF4.1		X		
SF4.2	X	X	X	X
SF5.1	X	X	X	X
SF5.2	X	X	X	
SF5.3		X	X	
SF5.4			X	X
SF6.1	X	X	X	
SF6.2		X		
SF7.1				
SF7.2			X	X
SF8.1		X	X	X

	Case1	Case2	Case3	Case4
C1.1	X	X	X	
C1.2	X	X	X	X
C1.3	X	X	X	X
C1.4				
C1.5		X		
C2.1		X	X	
C2.2	X	X	X	
C2.3	X	X		
C2.4				
C3.1			X	
C3.2	X	X	X	
C3.3				
C3.4			X	
C4.1		X		X
C4.2	X	X	X	
C4.3		X	X	
C4.4		X		X
C4.5	X	X		
C5.1	X	X	X	X
C5.2				
C5.3				
C6.1		X		
C6.2			X	
C6.3		X		
C7.1		X	X	X
C7.2		X		X
C7.3				
C7.4		X		
C8.1				
C8.2				
C8.3	X			
C8.4	X			
C9.1	X	X	X	X
C9.2		X	X	X
C10.1		X	X	
C10.2				

Table G.1: Occurrence of success factors and challenges in the cases.