

Graduation Thesis

“Exploring the Organizational Context around Agile Software Delivery”



Graduation thesis

*Exploring the organizational context
around agile software delivery*

University Delft University of Technology
Faculty Faculty of Civil Engineering (CitG)
Master Construction Management and Engineering (CME)



Author

Name R.G.D. (Roeland) Rustema
Student number 1177575
E-mail roelandrustema@hotmail.com

Graduation committee

Chairman Prof. dr. H.L.M. (Hans) Bakker
 Faculty of Civil Engineering and Geosciences (CitG)

Supervisor Dr. ir. M.G.C. (Marian) Bosch-Rekveldt
 Faculty of Civil Engineering and Geosciences (CitG)

Supervisor Dr. ir. B.M. (Bauke) Steenhuisen
 Faculty of Technology, Policy and Management (TPM)

Supervisor external Drs. M.P.C. (Martijn) van Nierop
 KWD Resultaatmanagement (KWDRM)

Commissioned by

Company KWD Resultaatmanagement
 Nieuwegein



This thesis is defended publicly on August 28, 2018 at 3:00 PM.

An electronic version of this thesis is available at <http://repository.tudelft.nl/>.

*“If you, or your organization, do not adapt to changed circumstances,
you will cease to exist”*

- General Tom Middendorp,
(Chief of Defence Dutch Armed Forces 2012-2017)

PREFACE

This report is the result of an intensive and interesting period as a graduation student in the world of agile software delivery. Conducting this research not only introduced me to agile management, but also offered me a unique chance to learn more about software development and gave me the chance to gain an insight in several interesting organizations and in that way enriched my knowledge as a manager in general.

Before starting this Master study I've worked for several years as a military officer and in this period gained some management experience. The management style that is commonly described as the 'Waterfall method' suited me well, being a person that prefers to stay in control and favours a certain level of predictability. Therefore, conducting a research into agile management offered me the opportunity to learn about a different management style, but also to reflect on my own experience as a manager.

I feel privileged with the possibility to have graduated at KWD Resultaatmanagement. The "KWD-ers" welcomed me with open arms and gave me a look inside in a unique company with highly professional and motivated people. The sincere interest that was shown and the help that was offered gave me an extra stimulation during the graduation process.

Also I'm very grateful to all the agile professionals that took the time to contribute to this research. Conducting the interviews was a part of the research that I much enjoyed. The cooperation, enthusiasm and openness of the interviewees was inspiring and their detailed information made this research as it is.

Above all, without the guidance, feedback and help from my graduation committee I would not have succeeded. I want to thank all of them for always finding time for me despite their enormous workload. I'm thankful to Hans for introducing me at KWD. Although we've met only a few times during the graduation process, I've always felt the trust and support that I needed to keep going. I want to thank Bauke for being part of my committee. I got to know Bauke as another type of academic teacher than that I was used to at CME. Although I sometimes left his office with my head spinning of all the feedback and the fierce discussions, I really appreciate the perspective that he offered on my work. Martijn, thank you for your support and dedication. I'm happy to have such an experienced and knowledgeable company supervisor. Martijn warned me in the beginning about be very critical, which I can happily confirm in hindsight. Many thanks to Marian for being my first supervisor. The way she gave guidance to me, and feedback on my work, fitted me really well as a person. She pushed me to keep the momentum in the graduating process, without making concessions to the quality of work.

I'm grateful to have family, friends and fellow students around to show interest in what I was doing and, above all, offer me the much-needed distraction. Lastly, I want to thank my girlfriend to share with me her recent experience as a graduation student and helping me to stay motivated at times.

Roeland Rustema
Amsterdam, 17 August 2018

SUMMARY

Introduction

Agile management has revealed itself as a management approach that copes with an unclear product scope and fast-changing circumstances. This approach has gained popularity by projects in fast changing environments, such as the information technology (IT) industry. Nevertheless, many companies that adopted agile methods are still structured according to a traditional, non-agile form of organization. Projects with an agile project management approach embedded in a non-agile organization might face numerous difficulties. What these difficulties exactly encompass is not fully understood yet and therefore this research strives to fulfil the following objective: *“Explore the interaction between agile project management and its organizational context.”*

In this way this research aims to contribute to literature about the implementation of agile within an organization. Next to that, this research provides insight to organizations about how an organization is best aligned with agile. This research uses the term *software delivery*, since this includes projects as well as on-going activities. This results in the following research question:

“What kind of adjustments can an organization make to better facilitate agile software delivery?”

Research Approach












This research can be described as an explorative research, in which a qualitative research approach is chosen to find an answer to the research question. A literature study gives a better understanding of the research subject and results in the identification of aspects that are relevant to examine the interaction between agile software delivery and the organizational context.

The core of the research is a multiple-case study. Each case is formed by an organization practicing agile software delivery. In total 8 cases are selected with 2 participants per case. Each participant is subjected to a semi-structured interview that is designed based on the outcome of the literature study.

The interview results give input to the cross-case analysis and in that way show a comprehensive presentation of the interaction of agile software delivery and its organization in practice. The patterns that are derived from the cross-case analysis are further interpreted to finally give an answer to the research question.

Results and Conclusion

The literature study resulted in a compilation of the following eleven aspects that are relevant to examine the interaction between agile software delivery and the organizational context.

1	Documentation		5	Budgeting and cost accounting		9	Staffing plan	
2	Decision-making		6	Mutual adjustment		10	Management style	
3	Planning and control systems		7	Division of tasks and responsibilities		11	Organizational culture	
4	Resource management		8	Performance and reward systems				

These aspects are used as guidance during the interviews. The interaction of agile with the organizational context is discussed with the interviewees in relation to these eleven aspects, in that way creating an image of the alignment of agile within the organization.

Although each case is unique in its own way, multiple commonalities are observed over the different cases. Interpretation of the results of the cross-case analysis resulted in three patterns.

The first pattern shows how organizations tend to focus on team level when implementing agile and have the tendency to *neglect the organizational adjustments around teams*. The organizational surrounding is in this research described as the governance structure around the teams and includes the division of tasks, responsibilities and other governance mechanisms. Several of the interview observations can be explained by an insufficient adjustment of the governance structure.

A second pattern is to what extent *agile is understood* and how it is interpreted. Some of the interview observations can be explained by an insufficient understanding of agile. Adjustment of the governance structure around the software delivery teams should be aligned with agile concepts.

The third pattern shows that several observations can be explained by the fact that *change needs time*. Every organization undertakes a transition when implementing agile software delivery to change the governance structure and to understand agile. Alignment of agile within the organization depends on the stage an organization is in during this transition.

Based on these patterns, this research **concludes** that an organization could consider *adjusting* its *governance structure* to better facilitate agile software delivery. When making these adjustments, a sufficient *understanding of agile* is required to ensure that adjustments of the governance structure are aligned with agile software delivery. Next to that, the implementation of agile and the adjustments of the governance structure can be considered as a *transition* that needs to be *managed* pro-actively.

This research suggests a few recommendations for future research:

- Extend the research field;
- Including of other perspectives;
- Comparative research into the governance structure;
- Further investigate the implementation of agile;
- Investigate how adjustments affect the success of agile software delivery;
- Investigate certain elements of the governance structure in relation to agile.

Recommendations for practice can roughly be divided in two. Firstly, organizations should reconsider the governance structure around the agile teams. A second recommendation is to actively manage the implementation of agile, with active involvement of higher management.

TABLE OF CONTENT

SUMMARY	vi
TABLE OF CONTENT	viii
LIST OF FIGURES	xi
LIST OF TABLES	xii
1. INTRODUCTION	1
1.1 Background: Agile management and the organizational context	1
1.2 Fitting agile management within an organization	1
1.3 KWD Resultaatmanagement	2
1.4 Reading guide	3
2. RESEARCH DESIGN	5
2.1 Research objective	5
2.2 Scoping the research	5
2.2.1 <i>Software engineering</i>	5
2.2.2 <i>Delivery instead of project</i>	6
2.2.3 <i>Focus on organization around delivery</i>	6
2.2.4 <i>Scoping the result</i>	6
2.3 Research questions.....	7
2.4 Methodology	8
3. THEORETICAL FRAMEWORK	11
3.1 Agile software delivery.....	11
3.1.1 <i>Introduction to software delivery</i>	11
3.1.2 <i>Managing software delivery</i>	13
3.1.3 <i>Defining the level of agility</i>	19
3.1.4 <i>Organization around software delivery</i>	20
3.2 Interaction between agile software delivery and the organizational context.....	25
3.2.1 <i>Method of gathering aspects</i>	25
3.2.2 <i>Gathering of aspects describing the interaction</i>	26
3.2.3 <i>Compilation of aspects</i>	31
3.3 Theoretical proposition	32
4. CASE STUDY APPROACH	33
4.1 Case selection	33
4.2 Case study protocol.....	35
4.3 Data collection	35

5. SINGLE CASES.....	37
5.1 Case 1: Publisher- Development of frontend functionalities using Scrum.....	37
5.2 Case 2: Public transport company- Replacement of transport planning application	39
5.3 Case 3: Legal assistance authority- Renewal of IT landscape.....	41
5.4 Case 4: Ministry of Defence- Agile development of command and control application	43
5.5 Case 5: Law system- Digitalization and automation of procedures	45
5.6 Case 6: Finance and assurance company: DevOps of datawarehousing tool	47
5.7 Case 7: Airline company- DevOps of business applications.....	48
5.8 Case 8: Transport and delivery company- DevOps business and customer applications.....	51
6. CROSS-CASE ANALYSIS	53
6.1 Approach of cross-case analysis	53
6.2 Cross-case overview	54
6.3 Comparison of organizational aspects	56
6.4 General features related to interview results.....	66
6.5 Expert meetings	70
7. INTERPRETATION	71
7.1 Neglecting of adjustments around teams.....	71
7.2 Understanding agile	77
7.3 Change needs time.....	80
7.4 Conclusion of interpretation: Three patterns	82
8. DISCUSSION	85
8.1 Contribution of findings	85
8.2 Discussing the limitations.....	86
9. CONCLUSION AND RECOMMENDATIONS	87
9.1 Answering the sub-questions.....	87
9.2 Answering the research question	90
9.3 Recommendations.....	91
9.4 Personal reflection	92
REFERENCES	95
APPENDICES	99
A. Interview structure	99
B. Questionnaire to score aspects	118
C. Scoring of organizational aspects.....	119

LIST OF FIGURES

Figure 1- Schematic presentation of difference in management style within an organization (own illustration).....	2
Figure 2- Schematic presentation of research scope (own illustration)	7
Figure 3- Coherence between research questions (own illustration)	8
Figure 4- Schematic overview of research method (own illustration)	9
Figure 5- The waterfall software development process (Braude & Bernstein, 2016).....	14
Figure 6- Plan driven method: variable costs and schedule with fixed requirements (Leffingwell, 2011)	14
Figure 7- Stacey Matrix- derived from (Bozzuto, 2011).....	15
Figure 8- The traditional triangle of constraints versus the agile triangle of constraints (Griffiths, 2015).....	17
Figure 9- The Scrum framework/ process derived from scrum.org	18
Figure 10- Overview agile frameworks (Portman, 2017, p. 45)	18
Figure 11- Conceptualization of software development in an organization - (Adopted from Mintzberg, 1983)	21
Figure 12- Six ideal-type organizational forms. (Hobday, 2000, p. 877)	23
Figure 13- Organizational form in an agile enterprise (Verbruggen, 2017, p. 125).....	24
Figure 14- Representation of McKinsey 7s model (Ravanfar, 2015)	26
Figure 15- Data collection and chain of evidence (own illustration).....	36
Figure 16- Schematic presentation of case 1 (own illustration).....	38
Figure 17- Schematic presentation of case 2 (own illustration).....	40
Figure 18- Schematic presentation of case 3 (own illustration).....	42
Figure 19- Schematic presentation of case 4 (own illustration).....	44
Figure 20- Schematic presentation of case 5 (own illustration).....	46
Figure 21- Schematic presentation of case 6 (own illustration).....	48
Figure 22- Schematic presentation of case 7 (own illustration).....	50
Figure 23- Schematic presentation of case 8 (own illustration).....	52
Figure 24- Schematic overview of cross-case analysis (own illustration)	53
Figure 25- Friction per aspect based on average of interview results, N=16. (own illustration).....	57
Figure 26- Friction per aspect based on average with distinction between public+ project and private+ BaU.....	67
Figure 27- Levels of project organization related to roles (own illustration)	73
Figure 28- Schematic presentation of flow of requirements from stakeholders to agile team(s) (own illustration). 75	
Figure 29- Schematic presentation of best practices derived from interviews regarding mutual adjustment.....	76
Figure 30- Schematic presentation of how nine interview aspects are directly related to the design of the IT governance above team level (own illustration).....	77
Figure 31- Schematic presentation of the adjustment of how the IT governance is related to the understanding of agile (own illustration).....	80
Figure 32- Schematic presentation of how the adjustment of the IT governance and understanding of agile are subjected to time (own illustration)	82
Figure 33- Schematic illustration of the patterns based on the interpretation of the interview findings. Linking the aspects to the IT governance, understanding agile and time (own illustration)	83
Figure 34- Agile values and agile triangle of constraints (own illustration).....	87

Figure 35- Schematic illustration of the patterns based on the interpretation of the interview findings. Linking the aspects to IT governance, understanding and time (own illustration)	89
Figure 36- Schematic presentation of adjustment of governance structure around agile teams (own illustration) ..	90
Figure 37- Schematische weergave opzet interview (eigen afbeelding)	99

LIST OF TABLES

Table 1- Agile and traditional project management comparators framework (Verbruggen, 2017)	20
Table 2- Explanation of the elements of McKinsey 7s model (Ravanfar, 2015)	26
Table 3- Aspects being relevant to an organization when using an agile approach and the originating sources (own table)	31
Table 4- Aspects obtained from literature that affect the success of agile software delivery in an organization (own table)	32
Table 5- Case selection criteria (own table)	34
Table 6- Overview of selected cases (own table)	34
Table 7- Overview of the agile software delivery cases with their general features (own table)	54
Table 8- Observations derived from the interviews in relation to the division of tasks and responsibilities (own table)	58
Table 9- Observations derived from the interviews in relation to the division of tasks and responsibilities (own table)	59
Table 10- Observations derived from the interviews in relation to mutual adjustment (own table)	60
Table 11- Observations derived from the interviews in relation to organizational culture (own table)	60
Table 12- Observations derived from the interviews in relation to management style (own table)	61
Table 13- Observations derived from the interviews in relation to resource management (own table)	62
Table 14- Observations derived from the interviews in relation to decision-making (own table)	62
Table 15- Observations derived from the interviews in relation to documentation style (own table)	63
Table 16- Observations derived from the interviews in relation to planning and control systems (own table)	64
Table 17- Observations derived from the interviews in relation to performance and reward systems (own table) ..	64
Table 18- Observations derived from the interviews in relation to staffing plan (own table)	65
Table 19- Relevant aspects in addition to the interview protocol (own table)	66
Table 20- Overlap between general features in relation to cases (own table)	66
Table 21- Case features describing the agile maturity (own table)	69
Table 22- Interview observations that can be explained by the absence of adjustments above team level (own table)	72
Table 23- Overview of some scaled agile frameworks and what roles these frameworks have defined above team level (own table)	74
Table 24- Overview showing how some of the interview observations can be related to the understanding of agile values or principles (own table)	78
Table 25- Time required and priority to change in relation to interview aspects (own table)	81
Table 26- Aspects obtained from literature that are relevant for the interaction between agile software delivery and the organizational context (own table)	88

1. INTRODUCTION

As the title of this research suggests, this research explores the organizational context around agile software delivery. This chapter presents a further introduction into the research topic. Section 1.1 describes the background by introducing agile management and the relevance of the organizational context on how projects are managed. The need for this research is justified by the perceived situation as discussed in Section 1.2. Section 1.3 introduces the graduation company, KWD Resultaatmanagement, which enabled the realisation of this research. Lastly, the reading guide in Section 1.4 presents the structure of the thesis.

1.1 Background: Agile management and the organizational context

Traditional project management approaches are not always well suited nowadays. Projects where the solutions or end-item requirements are uncertain or subjected to change demand a different approach to meet the desired results (Wysocki, 2014). This phenomenon is often being faced by projects in fast changing environments, such as the information technology (IT) industry (Xia & Lee, 2004). Agile project management has revealed itself as a management approach that copes with an unclear product scope and fast-changing circumstances (Owen, Koskela, Henrich, & Codinhoto, 2006; Wysocki, 2014).

The use of agile methods has become widespread over the last decades. Agility is increasing throughout organizations and across almost all industries at an accelerated rate as reported in the 11th Annual State of Agile report (Allisy-Roberts et al., 2016). Besides the increased use of agile management in practice, the research community has shown a growing interest in agile management as well (Dingsøy, Nerur, Balijepally, & Moe, 2012).

Projects and project management take place in an environment that is broader than that of the project. Understanding this broader context helps ensure that work is carried out in alignment with the organization's goals and managed in accordance with the organization's established practices. An organization's culture, style, and structure influence how its projects are performed (PMI, 2013). Therefore, research on project management should not be limited to the isolated project perspective, but should include an extended perspective on the organizational context (Engwall, 2003). Companies that are able to integrate the organisational processes with the company's projects will significantly increase the effectiveness of project management (Bodych, 2012). Thus, it seems of great importance to understand the organizational context when investigating the management of projects.

1.2 Fitting agile management within an organization

Many companies are still structured according to a traditional form of organization, characterised by a clear chain of command and existence of rules, procedures and different levels of management. These organizations often tend to be bureaucratic, less flexible, and slowly responding to change (Nicholas & Steyn, 2017, p. 486). This interpretation of a traditional organization tends to contradict the agile mind-set; lacking a strong hierarchy, flat working-structures, informality and embracing change (Fowler & Highsmith, 2001).

Figure 1 depicts a schematic presentation of a traditional, non-agile, structured organization, which conducts projects according to the agile management approach. Projects with an agile

project management approach embedded in a traditional organization might face numerous difficulties (Qumer & Henderson-Sellers, 2008) and are likely to be less successful (Cockburn & Highsmith, 2001). Difficulties can be traced back to organizational, technical, process or people factors, such as lack of management commitment or skill-set for example (Chow & Cao, 2008). Other research points out several management challenges, real and perceived, when implementing agile processes in traditional development organizations (Barry Boehm & Turner, 2005).

The MSc graduation research of Verbruggen (2017) showed how the role of the project manager is strongly changing in the evolution from traditional to agile project management. The qualitative multiple-case analysis of her research also points out that the difference in management approach is related to the project environment. This led to recommendations for future research to investigate the interaction of the organisational context with the agile management of projects.

The difficulties that organizations face when embedding an agile management approach within a non-agile organization, as portrayed above, is recognized in practice. Informal interviews with project managers of *KWD Resultaatmanagement*, see Section 1.3, confirm this view.

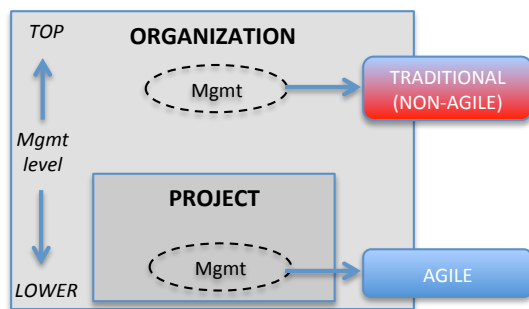


Figure 1- Schematic presentation of difference in management style within an organization (own illustration)

1.3 KWD Resultaatmanagement

The pursuance of this research is facilitated and supported by KWD Resultaatmanagement (KWDRM), a management consultancy firm specialized in project-, program-, and agile-management. KWDRM has been involved in a wide-range of different projects, with a predominant focus on the interface between IT and business. Its clients can be characterized as large, complex organizations in different type of branches, such as finance, logistics, industry and government.

KWDRMs primary ambition is to achieve results for clients in a pragmatic way. Next to that, KWDRM strives to be a learning company and therefore the experiences that are gained in practice are being used to acquire more knowledge in the field of (agile) project management. The necessity of this research is emphasized by KWDRMs observations from practice, since many organizations struggle to implement agile effectively. Next to that, this research contributes to their ambition to keep developing in the field of agile management.

1.4 Reading guide

This research is structured in the following way.

Chapter 1- **Introduction**

This chapter explains the broader context and the incentive for the research by explaining the problem.

Chapter 2- **Research design**

The research objective, research questions and scope explain what exactly is investigated. A description of the research method clarifies the approach that is utilized.

Chapter 3- **Theoretical framework**

Literature is discussed that is relevant to the research question. This chapter also presents the theoretical propositions resulting from the literature study.

Chapter 4- **Case study approach**

The case selection criteria are described and an introduction to the selected cases is given. An explanation is presented about how the multiple-case study is conducted.

Chapter 5- **Single cases**

All cases are individually described to get an impression of each single case. For each case, an introduction is given and some of the interview results are emphasized.

Chapter 6- **Cross-case analysis**

The separate interview results are compared with each other to find possible similarities across the cases. In this way an integral impression of the multiple-case study is presented.

Chapter 7- **Interpretation**

The outcome of the cross-case analysis is further interpreted to identify possible patterns. The interpretations are linked back to existing literature.

Chapter 8- **Discussion**

The discussion describes the significance of the findings, explains new insights and considers the limitations of the research.

Chapter 9- **Conclusion and recommendations**

This chapter discusses the answers to the sub-questions and presents an answer to the research question. Lastly, recommendations for future research and practice are presented.

2. RESEARCH DESIGN

This chapter describes the research design. The research objective is extracted from the perceived context, as described in the previous chapter. The determination of the research scope sets the boundaries of the research and results in the formulation of the research questions. Lastly, the method is discussed that is used to find an answer to the research questions.

2.1 Research objective

As described in the introduction, the implementation of agile methods is in practice often facing difficulties, resulting in disappointing outcomes. Some of these difficulties can be clarified by the presumption that the organizational context is unsuited to facilitate the implementation of agile approaches to projects. As Engwall (2003) points out, no project is an island and therefore should be considered in its organizational context. The aim of this research is to explore how organizations interact with the agile managing of projects. The outcome of this explorative research should lead to a better understanding of the challenges that organizations are facing when working agile and preferably lead to recommendations about how organizations can facilitate the effective use of agile management. The **research objective** is formulated in the following way:

“Explore the interaction between agile project management and its organizational context.”

2.2 Scoping the research

Defining the scope sets the boundaries of the research. Describing in what field of technology the research will be conducted, will narrow the research scope. The consideration of particular terms will further clarify the focal point of the research. Finally, the perspective of this research is best explained by describing the intended results.

2.2.1 Software engineering

Important part of this research is to explore how different organizations execute agile methods in practice. This part of the research requires organizations with a certain common ground. More common aspects between different organizations do better facilitate the comparison on the variables that are being examined.

Information technology (IT) is still the leading industry when it comes to practicing agile methods (Allisy-Roberts et al., 2016). Next to that, most of the scientific research regarding agile has been conducted in this field of industry (Dingsøyr et al., 2012). To ensure the availability of sufficient information in both the literature study and the practical study, this research is focussed on software engineering. Software engineering can be described as the application of a disciplined approach for the development and maintenance of computer software (Salem, 2009).

A drawback of narrowing down the scope to a specific set of organization is that possible biases are not identified and will influence the results of the research.

2.2.2 Delivery instead of project

Projects are defined in different ways. A well-known and widely accepted definition of a project is the following. A project is a sequence of unique, complex, and connected activities that have one goal or purpose and that must be completed by a specific time, within budget, and according to specifications (Wysocki, 2014). This definition implies a certain level of predictability in terms of the goal, time, budget and specification, and in that way describes a project as a *plan-driven* activity. Whereas agile strongly builds on a *value-driven* approach (Griffiths, 2015, pp. 84–85), which indicates a lower level of predictability. This way of thinking has an influence on project management. For example, more organizations choose to work in permanent development teams instead of project-based teams, to keep optimizing products and search for added value (Portman, 2017). Projects are started without defining a specific end time, budget and specifications, but are structured as an on-going activity.

As a result, a thin line is observed in the field of IT development between projects and on-going delivery. For example, the demand for a new IT product might be started as a new project, but in practice this product is developed with small delivery intervals and without a predefined end-date of product development. To deal with this apparent misunderstanding in definitions, this research is focussed on software *delivery* within organizations, whether this is project-based or on-going.

2.2.3 Focus on organization around delivery

This research builds on the research of Verbruggen (2017). Verbruggen investigated the role of the project manager, in the evolution from the traditional approach of project management towards the agile approach. Next to the perceived understanding of the changing role of the project manager, her multiple-case study results also emphasized the influence of the project environment on the practice of agile management. Various interviewees highlighted the influence of the organizational context on agile project management and experienced some kind of friction. Recommendations for future research suggested to further explore this phenomenon.

The case study results of Verbruggen exhibited how the transformation to agile project management affected the role of the project manager, but could not always clarify how the roles and responsibilities were distributed in the transformation to agile and how the surrounding organization filled in the gaps. One of the aspects that this research explores is how tasks and responsibilities are changing within traditional organizations that practice agile project management.

An important perspective in this research is that it investigates the interaction of agile delivery with the organizational context, from the perspective of software delivery. Although the external environment affects an organization, this won't be investigated into depth.

2.2.4 Scoping the result

Figure 2 depicts a schematic presentation of the research scope. The research focuses on the interaction between agile software delivery and its organizational context. The red arrows in the figure schematically present the interaction. This research aims to understand this

interaction and to indicate what kind of organizational context is suited for agile software delivery.

The types of organizations that are being examined are not agile, but rather more the traditional, command-and-control type of organization as described earlier. Within these organizations software delivery is being practiced, whether this is project-based or on-going. This software delivery is conducted agile, in which the agile maturity level may vary per organization.

The results of this explorative research contribute to an improved understanding of the difficulties that organizations are facing when conducting agile software delivery. Ideally, these results lead to recommendations about how organizations can facilitate the effective use of agile software delivery.

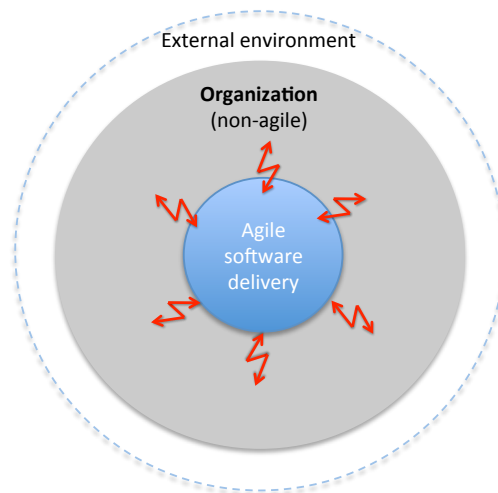


Figure 2- Schematic presentation of research scope (own illustration)

2.3 Research questions

The described objective and scope has resulted in the following **research question**:

“What kind of adjustments can an organization make to better facilitate agile software delivery?”

The following sub-questions are identified to help answering the above research question. Figure 3 schematically describes the coherence between the questions.

1. *How is agile software delivery described in literature?*

Existing literature should be studied to understand how software delivery is managed. Within this search into software delivery, closer attention will be paid to the agile approach. The answer to this question creates the context around the research subject.

2. *Which aspects are described in literature as being relevant for the interaction between agile software delivery and the organizational context?*

This question will be answered by conducting an explorative literature study to discover what is written about the interaction between an organization and agile management. An overview is gained of the organizational aspects that interact with agile delivery. A framework will be used to structure these aspects.

3. *In what way does agile software delivery interact with its organizational context in practice?*

An answer to this question is given by conducting a multiple-case study into different organizations. The aspects from sub-question 2 are used to make an assessment of the different cases.

4. *What patterns can be obtained from the interaction of agile software delivery with its organizational context?*

The data gathered in the case studies are subjected to a cross-case analysis. By comparing the case data, an appreciation can be given about potential differences and similarities. Observed patterns will be explained by linking the findings back to literature.

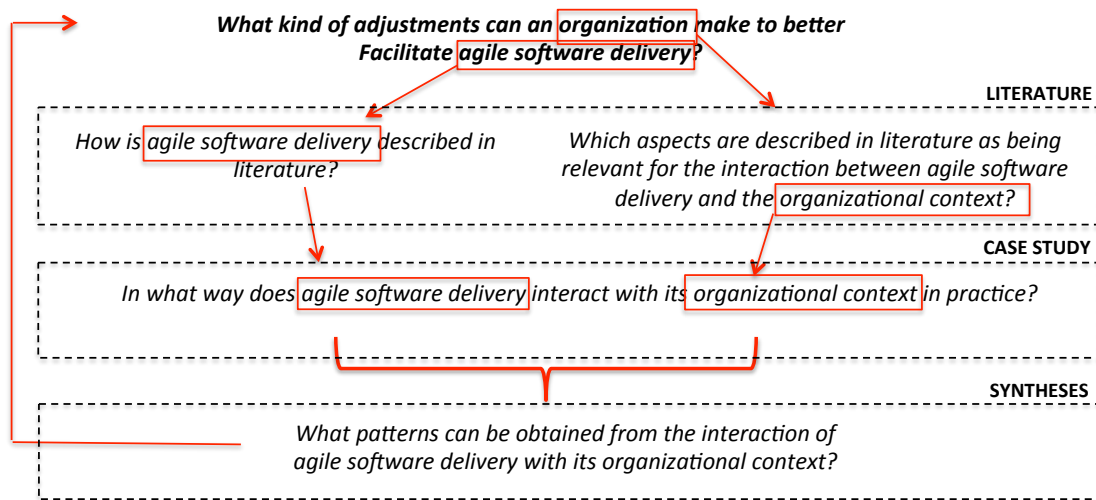


Figure 3- Coherence between research questions (own illustration)

2.4 Methodology

This section explains in what way the research is conducted. Qualitative research is primarily a suitable research strategy for exploration (Boeije, 2014, p. 50). Therefore, the practical data for this qualitative research is collected by executing a multiple-case study, meaning that multiple, contemporary phenomena are investigated in depth and in its real-world context (Yin, 2014, p. 237). More specific, multiple cases are scrutinized in a qualitative way by conducting semi-structured interviews. The results of the literature study are deductively derived, meaning that existing theory is studied prior to the case studies (Boeije, 2014, pp. 100–104). The studied theory results in theoretical propositions that steer the data collection. Figure 4 shows a schematic overview of the research approach. An explanation of each research step is presented below the figure.

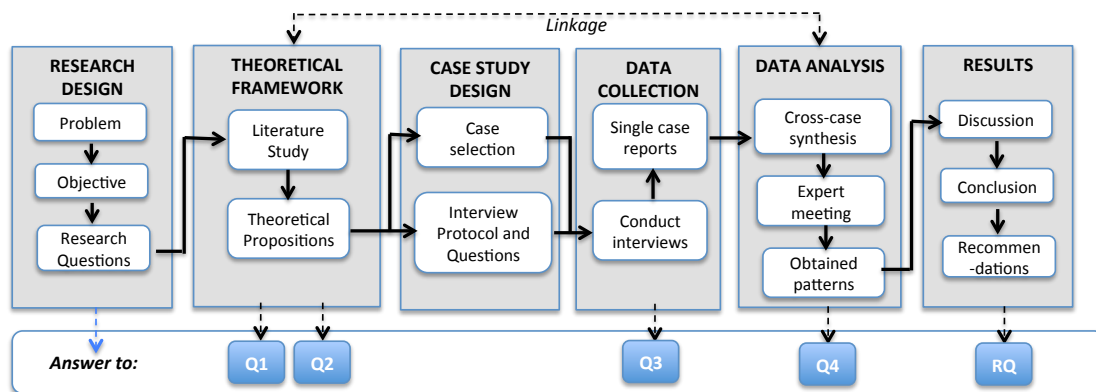


Figure 4- Schematic overview of research method (own illustration)

Research Design

The first step is the research design, or conceptual design, and describes *what*, *why* and *how much* will be investigated (Verschuren & Doorewaard, 2010, p. 16), by presenting the problem, objective and research questions. The formulated sub-questions are answered throughout the research process, as presented in Figure 4, finally contributing to the conclusion and giving an answer to the research question.

Theoretical Framework

The research design is followed by a literature study to design the theoretical framework. The role of the theoretical framework is to give input to the multiple-case study; for example to select suitable cases and draw up interview questions. The literature study will give an answer to the first two sub-questions and in that way identify concepts that are relevant to the research. An important step to finalize the theoretical framework is to translate abstractly defined core concepts into observable indicators, which can be used to collect qualitative data. This process is called *defining the key concepts* (Verschuren & Doorewaard, 2010) or *theory development* (Yin, 2014, p. 37). This results in *theoretical propositions* that are used to design the multiple-case study and to generalize the case study results in a later stage (Yin, 2014, p. 40).

Case Study Design

A multiple-case study is conducted to explore how agile software delivery interacts with its organizational context in practice. Suitable cases are selected based on the outcome of the literature study, which framed the selection criteria. During the case selection an estimate is made for each case, about the available data and relevance to the research subject. Each case study aims to obtain a general idea of the object as a whole, i.e. holistic method. This method manifests itself in the use of a qualitative and open way of data gathering, in this case most likely in the form of semi-structured, open interviews (Verschuren & Doorewaard, 2010, p. 179). The theoretical propositions derived from the literature study will serve as a basis to formulate the interview questions.

Data Collection

Face-to-face interviews consisting of open questions are held with the selected candidates. Principles that are of importance during the data collection are the use of multiple sources of evidence, usage of a case study database and maintaining a chain of evidence (Yin, 2014, pp.

118–128). The results of each case will be documented in a single case report. The single case reports will provide an answer to sub-question 3.

Data Analysis

The found data will be subjected to a cross-case synthesis. In this synthesis, the results for each individual case will be examined and the pattern of results across the cases will be observed. In that way exploring whether the cases being studied had replicated or contrasted with each other (Yin, 2014). An expert meeting will be held to verify the findings of the cross-case synthesis and enrich the analysis if possible.

Further interpretation of the findings give an answer to sub-question 4. The results of the analysis will be linked to the findings of the theoretical framework and be further interpreted to explain the observations. By analysing how the organizational aspects relate to the agile management of software delivery, an impression can be obtained about what kind of organizational design is best suited to facilitate agile delivery.

Results

The research findings will be interpreted in the discussion, to identify possible limitations and describe the significance of the findings in a wider context. The limitations of the case study results will be tested in accordance with four tests that are commonly used to establish the quality of empirical research; construct validity, internal validity, external validity and reliability (Yin, 2014, pp. 45–49). By generalizing the findings of the case study, an answer to the formulated research question can be found, presenting the conclusion of the research. Recommendations will be made about possible practical implications of the findings and suggestions for future research.

3. THEORETICAL FRAMEWORK

This chapter presents the theoretical framework. The theoretical framework is the structure that can hold or support a theory of a research study and introduces plus describes the theory that explains why the research problem under study exists (Abend, 2008). The outcome of this theoretical framework results in the formulation of theoretical propositions that will be used to design the multiple-case study, discussed in the last section of this chapter.

The aim of this theoretical framework is twofold and related to the first two sub-questions:

- To acquire a better understanding of agile software delivery (sub-question 1);
- To obtain aspects which are relevant to describe the interaction between agile software delivery and the organizational context (sub-question 2).

These aspects will respectively be covered in the following two sections. The last section (3.3) will discuss the theoretical propositions that are derived from the literature review.

3.1 Agile software delivery

This section presents the essentials of software delivery, more specifically *agile* software delivery, sketching the context around the research subject. This part of the literature study is primarily focussed on the fundamentals of software delivery. Many of the findings will therefor stem from theoretical books in the field of software delivery, such as *Software Engineering -Modern Approaches* (Braude & Bernstein, 2016), *Agile Software Requirements* (Leffingwell, 2011), *Software Engineering- Architecture-driven Software Development* (Schmidt, 2013) and *Software Engineering* (Salem, 2009). This section will contribute to the research by answering the **first sub-question**:

How is agile software delivery described in literature?

The answer to this question can in a later stage of the research be used to assess cases regarding the level of agility. The question is further decomposed into smaller aspects to scope the literature study. The following questions will be answered in the sections 3.1.1 to 3.1.4:

- What is software delivery?
- How is software delivery managed?
- What aspects describe agile software delivery?
- How can organizations that conduct software delivery be described?

3.1.1 Introduction to software delivery

As described earlier, the scope of this research is bounded to software delivery. Therefor, this literature review starts by giving an introduction to the fundamentals of software delivery. The intent of this part of the literature review is not to gain an in-depth understanding of software delivery, but rather to attain an impression of what this research interprets as software delivery and what aspects are relevant to take into account. This introduction includes rudimental information regarding development stages, the business in relation to technical development, and types of releases.

Software or computer software consists of the computer program and its related documentation. A computer program consists of instructions that perform certain tasks on computer hardware (Salem, 2009, p. 2). The Institute of Electrical and Electronics Engineers (IEEE) describes software as ‘computer programs, procedures, and possibly associated documentation and data pertaining to the operation of computer system’ (IEEE Standard, 1990). Software can broadly be classified into two categories; system software and application software. *Systems software* is usually engaged in background processes and typically acts as a middle layer between hardware and user applications. *Application software*, also referred to as end user software, allows users to perform some specific tasks and is more common to the users (Salem, 2009, pp. 4–5). An operating system is an example of systems software. Application software is also commonly known as *app* and has a diverse field of use, such as education, gaming, social networking, etc.

Software engineering is an engineering discipline that involves all aspects of developing and maintaining a software product (Braude & Bernstein, 2016, p. 2). A commonly used definition of software engineering is ‘The application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software’ (IEEE Standard, 1990). As this definition indicates, software engineering is not only about what is delivered, but also about *how* to deliver it.

To understand how software is delivered, the so-called **4P’s** of software engineering will be discussed: people, product, project and process (Braude & Bernstein, 2016; Salem, 2009).

People

Different groups of people are involved or have a stake in a project’s outcome; these people are called the stakeholders. These stakeholders might have different viewpoints on the process and project success, given the different roles and perspectives. The exact division of roles within software development differs per project, but roughly speaking the following groups can be identified (Braude & Bernstein, 2016; Yilmaz, O’connor, & Clarke, 2015).

The core is formed by the **development teams**, consisting out of software engineers, testers and architects, and are responsible for developing and maintaining the software. Software development includes many tasks such as requirements gathering, software architecture and design, implementation, testing and documentation. Organizations can appoint **project managers**, which in many ways serve as the link between the development team and the business. Project managers are responsible for planning and tracking a project, and are involved through managing the people, process, and activities.

Business management are the people responsible for the business side of the company developing the software, such as senior management, marketing or development managers. They are typically not particularly knowledgeable about or involved in the technical aspects of the project. People who use the software after it is developed are named **end users**. These end users might be people within the organization that developed the software, or external people that purchased the software, i.e. **customers**.

The above description of stakeholders with the adherent tasks and responsibilities is to a large extent based on traditional methodologies. How this is exercised in agile approaches will be further discussed in the following sections of this chapter.

Product

Software products may support a business process, control the operation of a system or process, support data gathering and analysis activities, or provide some entertainment

relevance (Schmidt, 2013). The requirements that describe the intended outcome of the software product are described by the business as functional requirements and translated into technical requirements by the development team. The products of software delivery do not only entail the software itself, but might also include documents, such as project-, test- and customer documents (Braude & Bernstein, 2016).

A special feature of software is that its medium is electronic in nature, which makes software intangible. Nevertheless, similar to other products, it needs to be designed, implemented and tested before it is operational. Most human-made products are comprised of multiple components and parts, which is also applicable to a software product. Software can be decomposed into smaller building blocks or elements that comprise software products, including functions, procedures, applications, etc. By assembling these building blocks in the correct matter, within the data processing system it is designed to perform with, a software product is created (Schmidt, 2013).

A common way to describe software is in terms of its behaviour, structure and architecture. The *structure* consists of the fixed set of instructions to the program. When subjected to internal events or external inputs from other systems, or users, the software behaves according to the behavioural specification. All possible software reaction determines the *behaviour* of the software, which can be modelled by diagrams. The *architectural* aspect of software addresses some of the non-functional issues such as robustness, performance, resiliency and security (Salem, 2009, p. 8).

Project and process

A software product is produced according to a set of activities. The delivery of every software product involves a similar set of activities, such as planning, requirement analysis, design, implementation and testing. As stated earlier, software engineering is related to the use of a disciplined approach to software development. A software *project* or *process* is the compilation of activities required to develop a particular software product, carried out in an organized and disciplined manner. It imposes structure and helps to guide the many people and activities in a coherent manner (Braude & Bernstein, 2016; Salem, 2009). Different frameworks, methods and techniques do exist to achieve this, which are further scrutinized in Section 3.1.2. Depending on whether the activities have one predefined end-date or the activities are considered to be on-going, one can speak of respectively a project or a process (Cagara, 2017). The on-going activities in processes can be demarked by regular end-dates, also known as releases.

3.1.2 Managing software delivery

Software delivery is about the execution of a set of activities in an organized and disciplined manner. Various software development methodologies and process frameworks do exist to structure, manage and control those activities. This section will further elaborate on the matter in which these activities can be arranged by elaborating on different management approaches and in that way answering the question: *How is software delivery managed?* This section will first discuss the principles of traditional project management; thereafter consider its shortfalls related to software delivery and how agile management can cope with this.

Traditional management- a plan driven approach

The software industry advanced quickly after its inception in the 1950s and 1960s and as it did, the need to be able to better predict and control ever larger-scale software projects

resulted in what has become known as the sequential, stage-gated “waterfall” software process model. The model was first defined by Winston Royce (Royce, 1970), and can usually be typified by an illustration as depicted in Figure 5. In this model, software development occurred in an orderly series of sequential stages (Leffingwell, 2011, p. 5), or in other words the phases of software development are conducted *linearly*.

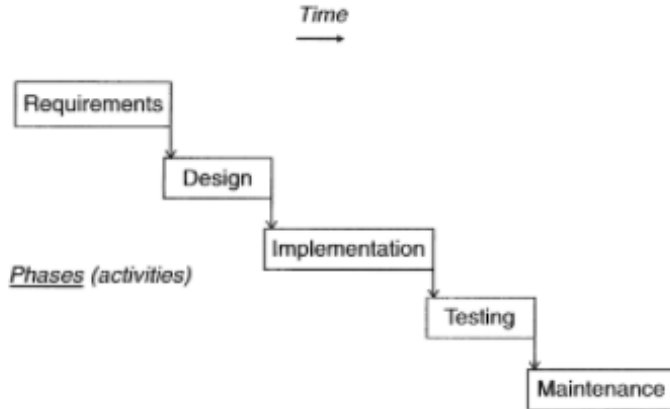


Figure 5- The waterfall software development process (Braude & Bernstein, 2016)

The waterfall method is considered to be plan-driven, which implies that there is a set of requirements that can be determined “up front” as a basis to estimate the schedule and the budget of the project, being more variable, as depicted in Figure 6 (Leffingwell, 2011). Based on the requirements, a complete project plan can be developed. It specifies all of the work that is needed to meet the requirements, the scheduling of that work, and the staff resources needed to deliver the planned work. This plan driven method tends to be suitable for projects with a low complexity, few scope changes and low risk (Wysocki, 2014). Plan-driven methods emphasize a rationalized, engineering- based approach in which it is claimed that problems are fully specifiable and that optimal and predictable solutions exist for every problem (Van Vliet, 2008).

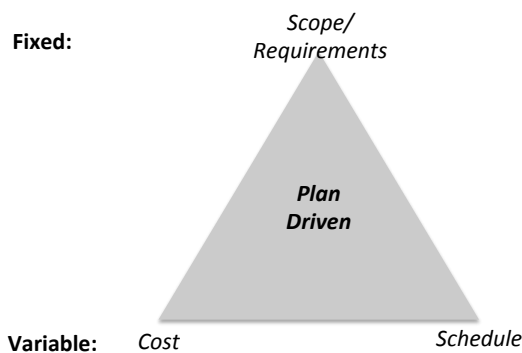


Figure 6- Plan driven method: variable costs and schedule with fixed requirements (Leffingwell, 2011)

The waterfall process is considered the simplest model and forms the basis for most others. A pure waterfall process indicates that phases are implemented in sequence, with no phase starting before the previous has almost completed (Braude & Bernstein, 2016). The best-known plan-driven methodology is Projects in Controlled Environments, or PRINCE2, released in 1996 (Axelos, 2018a).

Why traditional management fails in software development

Large scale research by the renowned Standish Group, an independent, international IT research advisory firm, reports a continuous high rate of failing software projects (Hastle & Wojewoda, 2015). Common factors that contribute to failure of software projects are among others, unrealistic or unarticulated project goals, inaccurate estimates of needed resources, badly defined system requirements and poor software design methodologies (Charette, 2005). Other research also appoints patterns of failure in software projects due to organizational risks, that are directly affected by the chosen management approach (Shehzad, Awan, Lali, & Aslam, 2017).

Software development can be considered as a design process rather than being a production process. This has implications for how it should be managed. In particular, it is futile to chase predictability and important to chase value. To maximize adaptability, it is essential to have good, fast feedback loops (Narayan, 2015). A plan driven approach is based on predictability, which is not always the case in software projects. A value driven approach on the other hand, doesn't demand to define everything upfront, but focuses on the value that is being added during the development.

The main reason why a value driven approach is better suited than a plan driven approach can be found by considering a key characteristic of software development, which is *we learn at least some of what it takes as we go about development* (Narayan, 2015, p. 28). Any software development team has to deal with the so-named 'unknown unknowns'; during the course of development, the team encounters unanticipated challenges. So in general, it is not possible to do upfront work to eradicate unknown unknowns.

A frequently used tool to understand when one should adapt a more traditional approach, or a more agile approach, is the Stacey Matrix as depicted in Figure 7. Ralph Stacey created a matrix based on agreement and certainty to show conditional management approaches based on these two dimensions (Stacey, 1993). The y-axis in this graph represents the level of agreement about a decision or problem to be confronted amongst the stakeholders. The x-axis represents the certainty around the technical details of the solution (Bozzuto, 2011). The traditional waterfall approach is in general well suited for projects that are classified as 'simple' according to this matrix, having a high level of certainty and are close to agreement. When no agreement is reached or a higher uncertainty is in place, for example due to the 'unknown unknowns', an agile project approach might be more appropriate.

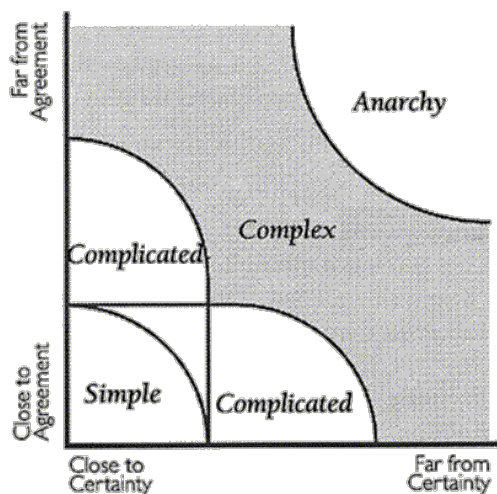


Figure 7- Stacey Matrix- derived from (Bozzuto, 2011)

Agile management- a value driven approach

Starting in the late 1990s and evolved through the current decade, agile management has revealed itself as a way-of-thinking that anticipates on changes. Agile is an umbrella term for several iterative development methodologies, that are based on the same vision and core values (Griffiths, 2015; Leffingwell, 2011). In 2001 a group of seventeen renowned software developers, including Martin Fowler and Jim Highsmith, came together to discuss the variety of agile approaches and to seek for common ground, resulting in the Agile Manifesto (Fowler & Highsmith, 2001). Although many different agile methodologies do exist, they can all be related to four values described in the Agile Manifesto.

Individuals and interaction	over	<i>processes and tools</i>
Customer collaboration	over	<i>contract negotiation</i>
Working software	over	<i>comprehensive documentation</i>
Responding to change	over	<i>following a plan</i>

In addition to the four agile values, the authors of the Manifesto created twelve guiding principles for agile methods (Fowler & Highsmith, 2001):

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

Several methodologies or frameworks stem from these four values and twelve principles. The best known methods are Scrum, Extreme Programming (XP), lean product development, Kanban, Feature-Driven Development (FDD), Dynamic Systems Development Method (DSDM), and the Crystal family of methods (Griffiths, 2015). Despite significant differences between the methods, they all originate from the same mind-set.

A key difference between the agile mind-set and traditional project management is the agile or “inverted” triangle of constraints, depicted in Figure 8. The turnaround of the traditional triangle implies that agile methods allow scope to vary within the fixed parameters of cost and time. In other words, those methods aim to deliver the most value they can before a set date and within a set budget (Griffiths, 2015). This conceptualisation can be explained by

looking at, for example, the agile method *Scrum*. According to Scrum, the costs are fixed since permanent teams are established that do not change throughout the delivery. The time is fixed by delivering software according to a predefined timeslot, named a sprint (Schwaber & Sutherland, 2017). The Scrum method is further elaborated below.

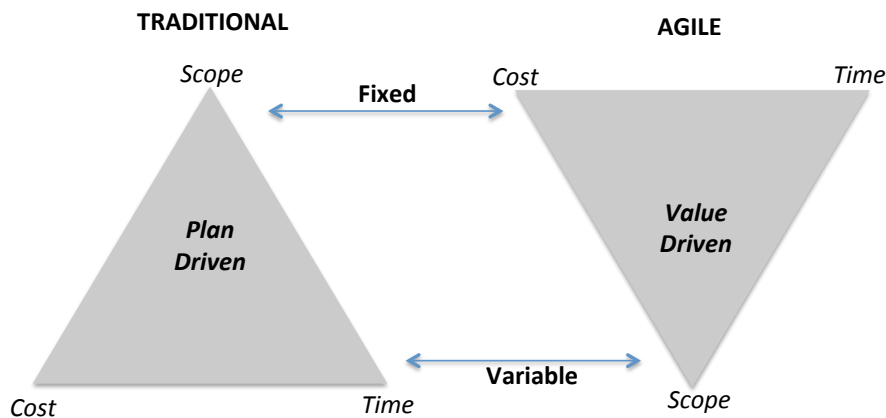


Figure 8- The traditional triangle of constraints versus the agile triangle of constraints (Griffiths, 2015)

Without discussing each agile methodology into detail, one method will concisely be reviewed to gain a better understanding of the working principal. By far the most widely used agile methodology nowadays is Scrum. The *11th Annual State of Agile* report states that of all the agile methodologies used within the respondents organizations, Scrum is still the most common with 68% (Allisy-Roberts et al., 2016).

Scrum in a nutshell

Figure 9 shows a schematic presentation of the Scrum process. This process will be discussed to gain a better understanding of the agile approach. Next to that, since Scrum is by far the most applied agile method, it is likely this method will be encountered during the multiple-case study. The following explanation is derived from the Scrum-guide (Schwaber & Sutherland, 2017) and the book *Scaling Agile in Organizations* (Portman, 2017, pp. 7–14).

Scrum is a framework for developing, delivering, and sustaining complex products. It was initially intended for software development, but has spread to other disciplines as well. Scrum is primarily focussed on team level. Central in this framework is the *Scrum team*, which is a self-organizing, co-located and cross-functional team, consisting of a *product owner*, the *development team*, and a *Scrum master*.

The Scrum master is responsible for promoting and supporting Scrum as defined by the Scrum-Guide. He acts as a facilitating coach of the development team as well as the product owner. The product owner is responsible for maximizing the value of the product and is the sole person responsible for managing the *product backlog*. The product backlog is a prioritized features list, containing a short description of all functionality desired in the product. Based on this product backlog, the Scrum team conducts the sprint planning in which the work is planned and divided, resulting in the *sprint backlog*; a list of tasks identified by the Scrum team to be completed during the Scrum sprint.

The heart of Scrum is the *sprint*, a time-box of one month or less during which a “done”, useable, and potentially releasable product increment is created. The *daily Scrum* is held every day of the sprint, in which the development teams come together for 15 minutes to discuss the progress and planning for the upcoming 24-hours. A *sprint review* is held at the

end of the sprint to inspect the increment and adapt the product backlog if needed. The *sprint retrospective* is an opportunity for the scrum team to inspect itself and create a plan for improvements to be enacted during the next sprint.

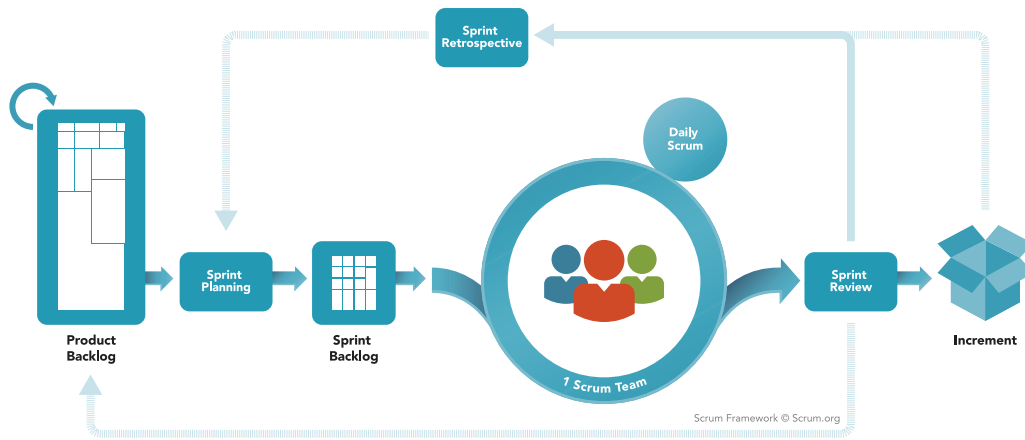


Figure 9- The Scrum framework/ process derived from scrum.org

Scaling agile

Multiple teams require coordination, managing of dependencies and integration of sub-products. In case an agile approach, like Scrum, is extended towards multiple teams or the whole organization, one speaks of *scaling agile*. This phenomenon was first described by Jeff Sutherland in the article ‘Agile Can Scale: Investing and Reinvesting SCRUM in Five Companies’ (2001) by introducing a mechanism named Scrum of Scrums, SoS.

Nowadays, approximately thirty frameworks do exist that can be used to scale agile within an organization. Figure 10 gives an overview of the best-known agile frameworks and how these can be used on which level of an organization (Portman, 2017).

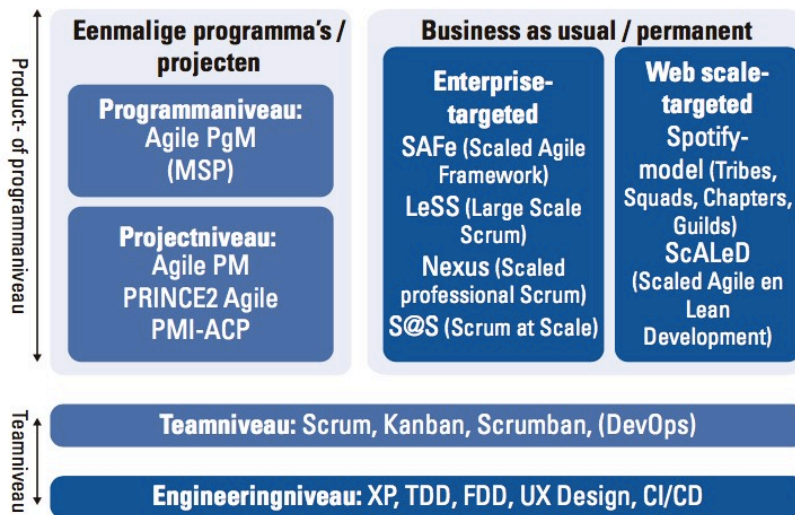


Figure 10- Overview agile frameworks (Portman, 2017, p. 45)

The agile frameworks presented on the right side are designed to facilitate permanent software development and can be divided into enterprise targeted and web scale-targeted. The enterprise targeted frameworks include some commonly used frameworks, such as SAFe, LeSS and Nexus. For these it applies that several teams are working on a single complex product or value stream. The other group includes frameworks to support IT-departments in

the development and maintenance of dozens to hundreds of different software applications, such as the Spotify-model and ScALeD. The frameworks that are positioned on the left side of Figure 10 present the frameworks that can be used for scaling agile within temporary endeavours, such as projects or programs (Portman, 2017).

The above comparison of frameworks shows that no framework exists, which is suitable for all organizations. Next to that, criticism does exist about the use of the scaled frameworks. Practitioners in the field of software development argue that SAFe and other frameworks are highly prescriptive, heavily emphasizing the use of its particular practices and rules, without leaving much room for customization on the part of the organization (Powell-Morse, 2017). Other criticism point to the observation that scaled framework diminish the level of agility, since it strives to bring decision-making and control back to the management and executives (Denning, 2015; Powell-Morse, 2017).

3.1.3 Defining the level of agility

Different projects require different management approaches. Although agile management seems to be a suitable method in the field of software delivery, different kinds of methods are being practiced. Since this research aims to investigate the relation between agile software delivery and the organizational context, it is desired to determine the agility of the software delivery. This section will present different aspects that can be used to define agile management. The obtained aspects can be used to classify the level of agility within a certain project in a qualitative way.

To determine the level of agility in practice, it is required to identify on which specific aspects the agile and traditional approaches differ. Existing literature elaborates on a wide span of different ways in which agile and traditional approaches differ. Next to that, researchers use different terms when comparing both approaches (Dingsøyr et al., 2012).

The MSc thesis of Verbruggen (2017) compiled aspects from literature that make a difference between the agile and traditional approach. The aspects are categorized into five comparators that repeatedly were found in literature (Nerur, Mahapatra, & Mangalaraj, 2005; Vinekar, Slinkman, & Nerur, 2006) resulting in philosophy, organisation and management, development process, people and team, and technology. The found aspects are summarized in a comparators framework, presenting a comprehensive overview of important differences between both management styles. Table 1 shows the agile and traditional management comparators framework (Verbruggen, 2017, p. 23).

COMPARATORS	ASPECTS	AGILE	TRADITIONAL
PHILOSOPHY			
Mindset		Individuals and interaction	Processes & Tools
		Working software	Comprehensive documentation
		Customer collaboration	Contract negotiation
		Responding to change	Following the plan
ORGANIZATION AND MANAGEMENT			
Organization	<i>Structure</i>	Flat team-based structure	Hierarchical structure
	<i>Form</i>	Flexible & participative encouraging cooperative social action (<i>organic</i>)	Bureaucratic with high formalization (<i>mechanic</i>)
	<i>Culture</i>	Comfort and empowerment via many degrees of freedom (<i>thriving on chaos</i>)	Comfort and empowerment via framework of policies and procedures (<i>thriving on order</i>)
Management	<i>Management style</i>	Leadership-and-collaboration	Command-and-control
	<i>Decision making</i>	Decentralized & pluralist decision making	Centralized & managerial decision making
DEVELOPMENT PROCESS			
Development methods	<i>Development style</i>	Iterative, adaptive, extreme	Linear, incremental
	<i>Development model</i>	Evolutionary delivery model; e.g. <i>Scrum, XP, DSDM, FDD</i>	Life cycle model; e.g. <i>Waterfall model, spiral model, V-model</i>
Development approach	<i>Project cycle</i>	Guided by project features	Guided by tasks or activities
	<i>Iron triangle</i>	Resources and time fixed	Scope (solution) is fixed
Development direction & nature of planning	<i>Development direction</i>	Adaptable; readily changeable	Pre-planned; fixed
	<i>Planning approach</i>	Planning is done prior and for every iteration	Rigorous planning for the entire project
Value delivery frequency	<i>Value delivery</i>	Frequent value delivery; after every iteration (timebox) value is delivered to the customer	At the end of each phase/ at the end of the project the value accepted by the customer
Dealing with change	<i>Change</i>	Change is inevitable, dealt with after every iteration	Threat for meeting the plan, not dealt with until the next release
PEOPLE AND TEAM			
Teamwork	<i>Team composition</i>	Small teams, collaborative work	Large teams, individual work
	<i>Team location</i>	Co-located teams	Not always co-located teams
	<i>Role assignment</i>	Self-organising teams & encourages role interchangeability	Individual & favours specialisation
	<i>Skills</i>	Interpersonal & multidisciplinary skills	Specialized skills
	<i>Reward systems</i>	Team award systems	Individual awards systems
Customer involvement	<i>Customer involvement</i>	High customer involvement; dedicated customers focused on prioritized increments	Low customer involvement; as-needed customer interactions focused on contract provisions
	<i>Customer location</i>	Co-located customer	Not always co-located customer
Attitude to learning	<i>Learning type</i>	Double loop learning	Single loop learning
TECHNOLOGY			
Requirements	<i>Definition of requirements</i>	Requirements can undergo unforeseeable change, and consist of prioritized informal stories	Requirements undergo a foreseeable evolution and formalized (e.g. <i>Projects, capabilities, interfaces, quality</i>)
	<i>Clarification of requirements</i>	Requirements discussed and clarified "just-in-time"	Requirements at the beginning of the project (<i>Contract driven; requirements serve as contract</i>)
Testing		Executable test cases define requirements testing	Documented test plans and procedures
		Write test prior to code (<i>test-driven development</i>)	Write code prior to test
	<i>Timing of testing</i>	Testing early in the development process	Testing late in the development process
	<i>Frequency of testing</i>	Testing on every iteration (incl. Customer acceptance testing)	Testing after coding phase completed (incl. <i>Customer acceptance testing</i>)
Release frequency		High release frequency (" <i>go live</i> " release per one to four weeks)	Low release frequency (" <i>go live</i> " release per six or more months)
Project metrics and documentation	<i>Documentation</i>	Minimal, up-to-date metrics	Emphasis on data collection
	<i>Knowledge & communication</i>	Tacit knowledge & informal communication	Explicit knowledge & formal communication
Coding	<i>Design</i>	Simple design; refactoring assumed in-expensiv	Extensive design; refactoring assumed expensive
	<i>Code ownership</i>	Collective code ownership	Not always collective code ownership

Table 1- Agile and traditional project management comparators framework (Verbruggen, 2017)

3.1.4 Organization around software delivery

Before the interaction between agile software delivery and the organization is scrutinized, it is relevant to derive a better understanding of such an organization. This section will focus on the type of organizations that are involved in software delivery and in that way give an answer to the question 'How can organizations that conduct software delivery be described?'

A large variety of different organizations are involved in the development of software products. Companies which primary products are various forms of software are named *software houses*, which are companies within the *software industry* (O'Grady, 2014). This research is focused on another segment of companies; the ones that develop and use software to support their primary business goal. This type of software could be part of the product or service that these organizations put on the market, or as a supportive tool within the company, e.g. administrative functionality. These kinds of organizations could operate in the private sector, such as companies within the financial or transport sector, but also in the public sector, such as educational institutions, agencies or ministries.

Although these companies vary in many different ways, the commonality in how software delivery is organized is found by conceptualizing these organizations into two entities; the *business* and *IT department*. Although the term *business* is used widespread in both literature and practice when discussing software development and its organizational context, a common definition cannot be found. Several scholars have studied the interaction between the IT department and the business, without defining what these entities do entail (Bassellier & Benbasat, 2004; De Haes & Van Grembergen, 2009; Luftman, 2003).

What this research exactly intends when discussing these entities is explained by using the five basic parts of the organization as described by Mintzberg (1983, pp. 9–19); the operating core, the strategic apex, the middle line, the technostructure and the support staff. Within this research, the IT department is conceptualized as part of the operating core of an organization and the business covers the other four parts and other departments of the operating core, as shown in Figure 11.

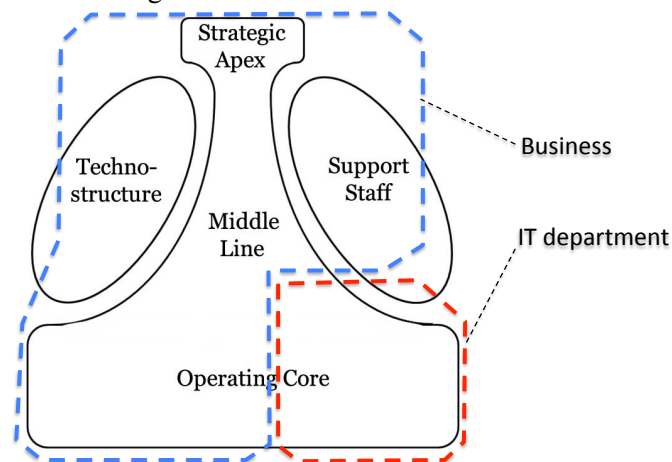


Figure 11- Conceptualization of software development in an organization into business and IT department- (Adopted from Mintzberg, 1983)

Business

Within this research, business is considered as other, not-IT, departments of the operational core and the part of the organization that steers, coordinates and facilitates the core activities of an organization, which is not part of the operating core. The other departments of the operational core perform the basic work related directly to the production of products and services that are not related to software delivery. In this research, these will be the departments that deliver the primary product or service of the organizations.

The strategic apex consists of people charged with overall responsibility for the organization, such as a chief executive officer, CEO, or any other top-level managers. The strategic apex is joined to the operating core by the chain of middle-line managers, which runs from the senior managers to the first-line supervisors, who have direct authority over the operators. In the

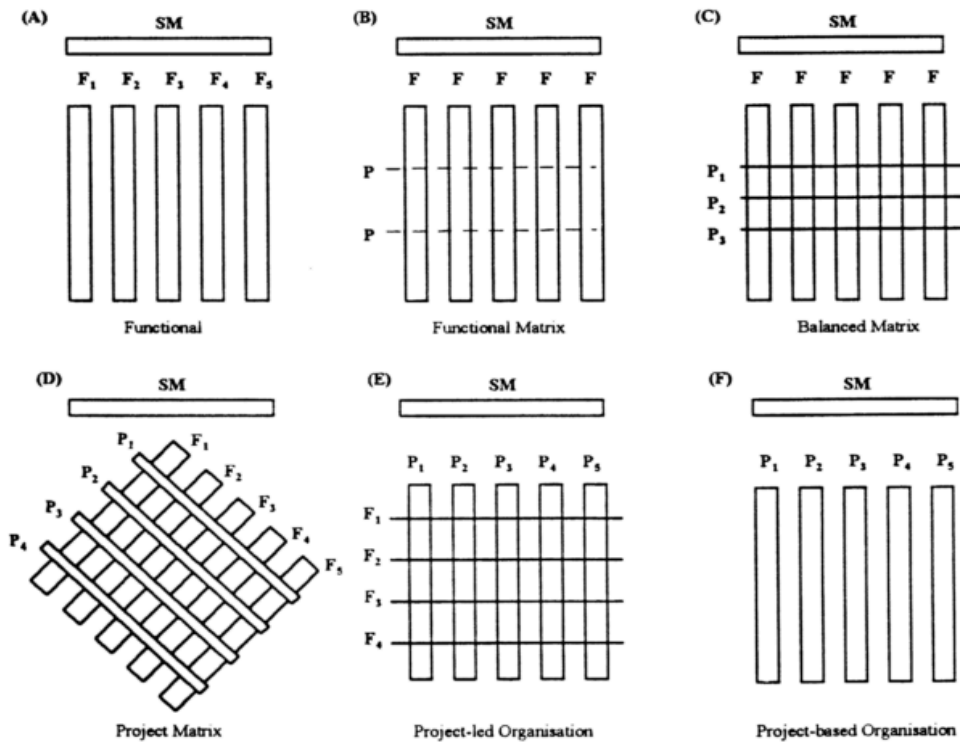
technostructure we find the analysts who serve to affect certain forms of standardization in the organization. Lastly, the support staff contains all specialized units that exist to provide support to the organization outside its operating work flow (Mintzberg, 1983, pp. 13–16). Although the IT department also supports the primary business goals of an organization by enabling working software, it is in general considered as an independent department within an organization and therefore conceptualized as part of the operational core within this research. A multitude of different business layouts do exist, of which a few entities are considered to be of importance to this research. Most organizations nowadays are engaged in some kind of project activity (Maylor, Brady, Cooke-Davies, & Hodgson, 2006, p. 1) and therefore might appoint a *project office* or *project management office*, *PMO* as part of the business. The project office is responsible for planning, directing and controlling project activities and for linking the project teams, users, and top management. When the office must coordinate multiple projects and is larger, it is commonly named the PMO (Nicholas & Steyn, 2017, p. 498).

A *program* is a set of projects and other activities organized and coordinated to achieve an overarching goal. The superlative is a *portfolio*, which is a group of projects and programs aimed at strategic objectives that share resources and compete for funding (Nicholas & Steyn, 2017). Organizations with a larger amount of projects might appoint program managers or portfolio managers within the business. The aim of the portfolio manager is to achieve organizational objectives through the investment in programs and projects, including selection and setting priorities.

IT department

The IT, Information Technology, department is in this thesis considered as part of the operating core. The operating core encompasses those members who perform the basic work related directly to the production of products and services (Mintzberg, 1983, p. 12). In case of the IT department, this relates to the production of software.

A wide range of frameworks and theories exist in literature that describe how this IT department can be structured. Based on this literature, Hobday (2000) provides a comprehensive overview of six ideal-type organisational forms, depicted in Figure 12. The senior management, SM, presents the middle line part as described by Mintzberg. F represents the various functional departments and P represents the projects. Type A is the pure functional form, in which the operating core is divided into different functionalities. In case of an IT department, this could be for example; designers, programmers and testers. Type B is a functionally oriented matrix, with weak project coordination. Type C is a balanced matrix with stronger project management authority. Type D is a project matrix, where project managers are of equal status to functional managers. Type E is called here a ‘project-led organization’, in which the needs of projects outweigh the functional influence on decision-making and representation to senior management, but some coordination across project lines occurs. Finally, type F is the pure project based organization.

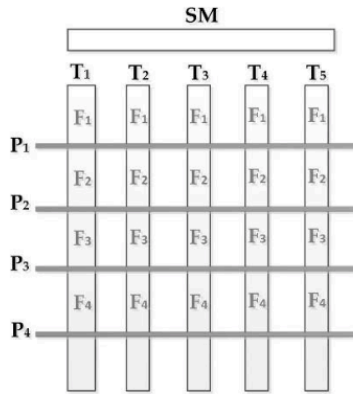
**Key:**

- * $F_1 - F_5$ = various functional departments of the organisation (eg Marketing, Finance, Human Resources, Engineering, Manufacturing, R&D)
- * $P_1 - P_5$ = major projects within the organisation (eg CoPS projects)
- * SM = senior management

Figure 12- Six ideal-type organizational forms. (A) Functional, (B) functional matrix, (C) balanced matrix, (D) project matrix, (E) project-led organization, (F) project-based organization (Hobday, 2000, p. 877)

A pure project based organization is a separate organization, created especially for and singularly devoted to achievement of the project goals. Such an organization can be placed within or outside the company (Nicholas & Steyn, 2017, p. 491). A company can also choose to fully *outsource* a project to another company. This research will focus on software delivery that is primarily conducted by an organization's own IT department. Possibly, some activities of such a project are outsourced to another organization, e.g. contractor or supplier.

An IT department working agile does not fit the organizational forms as described above. An agile team is already cross-functional and consists of employees from different functional departments that are brought together to form a permanent work structure. In contrast with a project-based organization where different functions are brought together for the duration of the project. Based on her research findings, Verbruggen (2017) proposed the organizational form within an agile IT department, as depicted in Figure 13. This form is based on cross-functional teams (T) as a fixed entity. These teams are part of cross-functional value streams, and a "project" can use various value streams in order to meet its goal.



T = cross-functional teams
 FM = various functional departments
 P = major projects
 SM = Senior management

Figure 13- Organizational form in an agile enterprise (Verbruggen, 2017, p. 125)

Interaction between IT department and business

The interaction between the IT department and business can be denoted in many ways. Mintzberg (1983, pp. 19–23) described the functioning of an organization in five different ways. Most obvious seems the flow of formal authority that flows from the strategic apex through the middle line towards the operating core. This formal line of authority will be used to coordinate and communicate regarding subjects like planning, resources, requirements and progress. The interface between the IT department and the business will in practice most likely be effectuated by the communication between a middle line manager and the development teams.

Outside the line of formal authority, interaction between the IT department and the business might take place through the flow of informal communication, the flow of an ad hoc decision process or the set of work constellations. This implies that the interface between IT and business cannot be simplified to only the interaction between a middle line manager and the development team.

3.2 Interaction between agile software delivery and the organizational context

This section focuses on agile software delivery and how this relates with its surrounding organization by conducting an in-depth review of existing literature to find aspects that explain the interaction. As discussed in Section 3.1.4, agile software delivery is conducted by the IT department and the organizational context is interpreted as the business. The outcome of this section will give an answer to the **second sub-question**:

Which aspects are described in literature as being relevant for the interaction between agile software delivery and the organizational context?

Section 3.2.1 discusses the method that is chosen to collect relevant findings out of the existing literature. Next, Section 3.2.2 describes the literature that has been studied and in what aspects this resulted. Lastly, Section 3.2.3 shows a compilation of the aspects.

3.2.1 Method of gathering aspects

Firstly, a framework is selected to structure the findings of the literature study, i.e. categorize the aspects. Therefore, multiple organizational models and frameworks were considered that are utilized to portray an organizational design. A few of these optional frameworks are briefly discussed.

A significant contribution to organization theory was made by Henry Mintzberg with his book *Structure in fives: designing effective organizations* (Mintzberg, 1983), in which a few frameworks are suggested to design an organization, including the five coordinating mechanisms, the five basic parts of an organization and five views of how the organization functions. The book *Organization Theory and Design* (Daft, 2004) describes the structural dimensions of an organization to describe the internal characteristics, these include formalization, specialization, hierarchy of authority, centralization, professionalism and personnel ratios (Daft, 2004, pp. 17–18). In the book *Organization Theory, Concepts and Cases* (Robbins & Barnwell, 2006), the authors list several dimensions that can be used to compare organizations, such as complexity, differentiation, formalisation, centralisation and coordination.

The frameworks were assessed on their suitability, resulting in the selection of *McKinsey 7s model*. This model is expected to be the most comprehensive framework, since its elements showed a better connection to the initial findings in literature compared to the other models.

The 7s model was developed in 1980s by three McKinsey consultants. The goal of the model was to show how 7 elements of the company: Structure, Strategy, Skills, Staff, Style, Systems, and Shared values, can be aligned together to achieve effectiveness in a company. The key point of the model is that all the seven areas are interconnected and a change in one area requires change in the rest of a firm for it to function effectively, see Figure 14. The Soft Ss include the elements that present an emphasis on human resources, where the Hard Ss represent the more tangible elements of an organization (Ravanfar, 2015). Table 2 presents an explanation of each element. Later organizational frameworks showed great resemblance with this model, such as the ESH-model (Wijnen, Weggeman, & Kor, 1989, p. 18).

The 7s model provides more guidance during the literature study, since it steers the literature search in a few direction and provides directions what to look for.

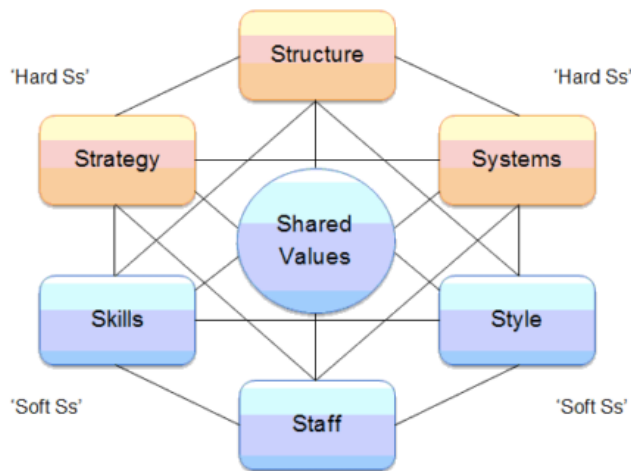


Figure 14- Representation of McKinsey 7s model (Ravanfar, 2015)

Strategy	Is a plan developed by a firm to achieve sustained competitive advantage and successfully compete in the market.
Structure	Represents the way business divisions and units are organized and includes the information of who is accountable to whom. In other words, structure is the organizational chart of the firm.
Systems	Are the processes and procedures of the company, which reveal business' daily activities and how decisions are made. Systems are the area of the firm that determines how business is done and it should be the main focus for managers during organizational change.
Skills	Are the abilities that firm's employees perform very well. They also include capabilities and competences.
Staff	This element is concerned with what type and how many employees an organization will need and how they will be recruited, trained, motivated and rewarded.
Style	Represents the way the company is managed by top-level managers, how they interact, what actions do they take and their symbolic value. I.e. Its management style of company's leaders.
Shared values	Are at the core of McKinsey 7s model. They are the norms and standards that guide employee behavior and company actions and thus, are the foundation of every organization.

Table 2- Explanation of the elements of McKinsey 7s model (Ravanfar, 2015)

3.2.2 Gathering of aspects describing the interaction

An explorative literature study has been conducted to trace aspects that are mentioned to express the interaction between agile delivery and the business.

Several scholars scrutinized certain barriers or success factors that could be distinguished when implementing agile into an existing organization. Nerur et al (2005) conducted a literature study into the search of several challenges of adopting agile methodologies. Boehm & Turner (2005) received input from workshop participants, including agile and traditional developers, and presented barriers and management challenges when implementing agile processes in traditional development organizations.

Other research results are based on empirical findings. Cohn & Ford (2003) have introduced Scrum to seven different organizations and identified approaches for successfully introducing

agile processes to organizations. Tolfo & Wazlawick (2008) conducted empirical observations with six software companies to identify aspects of organizational culture that may influence the use of extreme programming, i.e. XP. Another paper investigates the experiences from integrating agile teams in traditional project management models, based on two cases, both within large system development companies, comprising both software and hardware development (Karlström & Runeson, 2006). More recent research is done by Van Waardenburg & Van Vliet (2013) to find out which challenges the co-existence of agile methods and plan-driven development bring, and how organizations deal with those challenges, based on a Grounded Theory research involving 21 agile practitioners from two large enterprise organizations in the Netherlands.

The relevant aspects from this literature are discussed according to the elements of the 7s model as described in Section 3.2.1.

Strategy

The literature study did not result in any aspects that can be related to the element Strategy. In itself this can be considered remarkable, since several scholars already denoted a relationship between strategy and the structure of an organization. The principles of this line of reasoning are described by Chandler in his book *Strategy and Structure* (1962) and have been the subject of a number of conceptual and empirical studies. This includes research by Hall & Saias (1980) who argue that strategy follows structure. Based on this research it seems evident that strategy will be affected when an organization adapts agile methods. Nevertheless, no relevant findings result from the literature study regarding the aspect 'strategy' and will therefore not further be scrutinized in this research.

Systems

Many of the findings are related to systems, which can be explained by the fact that this is described as the area of the firm that determines how business is done (Ravanfar, 2015).

Documentation

One of the values debated in the Agile Manifesto is working software over comprehensive documentation (Fowler & Highsmith, 2001). Although this doesn't imply that documentation should be fully abandoned, it seems to be an aspect to take into account when implementing agile approaches in an organization. Much of the knowledge in agile development is tacit and resides in the heads of the development teams (Nerur et al., 2005, p. 76) or was captured in informal documents. Although these informal documents often contain much of the information required, this is not always realized by the business. (Karlström & Runeson, 2006, p. 216). As a result, the documentation that the business requires isn't a natural output of agile methods (Barry Boehm & Turner, 2005; Van Waardenburg & Van Vliet, 2013).

Decision-making

In agile, the development team and the customer make most of the decisions, creating a pluralist decision-making environment due to the diverse backgrounds, attitudes, goals and cognitive dispositions of team members. Decision-making in this environment is more difficult compared to the traditional approach where the project manager is responsible for most decisions. It may take an organization enormous effort, time, and patience to build a culture of trust and respect to facilitate such collaborative decision-making (Nerur et al., 2005, p. 76). Next to that, teams using agile processes tend to make decisions more quickly than plan-driven teams, relying on more frequent (and usually informal) communication to

support this pace (Cohn & Ford, 2003, p. 75). A discrepancy might emerge if the perspective regarding decision-making differs between the business and the development team.

Planning and control systems

Tension seems to exist between the business and agile development teams when it comes to planning and control systems. This tension can be explained by the fact that many organizations are based on traditional processes that are compliance-driven and activities- and measurement-based, aimed at providing assurance. Agile methodologies however rely on speculation, or planning with the understanding that everything is uncertain (Nerur et al., 2005, p. 77). This misfit might result in negative consequences, such as a prefixed scope, problems with requirement gathering and prioritization, slow reaction to change and limited feedback from the business (Van Waardenburg & Van Vliet, 2013, pp. 2163–2167).

Plan-driven processes still appeal to many upper managers because they facilitate progress tracking (Cohn & Ford, 2003, p. 77). Nevertheless, traditional contracts, milestones, and progress measurement techniques might be inadequate to support agile processes' rapid pace. Traditional earned-value processes are difficult if not impossible to apply to agile work because of work breakdown structure inadequacies and the flexibility time boxing requires (Barry Boehm & Turner, 2005, p. 34). Abandoning plan-driven processes could result in a perceived lost of control by management, since they do not recognise the usual planning models and cannot track on-going work accordingly (Karlström & Runeson, 2006, p. 219). Obviously, abandoning the old control mechanisms without introduction a new one will result in a real loss of control. Solutions are offered by using scaled agile frameworks.

The Scrum Guide stretches the ability to control progress with two of its three pillars; transparency and inspection (Schwaber & Sutherland, 2017, p. 5). Scrum aims to make progress transparent by using techniques such as burn-downs, burn-ups or cumulative flows. These tools could facilitate control, if used and understood correctly by executive management (Barton et al., 2005).

Resource management

The inverted triangle of constraints, discussed in Section 3.1.2, implies that agile methods allow scope to vary within the fixed parameters of cost and time. Obviously, most managers are comfortable with a model in which project budgets are approved and the project remains within the budget confines. But they are less comfortable when told that project iterations will persist as long as the customer or a customer proxy continues to identify high-priority, high-value work (Cohn & Ford, 2003, p. 77). Karlström (2006) indicates that within agile teams, engineering resources were more efficiently allocated to tasks, but also recognized a crucial point on the start of full-scale development, where it is decided what resources are allocated. Also Boehm (2005, p. 34) points out a difference between agile and traditional processes when it comes to resource loading and slack calculations.

The key difference in the perspective on resource allocation is explained by SAFe as 'agile brings work to the team instead of teams to the work' (SAFe®, 2017). Traditionally people were gathered from different departments of an organization to form a project team around a project. Agile is based on permanent, multifunctional teams that are assigned to work.

Little evidence can be found in literature about how this difference is perceived in practice. Obviously, allocation of human resources has a strong relation to the elements skills and staff, which are discussed below.

Budgeting and cost accounting

Organizations using agile software development methods do experience budgeting and financial control as a restriction. This problem is addressed in the whitepaper of Sirkiä & Laanti (2017), which is later adopted by SAFe 4.0 (Knaster & Leffingwell, 2017). Sirkiä & Laanti argue that traditional budgeting and cost reporting is a system based on rigid-frames, which burdens the agile enterprise with unnecessary and counterproductive overhead and friction. This friction is assigned to different reasons. For example, traditional cost accounting expects a long horizon with detailed and long horizon planning, but agile tries to avoid this. Another example is that traditional budgeting draws attention to budget overruns, whereas in agile development, if the initial feedback proves to be positive, further investment is allowed or even encouraged (Sirkiä & Laanti, 2017, p. 3).

Structure

Structure represents the way business divisions and units are organized and includes the information of who is accountable to whom (Ravanfar, 2015). The relating aspects found in literature can be assigned to mutual adjustment.

Mutual adjustment

Mintzberg (1983, p. 4) considered mutual adjustment as the control of work rests in the hands of the doers. In this case it is considered as the interaction and coordination between subunits within an organization. Managing variability in subsystems and teams has proven difficult. If both agile and traditional teams are developing software for the same product, they can develop radically different artefacts that might not integrate easily. Without some means of coordination, an agile team's assumptions and choices could vary significantly from other developer's counterpart's assumptions (Barry Boehm & Turner, 2005, p. 31). When introducing an agile process into an organization, upper management must understand and agree on how this will impact groups outside the development group (Cohn & Ford, 2003, p. 77). Different approaches poses threats to communication (Van Waardenburg & Van Vliet, 2013, p. 2159) or a mismatch in work synchronization of co-dependent teams, i.e. iteration frequency (Karlström & Runeson, 2006, p. 218).

Skills and staff

Skills are the abilities that firm's employees perform very well, including capabilities and competences. During organizational change, for example introducing an agile approach, the question often arises of what skills the company will really need to reinforce its new strategy or new structure. Staff is concerned with what type and how many employees an organization will need and how they will be recruited, trained, motivated and rewarded (Ravanfar, 2015). A significant overlap can be found between these two elements. Aspects that are considered to have a relevance to both skills and staff are changing roles, performance and reward systems, and staffing plan.

Division of tasks and responsibilities

Verbruggen (2017) explored the changing role of the project manager within an agile environment. This change is established by other research. Nerur et al (2005) plead that the project manager's traditional role of planner and controller must be altered to that of a facilitator who directs and coordinates the collaborative efforts of those involved in development. Van Waardenburg & Van Vliet (2013) also argue that the project manager is one of the first persons that feel change. Based on this change, one could say that special

attention should be paid to the division of tasks and responsibilities when using an agile approach. Revision of jobs and roles should be considered. Nevertheless, little can be found in literature about how this changing role affects other roles and how organizations cope with this change.

Performance and reward systems

Agile development relies on teamwork, as opposed to individual role assignment that characterizes traditional development. Performance measurement and reward systems, therefore, must be suitably designed for successful adoption of agile methodologies (Nerur et al., 2005, p. 76). Organizations must learn to accommodate team-oriented versus individual rewards (Barry Boehm & Turner, 2005, p. 34).

Staffing plan

Nerur et al state that there is little evidence to suggest that agile principles will work in the absence of competent and above-average people (2005, p. 76). Bolder formulated, one could say that agile doesn't work in the absence of skilled people. This can pose serious problems related to staffing. When fully engaged and comfortable with an agile process, a development team moves very quickly. Too many slow workers either slow the entire team or end up left behind by their faster colleagues (Cohn & Ford, 2003). Organizations must learn to accommodate human resource issues such as position description and required skills. Agile development team members often cross the boundaries between standard development position descriptions and might require significantly more skills and experience to adequately perform (Barry Boehm & Turner, 2005, p. 34). Lack of resources to hire competent and skilled personal create unfavourable aspects to the adaptation of agile (Tolfo & Wazlawick, 2008, p. 1960).

Style

Style represents the way the company is managed by top-level managers, how they interact, what actions do they take and their symbolic value (Ravanfar, 2015). The obtained aspect from literature is named management style.

Management style

Agile methodologies require a shift from command-and-control management to leadership-and-collaboration. The organizational form that facilitates this shift needs the right blend of autonomy and cooperation to achieve the advantages of synergy while providing flexibility and responsiveness (Nerur et al., 2005, p. 76). Many managers, particularly those at higher levels, are reluctant to surrender the feeling of control that Gantt charts and other plan-driven process artefacts give them (Cohn & Ford, 2003, p. 76). Highsmith (2012) describes the need for creative leadership, which includes embracing ambiguity, taking risks that disrupt the status quo, instituting new management styles, and faster decision-making.

Shared values

Shared values are the core of McKinsey 7s model, being the norms and standards that guide employee behaviour and company actions and thus, are the foundation of every organization (Ravanfar, 2015). In important aspect found in literature is the organizational culture.

Organizational culture

The main challenge to overcome when introducing any changes is the inherent resistance to change (Barry Boehm & Turner, 2005; Heidenberg, Matinlassi, Pikkarainen, Hirkman, & Partanen, 2010; Karlström & Runeson, 2006). Some companies are not very dedicated to adapting to the market and new technologies, attributing more value to predictability and security already obtained through techniques that have been successful at the company with time (Tolfo & Wazlawick, 2008, p. 1965).

The values, norms, and assumptions of an organization are stabilized and reinforced over time, and are reflected in the policies embodied in organizational routines. Culture exerts considerable influence on decision-making processes, problem-solving strategies, innovative practices, information filtering, social negotiations, relationships, and planning and control mechanisms (Nerur et al., 2005).

3.2.3 Compilation of aspects

The results of the literature study are presented in Table 3. The table shows a compilation of aspects that are relevant to an organization when utilizing an agile approach. The results of the literature review show that a significant amount of the aspects can be related to systems that are used within an organization.

COMPILATION OF ASPECTS DERIVED FROM LITERATURE		SOURCES
Systems	<i>Documentation</i>	<i>Nerur et al., 2005; Karlström & Runeson, 2006; Boehm & Turner, 2005; Van Waardenburg & Van Vliet, 2013</i>
	<i>Decision-making</i>	<i>Nerur et al., 2005; Cohn & Ford, 2003</i>
	<i>Planning and control systems</i>	<i>Nerur et al., 2005; Van Waardenburg & Van Vliet, 2013; Cohn & Ford, 2003; Boehm & Turner, 2005; Karlström & Runeson, 2006, Barton et al., 2005</i>
	<i>Resource management</i>	<i>Cohn & Ford, 2003; Karlström & Runeson, 2006; Boehm & Turner, 2005; SAFe®, 2017</i>
	<i>Budgeting and cost accounting</i>	<i>Sirkiä & Laanti, 2017; Knaster & Leffingwell, 2017</i>
Structure	<i>Mutual adjustment</i>	<i>Boehm & Turner, 2005; Cohn & Ford, 2003; Van Waardenburg & Van Vliet, 2013; Karlström & Runeson, 2006</i>
Skills & Staff	<i>Division of tasks and responsibilities</i>	<i>Verbruggen, 2017; Nerur et al., 2005; Van Waardenburg & Van Vliet, 2013</i>
	<i>Performance and reward systems</i>	<i>Nerur et al., 2005; Boehm & Turner, 2005</i>
	<i>Staffing plan</i>	<i>Nerur et al., 2005; Cohn & Ford, 2003; Tolfo & Wazlawick, 2008</i>
Style	<i>Management style</i>	<i>Nerur et al., 2005; Cohn & Ford, 2003</i>
Shared values	<i>Organizational culture</i>	<i>Boehm & Turner, 2005; Heidenberg et al., 2010; Karlström & Runeson, 2006; Tolfo & Wazlawick, 2008; Nerur et al., 2005</i>

Table 3- Aspects being relevant to an organization when using an agile approach and the originating sources (own table)

3.3 Theoretical proposition

This section aims to explain how the findings of the previous two sections can be used as input to the multiple-case study. The essence of the literature results is framed as a theoretical proposition that will directly serve as input to the interview protocol.

Section 3.1 described the context and is used to select appropriate cases and understand the units of analysis. An organization conducting software delivery is conceptualized into two entities; the IT department and the business. Focus during the case study will be on the interface between these two entities and will determine the type of participants that are required.

The outcome of Section 3.2 gives direct input to the qualitative case study. According to literature, the aspects as shown in Table 3 are of importance to an organization when using an agile method in software delivery. The following theoretical proposition will guide the multiple-case study:

The success of agile software delivery in an organization is affected by the adjustment of:












1	Documentation	
2	Decision-making	
3	Planning and control systems	
4	Resource management	
5	Budgeting and cost accounting	
6	Mutual adjustment	
7	Division of tasks and responsibilities	
8	Performance and reward systems	
9	Staffing plan	
10	Management style	
11	Organizational culture	

Table 4- Aspects obtained from literature that affect the success of agile software delivery in an organization (own table)

The multiple-case study strives to verify this proposition. As a result, an appreciation can be obtained about the relevance of each aspect in relation to the proposition and the list of aspects can possibly be extended.

As mentioned earlier, some aspects show a clear overlap with other aspects. Observations from practice could therefore be assigned to different aspects and are in that way subdued to a level of subjectivity. Therefore it is of importance that the framework in Table 4 won't be used too rigid, but rather used as guidance during the multiple-case study.

4. CASE STUDY APPROACH

This chapter describes how the multiple-case study is designed and how the data is collected in practice. Firstly, Section 4.1 defines the units of analysis and introduces the cases that have been selected. Section 4.2 presents the case study protocol and in that way explains how the data will be collected. Lastly, Section 4.3 presents four principles that are safeguarded during the collection of data. The results of the multiple-case study will be presented and discussed in the following chapters and in that way give an answer to the fourth sub-question.

4.1 Case selection

An indispensable step in preparation of the multiple-case study is defining the units of analysis and bounding the cases (Yin, 2014, pp. 31–34), resulting in the case selection criteria. The research scope as presented in Chapter 2 and the literature study findings of Chapter 3, resulted in the case study criteria as presented in Table 5.

Each case includes a different organization that delivers software to the business and applies agile methods on the level of software delivery. To gain an impression of how software delivery interacts with its organizational context in practice, the amount of different cases should not be too limited. Given the available time, an optimum is found by investigating 8 different organizations. For each case two participants will be interviewed to compare different perspectives and limit the change of preconception.

Case selection is based on the theoretical sampling method (Eisenhardt, 1989), since cases were purposely selected on particular similarities on one hand and specific differences on the other hand. The primary selection criterion for each case is the execution of software delivery according to an agile approach. Next to that, in each case the software delivery is conducted within the own organization itself, or at most, the own organization has full control over the work that is outsourced to an external party. As denoted by selection criterion number 4, the portfolio should include cases that vary in the level of agility within the organizations. This could include cases that have implemented an agile approach throughout the whole organization, unto organizations that have limited the introduction of agile only up to the level of software delivery.

To generalize the outcome of the multiple-case study, the case selection is not bounded to a specific sector. Therefore, cases are selected from different sectors, including both private and public organizations, to acquire a general group of representatives (Graziano & Raulin, 2007, p. 137). Next to that, interviewing individuals who reflect different perspectives will enhance the credibility of the findings (Rubin & Rubin, 2005, p. 67). Ideally, for each case one interviewee is directly involved in the agile software delivery and the other interviewee is working on a different location within the same organization.

Lastly, the organizations are willing and able to be sufficiently transparent and in that way provide the required information and documentation. Next to that, the interviewees do have sufficient experience and knowledge regarding the selected case to provide the required data.

CASE SELECTION CRITERIA
<u>Individual case</u> <ol style="list-style-type: none"> 1. Agile software delivery. 2. Delivery is not (fully) outsourced, but is conducted or directed by in-house (IT) department. 3. Working with multiple agile teams.
<u>Case portfolio</u> <ol style="list-style-type: none"> 4. Case portfolio contains organizations with different levels of agility. 5. Case portfolio includes organizations in different sectors, both private and public.
<u>Practical</u> <ol style="list-style-type: none"> 6. At least two persons are willing to participate per case and are involved in different ways in the software delivery. In that way illustrating different perspectives within the organization, e.g. IT department and business. 7. The organization is willing to share the required information and documentation. 8. Interviewees are experienced and knowledgeable in relation to the case.

Table 5- Case selection criteria (own table)

The variety of projects available within KWD has been screened on these selection criteria and extended with organizations that are not part of the KWD portfolio. This has resulted in 8 cases as presented in Table 6. For each case this table shows a description of the organization, the software product that is being delivered and what type of agile method is utilized. The maximum agile level denotes the level within the organization in which the agile way of working is embedded. The roles of the interviewees are mentioned in the last column.

#	ORGANIZATION	SOFTWARE PRODUCT	AGILE METHOD	MAXIMUM AGILE LEVEL	ROLES OF PARTICIPANTS
1	Publisher (private)	Content Management System (CMS)	Scrum	Multiple teams in same department	1. IT manager 2. Scrum master
2	Public transport company (semi-public)	Logistic planning tool	Scrum	Multiple teams in different IT units	1. Project manager 2. Product owner
3	Legal assistance authority (public)	Renewal of IT landscape	Scrum	Enhanced team	1. Project manager 2. Scrum master
4	Ministry of Defence (public)	Tactical command and control application	Scrum	Multiple teams in same department	1. Project manager 2. Team leader
5	Law system (public)	Digitalization and automation of procedures	Scrum	Multiple teams in different IT units	1. Scrum master 2. Product owner
6	Finance and assurance company (private)	DevOps enterprise datawarehouse	Scrum, DevOps	Multiple teams in different IT units	1. Product owner 2. Platform lead
7	Airline company (private)	DevOps business application	Scrum and SAFe	Multiple teams in different IT units	1. Agile coach 2. Product manager
8	Transport company and delivery service (private)	DevOps business and customer applications	Scrum, Kanban, aspects of Nexus	Multiple teams in same department	1. Head of architecture 2. Product owner

Table 6- Overview of selected cases (own table)

4.2 Case study protocol

The main effort of the multiple-case study is the execution of an in-depth, semi-structured interview with each participant. In preparation of these interviews, a case study protocol is designed to ensure that each interview is conducted following the same approach, by describing the questions, procedures and general rules to be followed. The protocol is a major way of increasing the reliability of case study research and is intended to guide the researcher in carrying out the data collection from each single case (Yin, 2014, p. 84).

Before each interview, an email is sent to each participant to confirm the time and location of the interview. Next to that, each participant is requested to deliver some general information regarding the background of the interviewee, the organization and the software delivery. By collecting this basic information in advance of the interview, a first impression is obtained about the software delivery and the organization. This information can be verified during the interview for clarification or getting more in-depth understanding.

Each participant is subjected to an interview following the same structure, see Appendix A, *Interview structure*. The interview is divided into several subjects, which are connected in a logical order and give structure to the interview. The interview structure contains a combination of main questions and follow-up questions. The main questions help to make sure that the required information is obtained; the follow-up questions ensure that the right level of depth, detail, vividness, richness and nuance is achieved (Rubin & Rubin, 2005, pp. 129–151).

The outcome of each interview is reported in an *interview report*, containing all relevant information that has been discussed during the interview. The interview report is sent to the interviewee for verification and validation of the content. Feedback on the content is processed, resulting in a single case report.

4.3 Data collection

Yin describes four principles of data collection that should be followed to deal with the problems of establishing the construct validity and reliability of the evidence (Yin, 2014, pp. 118–129). This section considers how these four principles are ensured in this research. Figure 15 depicts a schematic presentation of the data collection and the chain of evidence to support the explanation below.

Principle 1: Use multiple sources of evidence

An important advantage presented by using multiple sources of evidence is the development of *converging lines of inquiry*. In this way, the case study findings are supported by more than a single source of evidence and increase the reliability. Figure 15 includes the primary sources of evidence that are transferred to an individual interview report, namely the notes taken during the interview, the voice records and relevant documents provided by the interviewee. Additionally, each single case is based on the input of two different interviewees, in that way presenting different perspectives on the same topic and strives to reduce the subjectivity by comparing the different views.

Principle 2: Create a case study database

The second principle is related to the organization and documentation of the data collected for the case studies. Within this research, the documentation can roughly be divided into two separate collections, namely the data, without any interpretation, and the reports in which the collected data is interpreted by the interviewer. The data is recorded in the notes, voice

records and the provided documents. The first interpretation of this data is presented in the individual interview reports, which are validated by the interviewees on completeness and correctness. Subsequently, a case report describes each single case, based on an interpretation of two different interviews. The total assembly of data and reports forms the database that is subjected to the cross-case analysis.

Principle 3: Maintain a chain of evidence

Maintaining a chain of evidence increases the reliability of the information in the multiple-case study. This principle allows an external observer to follow the derivation of any evidence from the initial research questions to ultimate case study conclusions. The next chapter shows how the literature study results give input to the interview protocol. The data collected during the interview, i.e. notes and voice records, follow the same structure as the interview protocol and is presented in the same order in the individual interview report. The single case report further interprets the findings, maintaining a relation to the interview structure. The conclusions that follow from the cross-case analysis can in that way be linked to the initial research questions.

Principle 4: Exercise care when using data from electronic sources

The last principle suggests exercising care when data from electronic sources is used. This principle seems to be irrelevant to this research, since no electronic sources are used as its own subject of study.

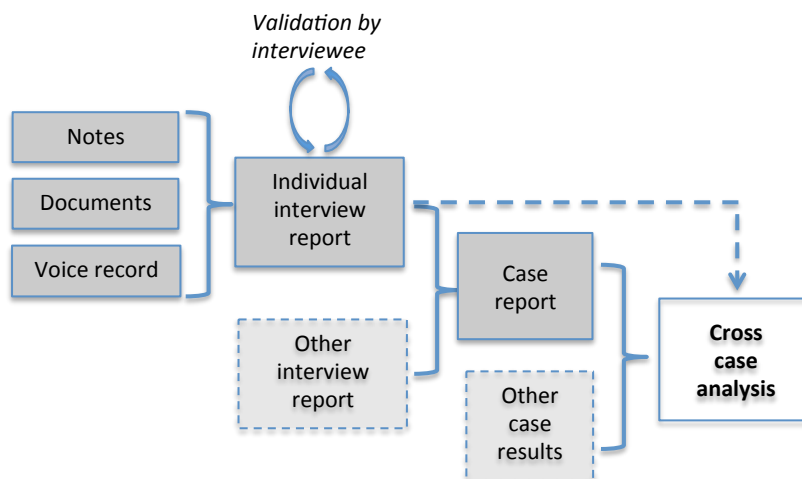


Figure 15- Data collection and chain of evidence (own illustration)

5. SINGLE CASES

The focal point of this research is the performance of the multiple-case study. The interviews that have been conducted at different organizations strive to answer the **third sub-question**:

In what way does agile software delivery interact with its organizational context in practice?

This chapter presents each case that has been selected in accordance with the selection criteria. Each section discusses a different organization by giving an introduction, general description of the organization and an explanation of the software delivery method. Next to that, some of the interview results will be emphasized that are relevant to the eleven aspects and give a concise illustration of how agile software delivery interacts with the organizational context.

Detailed information of each case is available in the interview reports, which are not included in this thesis. The data in these interview reports will also form the basis for conducting the cross-case analysis, as presented in the next chapter.

5.1 Case 1: Publisher- Development of frontend functionalities using Scrum

Introduction of organization and software delivery

This private organization consists of circa 650 employees and is a subsidiary of an international company. The organization can be described as a multimedia publisher of professional information and is divided into several business units with each its own information area. Next to publishing of books and magazines, most of the business units have a large stake in provision of online information, which is facilitated by the IT department.

The IT department consist of circa 25 to 30 employees and is responsible to facilitate the IT functionalities in accordance with the business strategy. Scrum has been introduced circa two years ago and is being used to develop all frontend functionalities. The frontend team is grouped into 7 permanent Scrum teams, consisting out of 3 to 6 employees per team. Qualified developers are scarce and the teams make use of external personnel. Next to that, personnel from the business are involved, mainly in the role of product owners. A difference can be distinguished between the Scrum teams when it comes to their maturity level of working with Scrum, but in general it seems that the principles are well understood and implemented on team level.

The IT manager, supported by one business analyst, forms the level above the Scrum teams. This is not only the upper level to the Scrum teams, but also the back office and IT services. As a result, the IT manager copes with a large span of control, including the management of the agile frontend teams, but also the management of waterfall projects conducted by other teams. The IT manager reports directly to the Chief Information Officer (CIO) of the company. The IT architecture is at the moment positioned outside the IT department and also reports to the CIO.

The functionalities that are required to be developed by the Scrum teams, find their origin at the business units, which are in this case different publishers. Each business unit report its IT demand to the head publisher. The head publisher, together with the CIO, determines the priority of each demand and communicates this to the IT manager. The IT manager and business analyst examine the request and how this affects the work to be done for the Scrum

teams. Figure 16 shows a schematic presentation of how the agile software delivery is positioned in the organization.

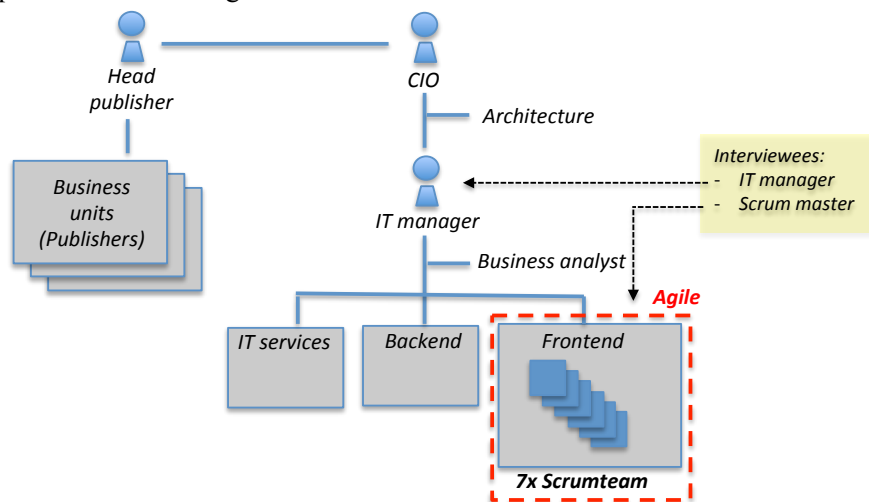


Figure 16- Schematic presentation of case 1 (own illustration)

Interaction software delivery and organization

Both interviewees give the impression that Scrum has been implemented quite well on team level, which has to a large extent to do with the effort of an agile coach for a couple of months. Problems are mostly perceived above team level, where the interaction between the Scrum team and the rest of the organization takes place.

The current division of tasks and responsibilities seems to be unsuitable. As described above, the IT manager has to deal with a wide span of control and is charged with the coordination between the Scrum teams, next to managing the rest of the department. At the moment the adjustment and coordination between the Scrum teams is mainly taken care of by individual team members, but is not an optimal solution, resulting in '*miscommunication and unverified assumptions*'. Next to that, no portfolio manager is appointed, as a result of which the IT manager is also charged with this task. Extra functions seem to be required to deal with the coordination, internal and external, and manage the projects portfolio.

Next to perceived challenges of coordination and integration, the IT manager is not satisfied with the current way of budgeting and cost accounting; the responsibilities are allocated to employees that do not actually manage the budgets, causing impractical situations. Next to that, the current performance and reward systems are not well arranged. Again these aspects can be related to the design of the level above the Scrum teams.

Furthermore, both interviewees point to the fact that not all employees are fully suited to fulfil their tasks, which can be explained in a few ways. The organization has difficulties to find suitable technical personnel, especially developers, which is a problem that is difficult to manage. Another fact is that personnel is assigned according to availability and to a lesser extent on their suitability. For example, during the implementation of Scrum and the corresponding organizational changes, many former project managers were assigned as product owners within a Scrum team. Not only does a new function require other qualities, the same goes for working within an agile environment. This is noticed on the work floor, but also on management level. The interviewees describe some colleagues less suited to work agile, due to their *high level of control* or their *direct management style*.

Apart from points of improvement, many aspects seem to be implemented effectively around the agile software delivery within the organization. A clear process has been established to describe the line of decision-making, which seems to be clear on the different levels within the organization. Although the process is precisely described, the implementation of this process could still be improved in practice. Next to that, the way of planning and progress reporting is effectively arranged, creating a flow within the software delivery.

5.2 Case 2: Public transport company- Replacement of transport planning application

Introduction of organization and software delivery

This case focuses on an organization formed by a partnership between two collaborating organisations in the public transport and logistics sector in the Netherlands. The main purpose of this organization is the replacement of the current IT landscape and the development and operations of the future IT landscape. The project subjected to the case study aims to develop a transport planning software application on a national scale. The project has started approximately 5 years ago and faced many setbacks so far. The current project manager (PM) has been involved for more than a year and is, next to the software development, also charged with setting up a permanent IT organization that can maintain the new application once it is in production. The software application strives to go in production in January 2020, but in practice parts of the application will go in production earlier if possible.

Figure 17 shows a schematic presentation of how the project is positioned in the organization. The organization's leadership consists of two representatives from the two different companies that entered the partnership. The company consists of circa 110 people, of which 80% external personnel, and is divided into four departments. The department *functional application management* and the department *technical application management* are responsible for the maintenance of the IT systems. The staff department supports the other departments and includes administration, finance and a project management office (PMO). This department includes program-, portfolio-, release-, and test management. The project subjected to the case study is positioned in the department *software development*. This department is divided into three groups, working on different software components. Each group consists of multiple Scrum teams and is led by a team manager (TM).

The project uses 5 Scrum teams, of which four are so-called build-and-test teams and one team is responsible for the work preparation. The build-and-test teams consist of circa 4 to 5 persons, one of which also acts as Scrum master. The work preparation team is composed of analysts and designers; each team member also acts as product owner (PO) of one of the build-and-test teams.

The business consists of several planning departments, or stakeholders, that are going to use the software. The PO's have close contact with these future users regarding the functional requirements, etc. Next to that, a fifth PO is assigned, who is officially not part of the project organization, but represents the future users and is closely involved with the project. This PO is named the *delegated PO*.

The organisation as depicted in Figure 17 represents the line organisation, which is not exactly the same as the project organisation. This also affects the project organisation, since a project manager (PM) is responsible for all project-related work and a team manager (TM) is charged with the line management responsibilities. This also implies that different lines of communication do exist; along the 'project-line' and along the 'management-line'.

Both interviewees indicate that Scrum is well implemented at the team level, but the level above the teams is facing several challenges. Recently the organisation restructured the way of working and implemented some changes to improve the coordination above team level. Some of these changes are derived from Scaled Agile Framework, SAFe.

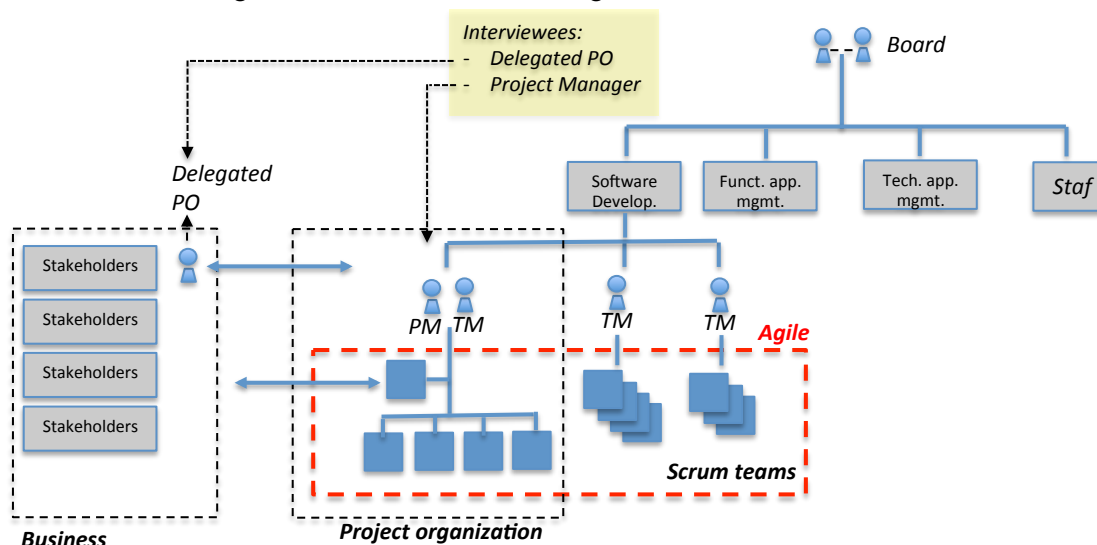


Figure 17- Schematic presentation of case 2 (own illustration)

Interaction software delivery and organization

During the interviews both participants indicate challenges that are perceived on tactical level, which is the level above the Scrum teams. One of the main causes of the perceived challenges is the ambiguity that exists regarding which persons has which mandate. Obvious choices regarding budget and scope changes are made by the steering group. Nevertheless, the project is struggling with several choices regarding technical and functional aspects that have a direct relation with the work to be done by the Scrum teams. In the current situation it is unclear where the decisions should be made for many technical and functional choices. An example that is given by one of the interviewees is the mandate of the business representatives and the IT architects, which is imprecise at the moment. This unclear delegation of mandates slows down the current decision-making process and in that way the flow of work.

Another point of attention is the coordination between different teams and also relates to the tactical level. On project level the Scrum teams work well together and much of the coordination automatically takes place between the teams. Challenges occur regarding dependencies with other projects or departments. The recent organizational restructuring aims to improve the coordination between different teams and departments, but not everyone is fully aware of this new structure. Next to that, one of the interviewees indicates that coordinating roles are not suitable for everyone and the organisation should carefully select a suitable candidate to fulfil these tasks.

Both these challenges, unclear mandates and lack of coordination, can partially be explained by the existence of the project and line organisation, since some tasks and responsibilities are assigned at different levels (e.g. architecture and release management). Next to that, both interviewees describe an informal, loose management style within the organization. Although this style suits agile on team level very well, more control is required on the upper level and

more processes and tools should be imposed to improve the coordination and the collaboration throughout the organization.

Despite the perceived challenges and points for improvement, the future users are positive about the progress and their involvement in the software development. Also other aspects are well implemented in the project. Despite the complexity of the scope, the project manager found an optimum in the way of planning, by combining waterfall with agile. Planning occurs on three different levels; operational, tactical and strategic, which combines looking ahead with flexibility on the details.

5.3 Case 3: Legal assistance authority- Renewal of IT landscape

Introduction of organization and software delivery

The organization subjected to the case study is a public authority that is responsible for ensuring that citizens have access to support for legal problems by the provision of subsidies. The organization initiated a program to fully renew the IT landscape, including its current Customer Relationship Management (CRM) system. Part of this program is the project that is scrutinized in this case study and aims to deliver a future-proof IT landscape for a specific department within the organization.

Figure 18 shows a schematic presentation of the project organization and its organizational context. The development of the new IT landscape is approached as a combined effort between this public authority, as described above, and the permanent IT supplier of the Dutch government. Within the figure the public authority is presented as the business, in which the stakeholders represent the department for which the project is conducted. Both entities have a seat in the steering group, or project board, as the senior supplier and senior user.

The project is initiated in the summer of 2017 and will be finished by the end of 2018. It is the first project within this organization that is executed according to agile. An amplified Scrum team, consisting of circa 15 people, forms the core of the project organization. Technical personnel is delivered by the IT supplier, such as developers and testers, but the team also consists of people from the business. Next to the project manager, the IT supplier appointed a software delivery manager that is responsible for the technical development and delivery of the solution. A challenge within the team is the colocation, since not all team members are fully available to the project and the workplaces differ.

The project is part of a wider program that is directed by a program manager. The project manager reports to the program manager or directly to the steering group. A staff, i.e. information management department, supports the projects, which include architecture and technical maintenance. Once the software is developed, it will be handed over to the support organization.

The project is partially conducted according to a traditional project setup and partially by the use of Scrum. During the first phase of the project the main effort was focussed on a thorough analysis of the business processes and resulted in a functional design. The functional design is subsequently translated into 14 sprints and developed by the use of Scrum.

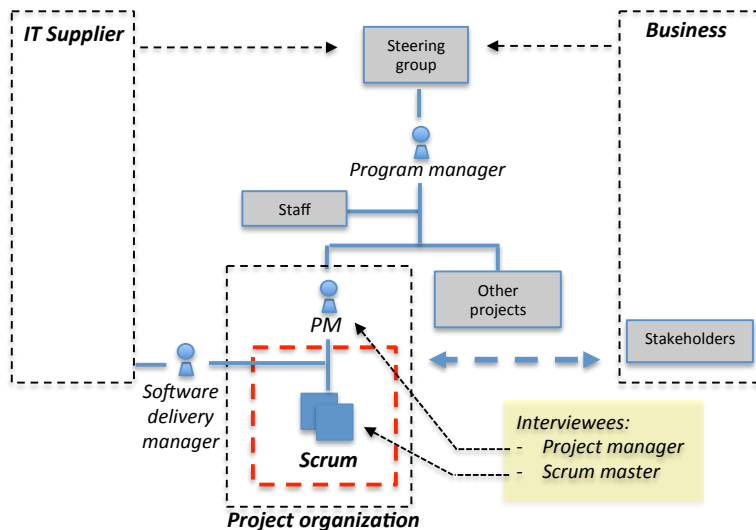


Figure 18- Schematic presentation of case 3 (own illustration)

Interaction software delivery and organization

As mentioned above, the organization had very little experience with agile software development. This lack of agile experience seems to have its affect on the execution, since the speed of the development is significantly lower than expected. Nevertheless, both interviewees emphasize that the business is very positive about the results so far and the collaboration between the project and the business is going very well, which benefits the user adoption.

The scope of the project is clearly defined by the functional design, which has been established by using a traditional project approach. Within this scope, deviations are made regarding the exact requirements. All changes regarding functional aspects are made in close coordination with the business and cause no friction. Changes regarding technical aspects on the other hand, are perceived to cause friction. One of the interviewees point to the fact that the developers are lacking clear guidelines from the staff or IT department. Next to that, the integral coordination with other software projects or teams can be improved. Technical integration is not dealt with sufficiently at the moment.

The current set-up of the project organization brings along its own challenges. The Scrum team is filled with personnel from different organizations. Since not all team members are all fully dedicated to this project and are working from different locations, it was difficult for the team to find a suitable solution in their way of working together. Next to this, the division of tasks and responsibilities between the project manager and software delivery manager seems to be not always clear within the team, which leads to miscommunication once in a while.

Although the organization is quite inexperienced in agile software delivery and facing many challenges, the steering group and business are positive about the results so far. The use of Scrum is customized for the project, in which aspects of a traditional project management approach are combined, for example in planning and budget control. The management level seems to give enough freedom and trust to the project team, but on the other hand one of the interviewees indicates the lacking of technical guidelines that should be provided by the upper level.

5.4 Case 4: Ministry of Defence- Agile development of command and control application

Introduction of organization and software delivery

This case study has been conducted within the Dutch Ministry of Defence (MoD). The department responsible for all IT systems within the MoD consists of ca. 3000 employees and within this department, a subunit of ca. 220 people is responsible for the development and maintenance of all software applications. One of the main efforts of this department is the development of a command and control (C2) application for all ground-based units. This project is started in 2016 and will be finished in the summer of 2018. Figure 19 depicts a schematic presentation of the organization around this project. About 2/3 of the people are permanent staff of the MoD and the other 1/3 is hired for a period of 4 to 5 years. Most of the people are non-military, only a few actual work as a soldier.

About seven years ago the department decided to use agile for its software delivery and adjusted the organization to this way of working. For this project, approximately 8 development teams are working according to Scrum. Each team consists of 6 to 10 people with different functionalities and is directed by a product owner and a team leader. A few of the development teams are dedicated and working fulltime to the project, the others are also involved in other projects.

Above these development teams a project manager has been installed who's responsible for achieving the results and reports to a steering group. The department also conducts other software development projects that are managed by the program manager. The head of the department is also seated in the steering group and has the final say about the allocation of personnel.

The Architecture Support Team, AST, supports the project manager. The AST consists of different disciplines and is responsible for several aspects, such as integration of the teams and products, architecture, coordination with the mandated user and decision-making within the set scope. This group consists of the project manager, a system manager, lead-architects, head of line management, the product owners and the so-named Operational Experts, or OE. The OE are military personnel and are able to give advice to the development teams from a user-perspective.

One unit within the MoD is appointed as a so-called mandated user. This mandated user is intensively involved in the development process and gives input to the functional demands of all future users, by communicating with the PM, product owners and the OE.

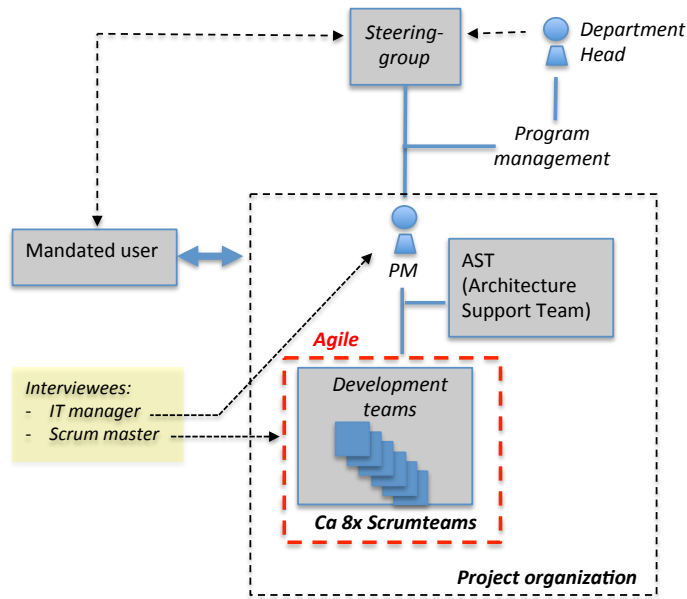


Figure 19- Schematic presentation of case 4 (own illustration)

Interaction software delivery and organization

The department has been reorganized several years ago to arrange the required conditions to implement agile. The development teams are using Scrum *by the book* and above team-level the organization made the essential adjustments to facilitate the teams, although no *scaled agile* method is used. The AST contains the required overhead capacities and in that way improves the coordination and adjustment between different teams and stakeholders. Although most of this coordination is well arranged, the interviewees notice the fact that improvements can be made. For example, when it comes to configuration management.

Both interviewees appoint to the collaboration with the mandated user and how this affects the decision-making. In general the mandates are clearly distributed, in which decisions regarding the scope, costs and time are taken by the steering group. Ambiguity arises when decisions have to be made about functional requirements that are within the scope. A few discrepancies in the current situation are mentioned by the interviewees, such as the presumption that the mandated user doesn't always represent the actual users sufficiently. Next to that, a deviation is made to Scrum *by the book*, since the appointed product owners do not represent the users, but are also IT technicians. As a result, tension does exist between the mandated user and the project organization.

The agile way of working seems to be well understood throughout the project organization. The style of 'management by exception' practiced by the steering group is appropriate and the development teams experience a large degree of freedom in their work. Small tensions are noticed between the project and other parts of the organization. For example, higher management within the organization does not fully embrace agile yet, which leads to resistance towards the project. Another example is that the users are not used to the frequency of the software releases and are having troubles to keep up with the development pace.

Besides some points of attention as discussed above, it seems that the department has implemented agile software development quite well. Relative little friction is experienced by the interviewees regarding the eleven aspects of the interview protocol.

5.5 Case 5: Law system- Digitalization and automation of procedures

Introduction of organization and software delivery

This case study focuses on a public authority that is responsible for the support and facilitation of the Dutch law system. Approximately 5 years ago, a program was initiated, which aims to digitise and automate the organisational procedures. An agile development approach is chosen for the software development. The project subjected to this case study is part of the program.

The project has faced several setbacks over the years, which can be explained by the technical and organizational complexity of the program. The organization had very little experience with working agile, but chose to use Scrum since it was facing an unclear scope and expected to cope with many changes along the way. The project has developed over the years and is now working with Scrum on a wide scale. Next to that, the program is facing severe criticism from society due to the exceedance of the initial budget and planning.

Figure 20 depicts a schematic presentation of the project organization and its organizational context. Five Scrum teams together with one project manager (PM) and one central product owner (PO) are charged with the software development. Four of these teams are build-and-test teams, consisting out of circa 8 to 10 persons per team, of which one Scrum master. The fifth team has a supporting role and consist of more than 15 persons, including a senior Scrum master, designers, acceptance-testers and project support. Within this development group, the product owner is responsible for *what* is made and the project manager about *how* it is made.

The project is faced by a large variety of stakeholders that are represented by the business in Figure 20. A group of circa 20 stakeholders represent the delegated users and are in direct contact with the product owner regarding the exact requirements. This group of delegated users is responsible for the design of the new procedures and also include a pilot group that already started working with the new software. Next to that, other users also have a stake in the project, but are less closely involved. These other stakeholders are represented by so-called portfolio holders that have a seat in the project team. The project manager and product owner are also seated in the project team, which is responsible for the support and decision-making of the project in the long run. This project team also includes implementation management.

The project is part of a wider IT landscape, together with other ICT projects. These other projects are conducted within the organizations ICT department, headed by the ICT board. These projects are also working agile and are formed by several Scrum teams. An ICT staff facilitates all Scrum teams and is responsible for the coordination and integration over the various teams. This includes IT architecture, release management and line managers, who are responsible for the allocation of resources.

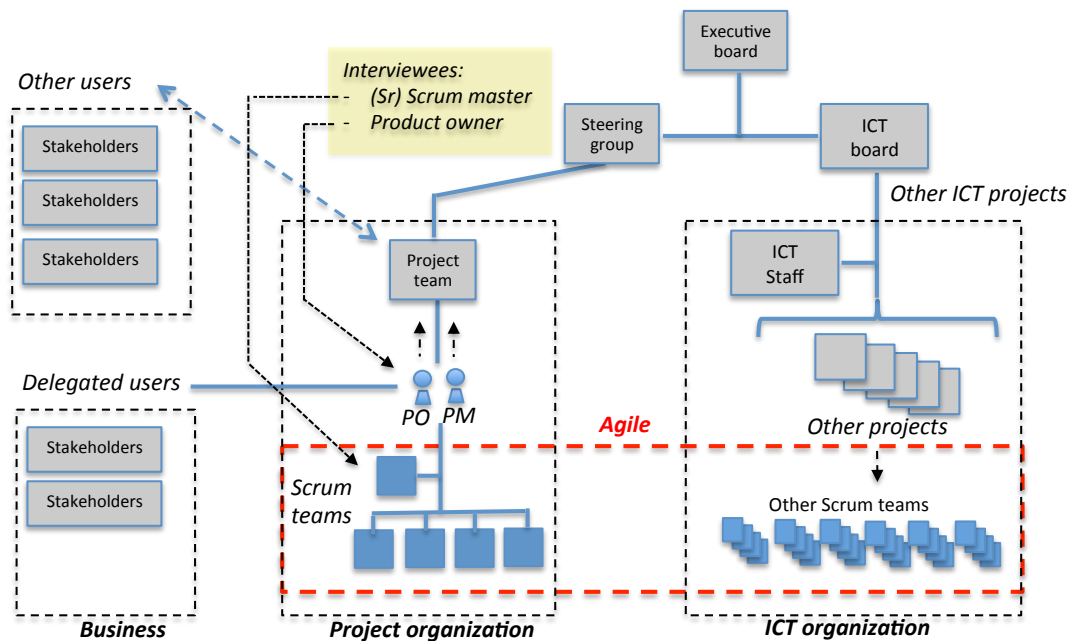


Figure 20- Schematic presentation of case 5 (own illustration)

Interaction software delivery and organization

The project is conducted within a traditional hierarchical organization with the involvement of a wide range of stakeholders. Next to these organizational challenges, the project has started with an unclear scope that significantly expanded along the way. As a result, the initial budget and planning have been largely exceeded.

Despite the project setbacks, both interviewees are very optimistic about the professionalism of the delivery team, i.e. Scrum teams. Scrum is well implemented on team level; the teams are well integrated and are able to deliver high quality within time. According to the interviewees, the problems occur above this delivery level. Which has mainly to do with the organization around the Scrum teams.

Given the amount of agile teams working in the same IT landscape, the program recently considered the use of a scale agile framework, in this case SAFe, to improve the integration and coordination above team level. Nevertheless, the implementation of SAFe was considered to further increase the costs and therefore not realized. Therefore a situation remains in which the delivery teams work agile on one hand, but the upper level is based on PRINCE2, or waterfall, on the other hand. According to the interviewees this affects the interaction between both levels, since managements expectations are not aligned with Scrum, which affects for example decision-making and planning.

One of the main challenges that the project is facing is stakeholder management. A multitude of different stakeholders are directly or indirectly involved with the project and each has its own interest and perspective. Quite recently the product owner and mandated users came together to discuss the functional requirements and frame the exact scope of the development. Nevertheless, debates about the functional requirements appear frequently and affect the work to be done by the Scrum teams. Next to the mandated users, the other users give input to the portfolio holders and influence the functional requirements indirectly. Besides the functional

requirements, the project is affected by the ICT organization regarding technical requirements that are aimed to integrate the project with other projects and the IT landscape.

As a result, decision-making takes more time than necessary and causes friction between participants, which can partially be explained by an unclear division of responsibilities and mandates. Although the division of tasks is clearly allocated within the Scrum teams, the upper level copes with an unclear distribution of responsibilities. Many stakeholders demand to have a voice, but the exact mandates are doubtful. Next to that, both interviewees underline the negative influence of the organizational culture on the decision-making around the project and the effect on the management style, caused by a high formality, lack of decisiveness and fear of handing off control.

5.6 Case 6: Finance and assurance company: DevOps of datawarehousing tool

Introduction of organization and software delivery

This organization is one of the bigger finance and assurance companies within the Netherlands. Within the organization multiple IT departments are placed to provide technical support to the companies primary processes. The IT department that is subjected to this case study is responsible for the development and operations of a large enterprise datawarehousing platform, which is used to collect, analyse and make reports of business data. Figure 21 depicts a schematisation of the department with its surrounding context. The organization has implemented agile approximately 3 to 4 years ago and practice Scrum *'by the book'*.

The department consists of circa 10, co-located, DevOps teams, responsible for the building 'Dev', and running 'Ops' of all the datawarehousing software components. The exact amount of Scrum teams differs frequently, depending on the kind and amount of software solutions that are being developed at a certain moment. Each Scrum team consists of minimal 4 up to maximum 10 software engineers, of which one acts as Scrum master. The department strives to recruit and train multidisciplinary team members. Circa 80% of the IT personnel is external.

A product owner, who is assigned from the business side of the organization, heads each team. The product owner is in direct contact with the business stakeholders that make use of the software products. A selection of stakeholders is appointed that represent the business. These stakeholders communicate their demands to the product owner, who subsequently translates this into a work-prioritisation, i.e. backlog. Most of the Scrum teams are in direct support of a specific business unit, except for two teams, which are tasked to give general support to the platform, such as architecture and maintenance of systems.

A so-called 'core team' is positioned above the multiple Scrum teams. This team consists of circa 5 to 6 people, including a manager, platform lead, HR staff member, functional lead and technical lead. This personnel is not fulltime dedicated to the team, but also perform other tasks within or around the Scrum teams. The core team fulfils an overarching role above the teams and is amongst others responsible for integration between the teams and coordination with other departments.

Both interviewees, the platform lead and one of the product owners, are very optimistic about the current way of working. A big strength that they both emphasize is the level of autonomy of the Scrum teams. All teams have a high degree of freedom and are able to work quite independently.

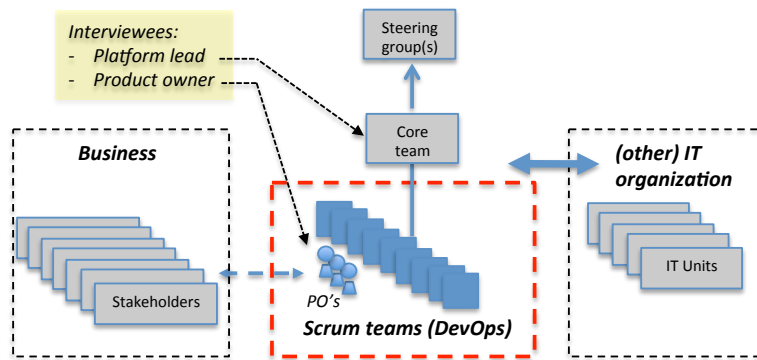


Figure 21- Schematic presentation of case 6 (own illustration)

Interaction software delivery and organization

As mentioned above, the organization strives to give the teams as much autonomy as possible, which reflects in the interview results. The product owner is given a clear mandate regarding the collection and prioritization of the work to be done. The product owner is tasked to funnel all the requirements, both functional and technical, including new initiatives, changes in on-going work, etc. This clear mandate has a positive effect on the decision-making around the Scrum teams.

Another advantage of the high autonomy of the Scrum teams is that little dependencies exist between a Scrum team and other teams or departments. Within the teams, the team members have a clear understanding of the division of tasks and responsibilities and indicate that they don't miss any roles. A disadvantage of the high degree of autonomy is that teams become isolated and are not fully aware of their organizational environment. Integration between teams is therefore stimulated by the organization of frequent meetings and the core team attempts to maintain an integral view across the teams.

Although both interviewees are very optimistic about the way agile is implemented in the teams, they also see room for improvement. This has partly to do with a required change of the organizational culture. The IT departments have well adopted agile, but this differs with the business side of the organization. Also some of the managers tend to hold on to traditional processes. As a result, some of the organizational processes are not optimally designed around agile. Examples mentioned by the interviewees are the inefficient way of budgeting and cost control, but also the demand of progress reports by some of the business stakeholders.

Other points for improvement mentioned by the interviewees is the current way of documenting by the teams, the out-dated job structure and the current way in which individual performance is assessed by the managers. Although these aspects could be improved, it doesn't seem to have a strong negative effect on the success of agile and is indicated by the interviewees as less relevant.

5.7 Case 7: Airline company- DevOps of business applications

Introduction of organization and software delivery

Figure 22 depicts a schematic presentation of the IT department that is subjected to this case study. The IT department is part of a big airline company and is responsible for the development of several front-end software applications. The IT department consists of five development groups that are directly related to one of the primary business processes within the airline company; passenger operations (PaxOps), engineering and maintenance (E&M),

human resources (HR), finance and cargo. These groups can be considered as software factories that continuously develop and release software products to support the operations of specific business units. Two other groups, digital consultancy and digital lab, form a specific branch within the department and don't aim to develop new software, but rather give advice about new software technologies. Both interviewees work within the development group PaxOps, as respectively product manager (PM) and agile coach.

Each software factory is roughly structured in the same way and consists of multiple DevOps teams, each working on their own software product. The DevOps teams have been working Scrum for multiple years and the department started using SAFe circa 1,5 years ago. Each Scrum team consists of circa 10 team members with each their own role, such as developer, tester or business analyst. Each team has a dedicated Scrum master and the team is headed by a product owner, who acts as the primary point of contact for the business.

The IT group PaxOps supports a big business unit that is responsible for all passenger operations on the airport and consists of thousands of employees. Within this big group of stakeholders, a specific group of representatives is appointed, who are closely involved in the software development and give input to the product owner. A business owner, who heads this group of stakeholders, has an influential position in the business and has a final say in case of any disputes.

Above the five Scrum teams within PaxOps, a triangle of three staff-members is positioned that play an overarching role above the teams. The product manager (PM) directs the several product owners and is responsible for the complete roadmap of all software initiatives for passenger operations. Next to that, a Release Train Engineer (RTE) has direct contact with the Scrum masters and is responsible to integrate the development processes and a system architect is responsible for the complete IT landscape in which the teams work. This triangle can also be found on a higher level, so above the development groups, but also on team level, formed by the product owner, Scrum master and architect or lead developer.

This IT department is only responsible for the development and operations of front-end software applications and has several dependencies with other IT units. This way a layered organization exists, in which *Intelligence* is responsible for adding and maintaining of all data and *IT systems* is responsible for the hardware and backend applications. IT units that are working on directly related IT systems are clustered into a domain. Nevertheless, both interviewees point to the fact that the integration within these domains is lacking.

The overarching IT organization consists of the Information Management Organization (IMO) and Information Services (IS), which are also responsible for the allocation of the required resources over the different IT units. Next to that, IMO and IS fulfil a facilitating role to the several IT units in terms of architecture, technical support, etc.

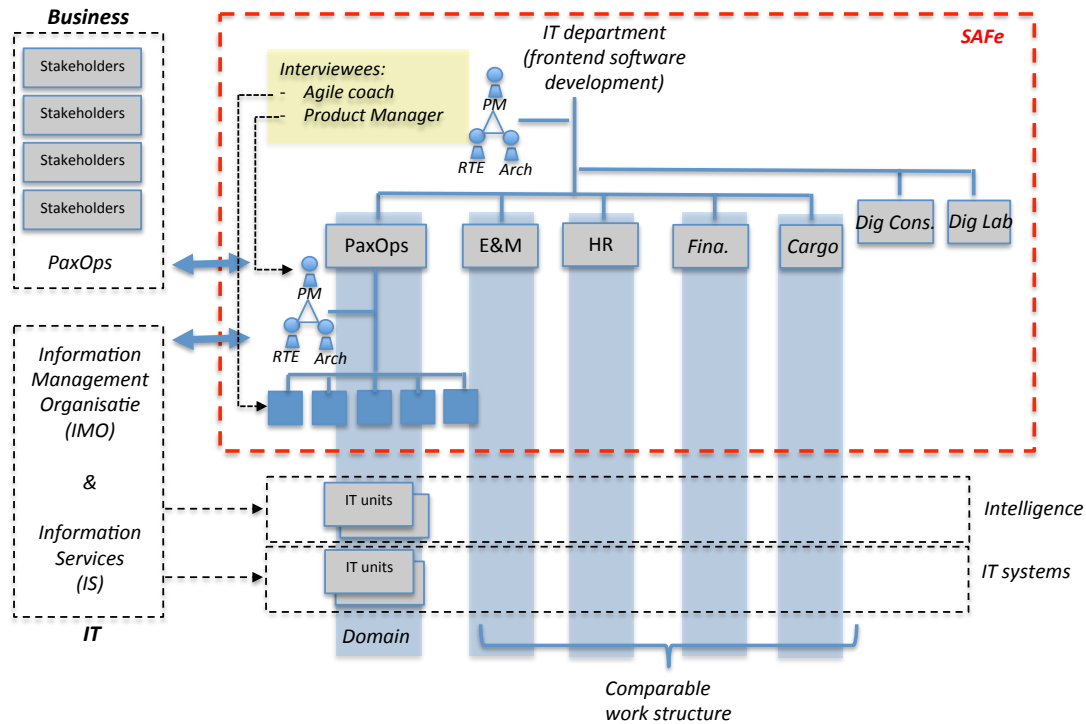


Figure 22- Schematic presentation of case 7 (own illustration)

Interaction software delivery and organization

The IT department that is subjected to the case study started working with SAFe circa 1,5 years ago. Both interviewees are very optimistic about how agile is implemented within the department and how the direct organizational surrounding is designed to support this way of working. SAFe enhanced the integration and coordination between the different development teams working within the department. Next to that, the contact and collaboration with the business is strongly increased. The interviewees describe a clear decision-making process and emphasize a high level of transparency in the current way of working, which benefits this decision-making, but also the progress control of the business and higher management.

On the other hand do the interviewees also clearly state some shortcomings in the current organization around software delivery, which has partly to do with sticking to ‘old-fashioned’ procedures that are not aligned with agile. Most emphasis is placed on the current way of budgeting and cost control. In contrast to agile, the current procedures are very untransparent and the budget authorities are not assigned to the correct people. Also the division of other tasks and responsibilities above the development teams is not well implemented according to the interviewees and ambiguity exists about the exact split between of particular roles.

More problems are perceived in the interaction with IT units in other positions in the organizations. As described above, the development teams working on front-end applications for PaxOps, are depending on hardware, data and backend applications that are developed and operated by other IT units. Whereas SAFe has been implemented within the ‘front-end’ department and has greatly improved the integration, no measures are taken to improve the integration with these other IT units. The existing dependencies and lack of integration and coordination measures have a negative effect on the flow of the development teams.

Lastly, the interviewed product manager mentions the support and involvement of higher management as troublesome. The product manager explains this by the existing organizational culture of the airline company, in which managers are mainly focussed on

primary business processes and consider IT as less important. Other points of improvement mentioned by the interviewees are an out-dated job structure, inefficiencies in resource management and insufficient documentation, but were perceived to have a negligible effect on the workflow and therefore considered to be less important.

5.8 Case 8: Transport and delivery company- DevOps business and customer applications

Introduction of organization and software delivery

The last case study has been conducted at a large transport and delivery company. The interviews have been conducted within the IT department of the Dutch division, which is part of an international organization. One of the interviewees has worked as head of architecture & innovation and was in that role closely involved with the implementation of agile a few years ago. The other interviewee works as a product owner of one of the Scrum teams.

Figure 23 shows a simplified schematic presentation of the IT organization. The IT organization is divided into several departments. The Run department is responsible for the IT operations, i.e. maintenance and operating of all the deployed IT hardware and software. The department Customer Integration enables the connection of the customers IT systems with the organization's own systems. The interviews were focussed on the Build department, which is tasked with the development of software. This department is divided into three groups; Business-to-business (B2B), Customer Facing (CF) and Business-to-consumer (B2C).

Agile is introduced at B2B and CF approximately 4 years ago. B2B consists of two Scrum teams of in total circa 10 persons and is responsible for developing (Dev) and running (Ops) of software to support the primary business operations. CF consists of circa 4 to 5 Scrum teams of in total 25 people and has a primary focus on the development of new IT technologies that increases the business value. B2B has in that way a stronger connection with the business, whereas CF has fewer dependencies and works more autonomous. The team members are described as multifunctional IT engineers of which circa 2/3 is external. Each team has its own product owner and a dedicated Scrum master.

The interviewees also describe some functions that are in support of the development teams and centrally positioned in the IT organization. This includes security and line management, but also a project management group. Since much of the initiatives are considered as projects, bounded in time and budget, project managers are used for controlling and reporting the progress of IT developments. Other functions that support the development teams are release management and architecture, which are placed at the Run department.

The product owners steer the work of the Scrum teams based on the demands of the business. B2B is in support of the business unit 'Operations', which is the biggest unit within the organization consisting of thousands of people. The product owner has direct contact with several stakeholders that represent Operations and are closely involved in the development process. Some aspects of the scaled agile framework Nexus have been implemented to improve the integration and synchronisation of the software development with the business.

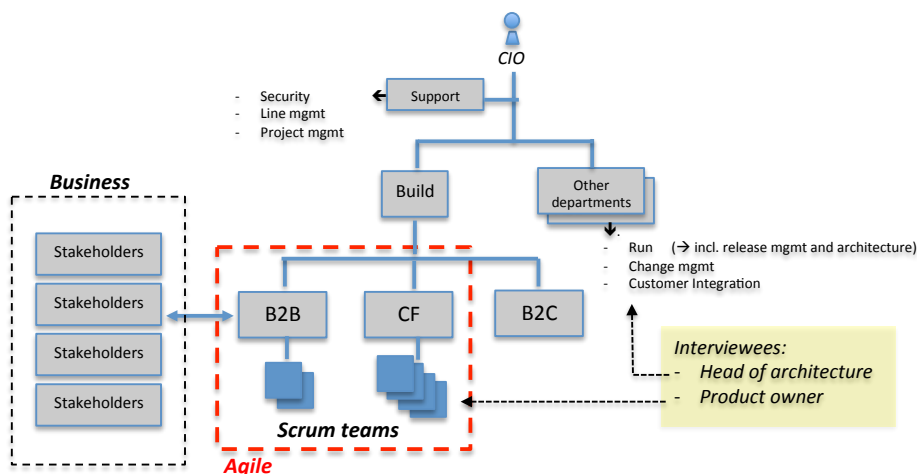


Figure 23- Schematic presentation of case 8 (own illustration)

Interaction software delivery and organization

The interviewed ‘head of architecture & innovation’, who was involved with the implementation of agile, is very positive about the results and points to several successes due to the implementation of agile, such as an increase of; predictability, transparency, flexibility and customer collaboration. On the other hand, he also points out the associated increase in costs, which generates resistance from some managers, especially since the added value of agile development prominently lies in improved adaptability and transparency, rather than in immediate cost savings.

Although both interviewees are quite optimistic about the use of Scrum, they also indicate that some organizational aspects are insufficiently adjusted. According to the interviewees, this can partly be explained by the prevailing management style and organizational culture, that is centred around cost-driven exploitation, while there is a growing need for value-driven business exploration. The support and involvement of senior management is in general perceived as inadequate; agile is accepted, but not actively supported or expanded within the organization. Next to that, not all departments within the IT organization have yet adopted agile, due to certain staff members that are reluctant to change.

As a result, some of the organizational preconditions are less suitable. The current way of budgeting and cost control is described as bureaucratic and slow. The allocation of budget is subjected to an inefficient application procedure and delays the workflow. Likewise, the division of some tasks and responsibilities around the development teams is perceived as confusing, which is explained by the hybrid situation; the use of Scrum in combination with traditional, non-agile processes.

One of the aspects that is emphasized as well arranged in the current situation is planning and progress control. The teams are able to easily deal with changes and the work process is very transparent. Next to that, decision-making about the software development is strongly increased, due to the enhanced involvement of the business, although improvements can be made about the exact division of mandates.

6. CROSS-CASE ANALYSIS

The previous chapter presented the single cases to get familiar with each individual case and gain an impression of some of the findings derived from practice. The cross-case analysis is aimed to create an integral impression of the practical findings by comparing the individual cases. The complete image that results from this cross-case analysis will further be interpreted in the next chapter and discuss some implications that are derived from the cross-case analysis. This chapter starts with Section 6.1 by presenting the approach of the cross-case analysis.

6.1 Approach of cross-case analysis

Figure 24 depicts a schematic presentation of the steps that were undertaken to perform the cross-case analysis. The case data contain the notes and voice records of each interview, plus the documents that were provided by the participating company. For each case a single case report was drafted to gain a general impression of each individual case. The case data and case reports provide information of each single case and give input to the general comparison of multiple cases. Section 6.2 provides an overview of the different cases and compares some general features of the cases, without going into the interview findings. These general features are considered to be relevant to find possible patterns during the cross-case analysis.

The cross-case analysis focuses on a comparison of the different cases in relation to the organizational aspects that are discussed during the interviews. Section 6.3 discusses the interview results for each aspect. Subsequently, Section 6.4 compares the general features within the interview results.

The findings of the cross-case analysis are discussed during experts meetings, which is further elaborated in Section 6.5. The interpretation of the results of the cross-case analysis is presented in the next chapter.

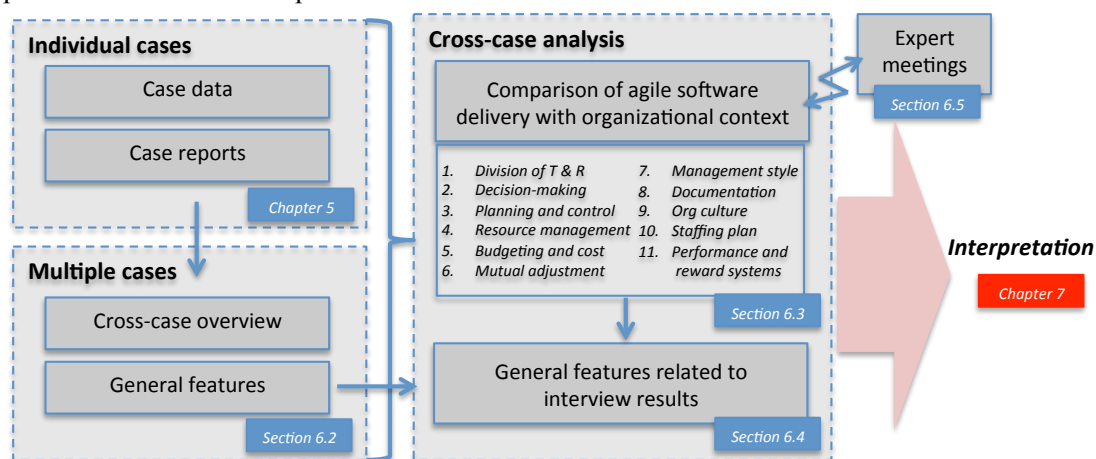


Figure 24- Schematic overview of cross-case analysis (own illustration)

6.2 Cross-case overview

Before getting into the results of the interview findings, this section first presents an overview of the different cases, as shown in Table 7. This table presents the cases on the basis of some general features, which are considered to be relevant for the further cross-case analysis, since these cover certain characteristics that affect the interaction of software delivery and the organization. The interpretation of these aspects and in what way they are relevant to the cross-case analysis is described below the table.

#	ORGANIZATION	PUBLIC VS PRIVATE SECTOR	PROJECT VS BUSINESS AS USUAL (BaU)	STRUCTURE OF CASE	AGILE MATURITY
1	Publisher (private)	Private sector	BaU	<ul style="list-style-type: none"> - 7x Scrum teams - SM and PO per team - IT manager and business analyst 	Working agile for circa 2 years. Agile only adopted by IT department (circa 25-30 people)
2	Public transport company (semi-public)	Semi-public sector	Transition from project to BaU	<ul style="list-style-type: none"> - 5x Scrum teams - SM per team - Delegated PO and subject-PO's centralized in team - PM with support (resource manager + 1x dedicated Scrum team) 	Agile is introduced multiple years ago (>4 years). Scrum is used at multiple projects or departments. First steps made to scale agile within organization
3	Legal assistance authority (public)	Public sector	Project	<ul style="list-style-type: none"> - Amplified Scrum team - SM and PO within team - PM and software delivery manager 	First project in which agile is used (<1 year). Not used at other departments yet.
4	Ministry of Defence (public)	Public sector	Project	<ul style="list-style-type: none"> - 8x Scrum teams - SM and PO within team - PM with staff support 	Agile is introduced multiple years ago (>4 years). Scrum is used for multiple projects within department. First steps made to scale agile within department
5	Law system (public)	Public sector	Project	<ul style="list-style-type: none"> - 5x Scrum teams - SM and PO within team - 1x central PO - PM with support (1x dedicated Scrum team) 	Agile is introduced multiple years ago (>4 years). Scrum is used for multiple projects within program. First steps made to scale agile within program
6	Financial service provider (private)	Private sector	BaU	<ul style="list-style-type: none"> - Ca 10x Scrum teams - SM per team - Overarching/ supporting staff 	Agile is introduced circa 3 to 4 years ago. DevOps is recently introduced. No use of scaled framework
7	Airline company (private)	Private sector	BaU	<ul style="list-style-type: none"> - 5x Scrum teams - SM and PO within team - 1x product manager above teams - Centralized staff capacity to support several delivery groups 	Agile has been widely used throughout the organisation for several years (>4 years). SAFe introduced to department circa 1,5 years ago. First steps made to improve integration between different IT departments.
8	Transport company and delivery service (private)	Private sector	Project and BaU	<ul style="list-style-type: none"> - 2 of 3 IT development groups using Scrum - Agile groups consist of respectively 2 and 5 Scrum teams - Centralized staff capacity to support development groups 	Started working agile circa 3-4 years ago. Implemented within two IT units, but not the whole department yet. Attempt to scale agile by using aspects of Nexus.

Table 7- Overview of the agile software delivery cases with their general features (own table)

Public versus private

A first distinction that can be made is whether the involved organization is operating in the private or the public sector. The public sector is engaged in the activities of providing government goods and services to the general public. These organizations are fully owned, controlled and run by the Government. While the private sector is the segment of national economy that is owned, controlled and managed by private individuals or enterprises (Surbhi, 2015). In this comparison, case 2 is categorized as being semi-public, since this involves a partnership between a private company and a governmental organization.

Significant differences are observed between organizations in the private and public sector regarding for example, their basic objective, earnings (Surbhi, 2015) and its business values (Van der Wal, De Graaf, & Lasthuizen, 2008). Since agile is based on value-driven delivery (Griffiths, 2015), these differences are of importance during the cross-case analysis.

For example, maximizing profit mainly drives decision-making within a private organization. Since this is not the case for public organizations, the interview results might show a clear difference between both types of organizations in relation to decision-making. Another example is the organizational culture, since public organizations are often described as being more bureaucratic (Aucoin, 1997).

Project versus business as usual

As discussed in the introduction of this report, this research focuses on software delivery, which includes projects, which are bounded in time, but also on-going endeavours that don't have a predetermined deadline. This distinction can also be observed in the different cases. Portman (2017) describes these different undertakings as a 'project' and 'business as usual', or *BaU*. BaU refers to the daily operations of the business and differs from a project in a few ways. Relevant differences in the light of this research, are that projects differ from BaU activities since they are temporary and require a predetermined investment (University of Leeds, 2018). This investment is attached to the expected work to be done, which is in accordance with the scope of the project. These differences are relevant to the cross-case analysis since it has a direct relation with the aspects that are discussed during the interviews, such as decision-making and planning.

Another relevant difference between a project and BaU is the way in which the people are involved. This can be described as *project-driven* delivery versus *line-driven* delivery. In case of project-driven delivery, a project organization is established with personnel from different departments within the organization, i.e. '*bringing people to the work*'. Line-driven delivery on the other hand, operates from a permanent work structure that is not dissolved after a product is delivered, i.e. '*bringing work to the people*'. Since agile favours a permanent work structure, i.e. line-driven delivery, this might be noticeable in the interview results. Moreover, the research of Verbruggen (2017, p. 100) shows that agile gives rise to challenges for the definition of projects.

Structure

The structure of each case concisely describes some features of how personnel around the software delivery is positioned. Relevant differences between cases are for example the maximum span of control within a software delivery department and the layout of the governance structure above the delivery teams.

For each case, the software teams work in roughly the same way, namely according to the Scrum Guide. Nevertheless, the layout of the level above the Scrum teams varies per case. For example, the type and amount of staff members differ per case; some cases have appointed a project manager, centralized product ownership or appointed a full staff to support the development teams. How this exactly is structured has a direct relationship with the organizational aspects, such as decision-making and mutual coordination.

An interesting aspect in relation to the structure of the cases is the development of scaled agile frameworks, discussed on page 18. These frameworks prescribe certain structures above team level that are aimed at improving the integration between agile teams. The implementation of scaled agile in the case studies might be noticeable in the interview results.

Agile maturity

The last feature that is taken into account in the general comparison is the agile maturity level of each case. A quick search into agile maturity models shows a wide range of available models, as for example discussed in *Assessment of Agile Maturity Models: A Multiple Case Study* (Ozcan-Top & Demirörs, 2013). However, in the general comparison presented below, a concise interpretation of the agile maturity in terms of time and scaling within the organization is considered to be sufficient, since the agile maturity has not been a focal point during the case studies.

The transition of an organization towards agile might take a while. During this transition, the organization implements changes in a certain order that facilitates agile. Some of these implementation steps might be revealed by linking the interview findings to the time of agile implementation for each case and to what extent they attempted to scale agile within the organization.

6.3 Comparison of organizational aspects

Chapter 3 presented a compilation of aspects that are relevant when investigating the interaction between agile software delivery and its organizational context. These aspects are used as a structure for the interview protocol and formed a common theme during the different interviews. This section discusses the results of the total interview results in relation to the different aspects.

Every interview was concluded with the completion of a questionnaire, in which the interviewee scored each aspect on how much friction the interviewee perceived in relation to this aspect, see Appendix B. The degree of friction should in this case not be interpreted too narrowly. It can also be considered as a measure to indicate how well an organization has embedded this aspect in relation to the software delivery. The results of this questionnaire are presented in Appendix C.

Figure 25 depicts the average friction per aspect, as perceived by the 16 interviewees. The vertical axis presents how much friction is perceived, with 0 being no friction and 5 being a lot of friction. Each aspect is discussed below in order of the average friction, starting with the aspect that scored the highest.

For each aspect a table is presented with some observations that are made during the interviews and for which cases this is applicable. Each observation did recur in different interviews, is described by at least one of the interviewees per case and is considered to be relevant in relation to the interaction of agile with the organizational context. Green boxes

depict observations with a positive connotation and red denote negative ones, see Table 8 to Table 19.

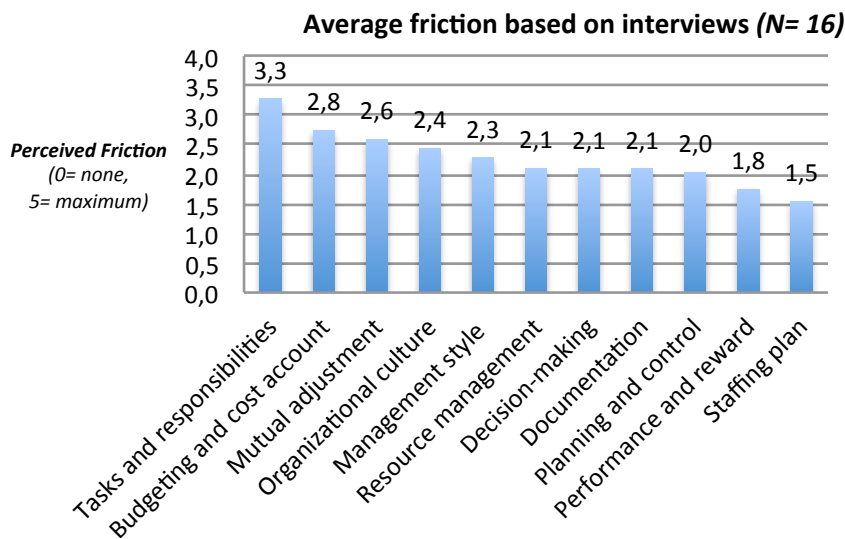


Figure 25- Friction per aspect based on average of interview results, N=16. The vertical axis depicts the perceived friction; 0 being no friction and 5 being a lot of friction (own illustration)



Division of tasks and responsibilities

The division of tasks and responsibilities is on average assessed as the aspect that causes the most friction, as perceived by the interviewees. Positive in relation to this aspect is that all interviewees indicate that the division of tasks and responsibilities is very clear within the development teams, which can be explained by the fact that the Scrum Guideline clearly prescribes the different roles within the Scrum teams. Friction is perceived above team level, since all cases indicate that the division of tasks and responsibilities is in a certain way unclear above team level. Although the own position of the interviewees will influence the perspective on this aspect, it must be noticed that only five of the sixteen interviewees are working within an agile team.

What exactly is unclear differs between the cases. In four cases the participants indicate that the mandates, or authority to make certain decisions, are not clearly assigned within the organization. In two other cases the hierarchical line and thus the authority of the managers is not clearly described. Some interviewees assign the emerged confusion to the misalignment of the delivery teams, working agile, with the upper level, not working agile.

Furthermore, certain tasks or responsibilities are not sufficiently assigned above team level. In three cases the participants are missing staff members that are responsible for the coordination and integration between different teams, i.e. centralized control. In two cases the current tasks related to cost control are not assigned to the suitable persons. Other tasks or responsibilities that are lacking according to the interviewees are for example, an agile coach, test manager, configuration manager and portfolio manager. Lastly, in two cases the interviewees emphasize their impression that particular individuals are unsuited to carry out their job, which can be related to a poor allocation of personnel. Table 8 shows an overview of observations that were derived from the interviews and in which cases this is applicable.

Case #	1	2	3	4	5	6	7	8
'Division of tasks and responsibilities clear on team level'								
'Division of tasks and responsibilities above team level is unclear'								
'Mandates or decision authority unclear'								
'Certain tasks or responsibilities are not sufficiently assigned above team level'								
'Personnel not suited to fulfil their role'								

Table 8- Observations derived from the interviews in relation to the division of tasks and responsibilities (own table)



Budgeting and cost accounting

According to the agile triangle of constraints, as shown in Figure 8, section 3.1.2, budget should be considered as a fixed variable. A recurring observation during the interviews was a description of a budget that was not fixed, but rather coupled to changes in scope and expected work.

Next to that, in approximately half of the cases the participants describe a bureaucratic process of budget approval, or budget allocation. The current processes are perceived as too slow, inefficient or impractical. Another argument is that too much people are involved, or the authorities are not allocated to the appropriate staff members.

In a few cases the interviewees described the current way of budgeting and cost accounting as not always clear. In two cases it is not clear to the interviewees to which budgets certain costs should be accounted, another interview describe their process as being untransparent.

Another observation is that several cases described the use of multiple budgets. Some of the interviewees describe the presence of '*different jars of money*' from where the work is financed. Distinctions are for example made between work caused by new initiatives and work related to maintenance or technical improvements. Difficulties are perceived about this, since this distinction is not always clear and also it causes unnecessary hassles according to the interviewees.

Some of the interviewees did not have sufficient knowledge about this aspect due to their role and were unable to describe how this aspect was implemented in their organization. One case endorses the above-mentioned friction points, but states that '*Although the current process is impractical, it is not perceived as a major bottleneck*'.

Most of the interviewees would like to see some adjustments to the current way of budgeting and cost accounting. Such as the allocation of a fixed budget to each development team in accordance with the required resources and within this budget the team strives to develop as much of the scope as possible. This way of working requires less control of managers and more trust in the teams. Table 9 gives an overview of the observations.

Case #	1	2	3	4	5	6	7	8
'Budget is not fixed; coupled to changes in scope and expected work'								
'Process of budget approval/ allocation is bureaucratic and too slow'								
'Current way of budgeting and cost accounting not always clear'								
Use of different budgets or sources of money.								

Table 9- Observations derived from the interviews in relation to the division of tasks and responsibilities (own table)



Mutual adjustment

In all cases the development teams have certain dependencies with other teams. In almost all cases the participants indicate room for improvement when it comes to adjustment and coordination over the development teams. This might differ per level; a few cases describe that the coordination between teams is well arranged within their own project, but is insufficiently beyond the project.

One of the main causes according to the interviewees is that teams work too isolated and are not sufficiently aware of the interfaces and dependencies with other teams. Many other cases explain this by a lack of central control or overview on higher level. As a result, it is not always clear what the consequences could be of certain choices. This central control can be realised by certain functionalities, such as portfolio or configuration management, but also by setting clear preconditions, i.e. in terms of architecture, technical requirements, etc.

Several adjustments are suggested during the interviews that could improve the mutual adjustment. One of these is already widespread used in the cases, which are meetings between the teams on a regular base. Although this is conducted in almost all cases, it seems to be insufficient. Some of the interviewees suggest that these meetings should be extended to a higher level, but some also argue that the current meetings are unproductive and should be organized differently.

Other measures or suggestions that are discussed during the interviews are the following. Creating shorter lines between teams to stimulate coordination. Use of competence centres to bring 'specialists' of different teams together on regular base and direction teams that are responsible for integration and coordination. Use of specific control measures, for example checking and approve the planning of other teams before starting the work. Extend the demarcation of responsibilities, during one of the interviews describes as '*changing the fence around the team*', i.e. make teams responsible and aware of certain interfaces. And lastly, automated use of version controls and checks during the development process.

Table 10 presents an overview of the different observations.

Case #	1	2	3	4	5	6	7	8
'Adjustment and coordination over development teams is insufficient'	Orange	Orange	Orange	Orange	White	Orange	Orange	Orange
'Teams work too isolated and not aware of interfaces'	Orange	Orange	White	White	White	Orange	White	Orange
'Lack of central control or overview'	White	Orange	Orange	Orange	White	White	Orange	Orange
'Regular meetings between teams'	Green	Green	White	Green	Green	Green	Green	Green

Table 10- Observations derived from the interviews in relation to mutual adjustment (own table)



Organizational culture

In circa half of the cases it is clearly stated that the prevailing organizational culture affects the way of working. Characteristics of the organizational culture that causes friction in relation with the agile way of working are for example bureaucratic, hierarchical or slow and cumbersome decision-making processes. In some cases the employees have little experience to work with deadlines or to work in teams, which is required to adapt to agile.

A few interviewees emphasize a significant difference in the organizational culture between different departments. In general a difference is noticeable between the IT department and the business, but also international differences play a role as is explained in case 8 by the interaction with the international headquarter.

The agile value 'individuals and interaction over processes and tools' is in many cases well implemented on the operational level. On a higher level this value seems to be less applicable. Case 2 even clearly accentuate that more value should be assigned to processes and tools on a higher level.

Furthermore, several interviewees describe a 'change in the right direction' when it comes to understanding and accepting agile within their company. Several interviewees compare the implementation and adaptation to agile with an oil stain, in which the agile mind-set slowly continues to spread further through the organization. The use of an agile coach plays an important role to stimulate this change, according to the interviewees. Table 11 depicts an overview of the different observations.

Case #	1	2	3	4	5	6	7	8
'Prevailing organizational culture affects way of working'	White	White	Orange	White	Orange	White	Orange	Orange
'Significant cultural differences within the organization'	White	White	White	Orange	White	White	Orange	Orange
'Agile value "individuals and interaction over processes and tools" well applied on operational level'	Green	Green	White	White	Green	Green	Green	White
'Change in right direction of understanding and accepting agile within the organization'	Green	White	White	White	Green	Green	White	Green

Table 11- Observations derived from the interviews in relation to organizational culture (own table)



Management style

In several cases the interviewees describe a mismatch between the prevailing management style and agile. Obviously, this strongly differs per person and also per department within the organization. In multiple cases this mismatch is due to managers that demand too much control, are not flexible or too formal. In contrary, in one case both interviewees emphasize the lack of control from higher management; the managers are described as too informal and applying a very loose management style, while the development teams require more guidance. Others emphasize a lack of involvement and support of managers, or blame the managers of holding on to old methods and processes. In general, it can be said that the higher within the organization managers are positioned, the more a mismatch is observed with agile. One of the interviewees mentioned the need for *servant leadership*, but also explained unawareness of higher management about the necessary change in leadership style.

In two cases the influence of the organizational culture on the management style is highlighted, for example being bureaucratic and conservative. In the two cases in which the management style is positively perceived, a fine balance is described between ‘*giving freedom to the development teams*’, while ‘*staying in control*’. Table 12 presents an overview of the observations derived from the interviews.

Case #	1	2	3	4	5	6	7	8
<i>‘Management style of (some of the) managers don’t match agile; hard to find balance between degree of control versus autonomy’</i>								
<i>‘Management style is strongly affected by prevailing organizational culture’</i>								

Table 12- Observations derived from the interviews in relation to management style (own table)



Resource management

One of the main reasons why the interviewees indicated friction in relation to human resource management is the difficulty to hire qualified personnel, due to the tight labour market. Obviously this has no direct relation with agile and cannot be affected by the organization. Nevertheless, it is perceived as friction by the interviewees.

Furthermore, in relation to human resources, in most cases a division is made between line management and functional, or project, management. This creates a situation of supply and demand, in which the development teams specify what human resources are required and line managers try to answer this demand by supplying suitable personnel. Although this situation is observed in most of the cases, the judgements differ; both positive and negative. A few cases describe the situation as being transparent and the presence of clear procedures. In a few other cases, the participants describe the situation of supply and demand as negative, for example due to missing central control or being chaotic.

Next to the management of human resources, multiple interviewees perceive friction regarding the management of other resources, such as the availability of servers, licences, testing environment and infrastructure. The agile development teams need to be facilitated in these kinds of technical resources to perform optimally. Table 13 presents the observations.

Case #	1	2	3	4	5	6	7	8
'Difficulties to hire qualified personnel due to tight labour market'	Orange				Orange	Orange	Orange	Orange
'Division between line management and project/ functional management, causing a situation of supply and demand'		Green	Orange	Green	Green	Orange	Orange	Green
'Other resources (technical) not sufficiently arranged'	Orange		Orange	Orange	Orange			Orange

Table 13- Observations derived from the interviews in relation to resource management (own table)



Decision-making

In most of the cases the interviewees experienced a certain degree of friction regarding decision-making due to an unclear division of mandates. 'Bigger decisions' regarding changes in scope or budget were clearly assigned, namely to the steering group or board, but decisions within the set scope were not clearly assigned. In many cases the development is surrounded by a wide variety of stakeholders that want to have a say regarding functional or technical requirements, causing a negative effect on the speed of work, since many decisions are being pushed back and forth.

Some interviewees described a disruption of the workflow due to top down decision-making. Nevertheless, these kinds of decisions from higher management did not occur very often according to the interviewees and were therefore not perceived as very negative. In one case the interviewees emphasized the affect of the organizational culture on the decision-making; being bureaucratic and too slow.

In some other cases, the interviewees indicate clear procedures and agreements about the decision-making process, which can be explained by a well arranged funnelling and prioritisation of the functional and technical requirements. One of the interviewees described the use of *relative weighting* when choosing between different business initiatives. Table 14 presents an overview of the observations that can be related to decision-making.

Case #	1	2	3	4	5	6	7	8
'Unclear or insufficient division of mandates'		Orange	Orange	Orange	Orange			Orange
'Top down decisions disturb the flow of work'	Orange	Orange					Orange	
'Negative effect of organizational culture on decision-making'					Orange			
'Clear procedures and agreements about the decision-making process'	Green					Green	Green	

Table 14- Observations derived from the interviews in relation to decision-making (own table)



Documentation

The interviewees were asked how the agile value ‘working software over comprehensive documentation’ reflected in practice. In most of the cases the current way of documenting is described as insufficient by at least one of the interviewees. As a result, problems may arise in the transferring of work, products or information, which has a direct relation with the dependencies between teams. Although many interviewees describe documentation as insufficient, it does not directly cause friction, since in most cases it does not directly lead to any problems. One of the interviewees explains this minimalistic documenting by the assumption that ‘*some employees use agile as an excuse to skip the annoying work*’.

Although multiple participants describe documenting as insufficient, this doesn’t apply for all. Moreover, in four cases a significant difference in perspective is observed in relation to the current way of documenting within the same organization. Table 15 presents the observations related to documentation.

Case #	1	2	3	4	5	6	7	8
‘Current way of documenting is insufficient’								
‘Different perspective within case in relation to documentation’								

Table 15- Observations derived from the interviews in relation to documentation style (own table)



Planning and control systems

In general, the interviewees perceived this aspect positively. In all cases it was explained that the organization in a certain way tried to combine some elements of a waterfall planning with agile. Waterfall was used to look further ahead to determine the work on a more abstract level, e.g. roadmap, which was translated into more details on the short run and finally resulting in a sprint planning of 2 to 4 weeks. In this way the participants tried to obtain an optimum in relation to the agile value ‘responding to change over following a plan’.

Although most participants describe this way of working as very practical, a few denote that looking too far ahead also creates inflexibility or as ‘non-agile’. In one case the executive board even imposed a margin on the planning deviations to keep control. Another difficulty that is described is that planning further ahead is challenging in practice, which can partially be explained by the many interdependencies between different development teams.

Also the way in which agile facilitates progress reporting is indicated as very positive. Scrum stimulates transparency due to the use of tools such as Jira, but also due to the execution of demos each sprint, demanding little extra effort to the development teams. Still, in a few cases negative experiences are described due to different expectations of higher management or the business, in relation to progress reporting. One participant explains how the report format of higher management demands fixed requirements and in that way does not match agile, another participant expresses how the progress demanded by the executive board is not aligned with the agile sprint planning. Table 16 presents an overview of the observations.

Case #	1	2	3	4	5	6	7	8
<i>'Combination between waterfall and agile, to find optimum of looking forward and being flexible'</i>								
<i>'Planning too far ahead creates inflexibility'</i>								
<i>'Planning further ahead is difficult in practice'</i>								
<i>'Progress report is efficient and transparent'</i>								
<i>'Different expectations of higher management or business regarding progress reports'</i>								

Table 16- Observations derived from the interviews in relation to planning and control systems (own table)



Performance and reward systems

In comparison with the other aspects, performance and reward systems were considered to cause relatively less friction. Several interviewees assigned little value to this aspect for the reason that a lot of the personnel involved in the software development are external, which makes performance assessments less relevant.

In most cases a so-named line-manager is installed that is responsible for conducting the performance assessments and rewarding of personnel. In addition, in all these cases the interviewees describe a distance between the position of these line-managers and their personnel. This affects the judgement of the line-managers, since they are able to sufficiently observe the performance of their personnel. Some interviewees describe a situation in which the line-managers use the observation of surrounding colleagues as input to a performance assessment. Although this situation is described as undesirable, it is in general not perceived as a big problem by the interviewees.

In all cases the performance and reward systems are focussed on the individual and no specific attention is paid to the team performances. This alternative is discussed in several interviewees, with diverging responses. Some participants were positive about this alternative since it is more in line with agile, others expected some practical challenges such as further isolation between teams. Table 17 shows the two observations related to performance and reward systems.

Case #	1	2	3	4	5	6	7	8
<i>'Predominant use of external personnel makes this aspect less relevant'</i>								
<i>'Line-managers unable to sufficiently observe the performance of their personnel'</i>								

Table 17- Observations derived from the interviews in relation to performance and reward systems (own table)



Staffing plan

Almost all cases show the absence of an up-to-date job structure. In some cases no formal job structure is present and in some other cases a formal job structure is present, but out-dated and not adapted to agile. The absence of a formal, up-to-date job structure can have several consequences, for example regarding the hiring of external personnel, the exact division of roles, but also the career development of employees. However, in all these cases the interviewees describe the absent job structure as non- or little problematic. Although it is formally not arranged, the work floor deals with this in a pragmatic way and acts according to the actual situation, which is in line with the relative little friction assigned to this aspect.

In a few cases the interviewees express a feeling that in some cases personnel is assigned to certain tasks based on their availability, rather than their suitability. This phenomenon is for example clearly described in the first case, in which the organization recently moved to agile and the former project managers were installed as product managers, although this new role requires a different set of competences and qualities. Table 18 shows the observations related to staffing plan.

	Case #	1	2	3	4	5	6	7	8
'No or outdated job structure'		Orange	Orange	Orange	White	Orange	Orange	Orange	White
'Nevertheless not perceived as a big problem on the work floor'		Green	Green	Green	White	Green	Green	Green	White
'Assigning personnel to tasks based on availability, rather than suitability'		Orange	White	Orange	White	White	White	White	Orange

Table 18- Observations derived from the interviews in relation to staffing plan (own table)

Other points

The eleven aspects as described above were used as guidance during the interviews. The open structure of the interview protocol intended to give enough freedom to the interviewees and not to steer too much in a certain direction.

All interviewees were asked to discuss any other topics that were not included in the eleven aspects, but were considered to be relevant in relation to the research question. Most interviewees found the interview protocol complete and had no additions that could not be assigned to one of the aspects, except for two cases.

Case 4 and 7 pointed to the importance of *user involvement* in relation with agile. In both cases the software solution is developed for a large group of people within the organization, i.e. the users, consisting in these cases of over more than 1000 people. Yet, a relatively small group that represent the users is directly involved with the software development from out the business.

Most cases emphasize an improved collaboration between software development and the business due to the implementation of agile, in line with the agile value 'customer collaboration over contract negotiation'. However, input from the business is given by a limited proxy that represents the future users. Case 4 describes an obtained discrepancy

between input regarding the functional requirements from the business representatives and the actual users, who did not always agree with these requirements. One of the interviewees of case 7 indicate that no attention is paid to the user experience, both in advance of the development process, as well as the absence of evaluating of user experiences. Table 19 shows user involvement as additional aspect and to what cases this is related.

Case #	1	2	3	4	5	6	7	8
'User involvement'								

Table 19- Relevant aspects in addition to the interview protocol (own table)

6.4 General features related to interview results

Section 6.2 presented an overview of the cases based on some general features, but did not yet consider any of the interview results in relation to these features. This section discusses some patterns that are obtained when relating these general features to the interview findings, as discussed in the previous section, 6.3.

Private versus public and project versus business as usual

The first distinction that can be made in relation to the general features is the difference between the organizations operating in the public sector and those organizations operating in the private sector. Another generic difference discussed in Section 6.2 is between the software delivery conducted as a project and the business as usual, BaU. Looking closer to these two distinctions it seems that an overlap exists between these distinctions and the selected cases. As Table 20 shows, half of the case studies were conducted at profit organizations, i.e. private sector, but are also characterized as business as usual. The other half involved non-profit organizations, i.e. public sector, in which the work is conducted as a project. This overlap can be considered as a remarkable concurrence without apparent causal connection. A possible explanation for this occurrence could be that public organizations are more bureaucratic and reluctant to change, therefore it takes longer to transform from a project-driven to a line-driven organization.

Case #	Private versus public sector	Project versus business as usual
1	Private	BaU
2	Public *	Project **
3	Public	Project
4	Public	Project
5	Public	Project
6	Private	BaU
7	Private	BaU
8	Private	BaU
* Semi-public		** Transformation from project- to line organization

Table 20- Overlap between general features in relation to cases (own table)

Again the results of the interview questionnaire were used, see Appendix C, which present the friction perceived by the interviewees in relation to the aspects. This time a distinction is made between the cases. One half includes the cases in the public sector and describing a project organization, i.e. case number 2, 3, 4 and 5, named *type I* cases. The other half

includes the cases in the private sector and working with a line organization, i.e. case numbers 1, 6, 7 and 8, named *type II* cases. The results are depicted in Figure 26.

When looking closer to the results per aspects the results show some differences between the two types of cases. Although the results are based on a very small amount of cases and one can *not* speak of *significant* differences, the observed differences are closer interpreted by looking at the interview results and in that way trying to understand the observed differences. Aspects that are further discussed are division of tasks and responsibilities, budgeting and cost accounting, resource management, decision-making, performance and rewards systems and staffing plan.

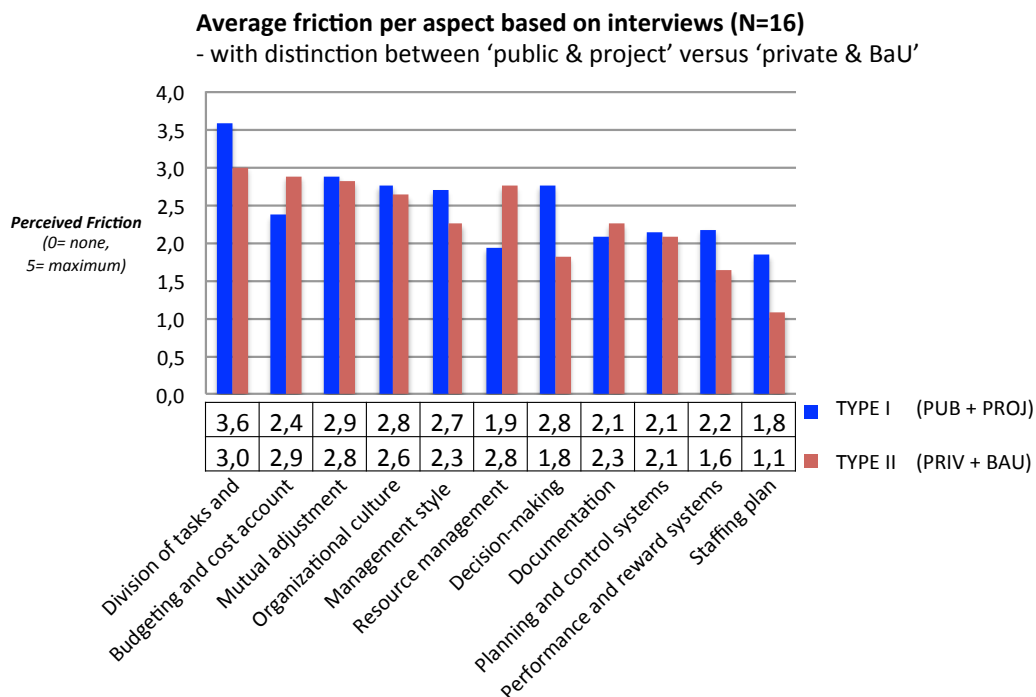


Figure 26- Friction per aspect based on average with distinction between public+ project and private+ BaU. The vertical axis depicts the perceived friction; 0 being no friction and 5 being a lot of friction (own illustration)

Although the scoring of the aspect **division of tasks and responsibilities** shows a difference between the two types of cases, no obvious explanation can be found in the answers of the participants in relation to this aspect. Both types of cases perceive difficulties in the division of tasks and responsibilities above team level. An argument that can be made to explain why the type I cases appoint more friction to this aspect is that projects involve a temporary work structure, while business as usual, or line organizations are more permanent, giving more time and opportunity to establish tasks and responsibilities around the agile teams.

The type II cases assign more friction to **budgeting and cost accounting**. When taking a closer look to the interviews, it seems that almost all type II cases describe a bureaucratic and too slow process of budget allocation, and describe how the current way of budgeting and cost accounting is not always clear. Moreover, all these cases are using different budgets or sources of money, while this is not described for the type I cases. A few of the type I cases also describe the use of a fixed project budget, which gives more clarity to the involved participants and is also more aligned with agile. Another explanation that can be assigned to this observation is that people working within a profit organization might have other expectations regarding budgeting in comparison of people working in a non-profit organization.

Figure 26 also shows a clear difference between the two types of cases in relation to **resource management**. However, no obvious explanation to this can be found in the interview reports. All type II cases emphasize the tight labour market conditions and the difficulties to hire qualified personnel, but this should be the same for all organizations.

The type I cases; public sector and projects, assign more friction to their current way of **decision-making**. Looking at the interview results shows that almost all type II cases describe the presence of clear procedures and agreements about their decision-making processes, while this is not expressed in the type I cases. For example, two of the type II cases describe the use of specific methods, e.g. relative weighting, to compare business initiatives with each other. This comparison gives direct input to the prioritization of work conducted by the software delivery teams and in that way enhances the decision-making. Moreover, all type I cases mention unclear or insufficient division of mandates, while this is not observed in all type II cases. For example, one of the commercial cases describes how one business owner is assigned who has a final say in case of any disputes.

Lastly, an obvious difference is found in relation to the aspects **performance and reward systems** and **staffing plan**. Again, the interview results show no noticeable difference in the interview reports that can be related to explain this difference. In line with the explanation given above in relation to the division between tasks and responsibilities, the higher friction perceived by the type I cases might be related to the temporary work structure of projects.

Maturity of cases

Although the transition towards agile has not been a focal point during the interviews, a distinction can be made between the cases how long agile has been adopted and to what extent it is implemented within the organization. The majority of the cases have implemented agile several years ago and only a few cases recently made the transition. The ‘youngest’ case that can be distinguished is case number 3, since it involves the first project within the organization where agile is used and started less than a year ago. Subsequently, in case number 1 agile is introduced less than two years ago and has so far only been adopted by the IT department. Cases number 6 and 8 have implemented agile circa 3 to 4 years ago, but agile has not been adopted by the whole IT department. In all other cases agile has been introduced more than 4 years ago. Moreover, most of these cases have made some adjustments to scale agile within the organization, e.g. case number 2, 4 and 5. Case number 7 has been working with SAFe for circa 1,5 years at the moment of the interview. Table 21 presents a summary of some case features that are related to the agile maturity.

#	Time agile introduced	Agile expansion within organization	Scaling of agile
1	Circa 2 years	Limited to IT department (circa 25-30 people)	None
2	More than 4 years ago	Agile used at multiple projects or departments (>100 people)	First steps made to scale agile within organization
3	Less than 1 year ago	Limited to one project (<20 people)	None
4	More than 4 years ago	Agile used for multiple projects within department (>100 people)	First steps made to scale agile within organization
5	More than 4 years ago	First steps made to scale agile within program (>100 people)	First steps made to scale agile within organization

6	Circa 3 to 4 years ago	Agile used at multiple projects or departments (>100 people)	None
7	More than 4 years ago	Agile has been widely used throughout the organization (>100 people)	SAFe introduced to department circa 1,5 years ago
8	Circa 3 to 4 years ago	Agile not adopted by whole IT department yet; only two units (<40 people)	Attempt to scale agile by using aspects of Nexus

Table 21- Case features describing the agile maturity (own table)

The transition from an organization towards agile has not been a focal point during the interviews. Therefore, the interview results don't show how the organizations encountered the different aspects during the implementation of agile and how their perception towards each aspect evolved over time. Analysing the perceived friction per aspect, i.e. Appendix C, in relation to the maturity of each case, also doesn't show any substantial patterns. In other words, **no correlation** can be found between the maturity and perceived friction.

A commonality that can be derived from the interviews is that agile is in each case implemented bottom up. In all cases agile is introduced on a small scale within the IT branch and from thereon expanded to other IT units or departments. Some of the interviews compare this expansion with the metaphor of a growing oil stain; starting at one point, get saturated and simultaneously extend in other directions. In this metaphor the saturation can be interpreted as the adaptation of the environment with agile.

Although the interview results don't show that a particular aspect causes more friction in a relatively immature case in comparison with a more mature case, there is a difference in where in the organization the friction is perceived. For example, friction is perceived regarding organizational culture and management style in both a case with a lower agile maturity, e.g. case 1, as is it in a case with a higher agile maturity, e.g. case 7. Nevertheless, in one case the friction is caused within the IT department where agile still needs to be adopted, but in the other case this friction is caused by the interaction between the agile IT department and higher management of the organization.

Structure

The last general feature that is suggested earlier is the structure of the different cases. In this case this entails the structure *above* the Scrum teams. This includes the way that personnel around the software delivery is positioned and making a distinction between, for example, the maximum span of control and the layout of the governance structure above the delivery teams. However, the difficulty with this feature is that it entails too many variables, making it quite impossible to make a founded comparison between the cases.

For example, the structure above the teams is for each case totally different. In a few cases a group of people is positioned which have been assigned an overarching role. Yet, the size and composition of these groups strongly differ. Another variable that can be distinguished is for example the cases with a single line of sight versus cases with multiple people involved. Looking closer into this variable doesn't show any convincing results that can be linked to this variable.

Another relevant aspect is the positioning of specific integrating roles, such as architecture or portfolio management. Again, also for this variable no commonalities are found, since each of the eight studied cases has got their own, unique organizational structure. For that reason it is

not possible to relate any of the interview results directly to the differences in structure of the studied cases.

6.5 Expert meetings

Expert meetings were held after the completion of the cross-case analysis. In total four experts from KWD Resultaatmanagement have been consulted, during one individual meeting and one plenary meeting with the other three experts. Both meetings took about two hours and were initiated by a short presentation, but most of the time was spend on discussion with the experts. All participants are considered as experts in the field of agile software delivery, since they all have several years of relevant experience with different projects and within different organizations working agile.

The goal of these meetings was twofold. The first goal was to reflect on the research findings by discussing the aspects that resulted from the literature study, see Section 3.3, and the observations that were derived from the interviews, see Section 6.3. The second goal was to enrich the interpretation of the findings by having discussions about the results and linking the obtained patterns to the experiences of the experts.

In general the experts assessed the aspects obtained from literature and later used as guideline for the interviews to be quite complete. Also the observations that were derived from the interviews were recognizable to the experts. New insights that were obtained during the discussions in relation to the interpretation of the findings are included in the following chapter. This can for example be related to possible explanations that were suggested by the experts in relation to observations that were made during the interviews, but also suggestions about what measures can be obtained by organizations to cope with flaws that are observed during the case study.

7. INTERPRETATION

The cross-case analysis as discussed in Chapter 6 presents an overview of the interview results after comparing the individual cases with each other. This chapter analyses the findings as a whole and strives to find a generic interpretation of the interview results. In that way this chapter gives an answer to the **fourth sub-question**:

What patterns can be obtained from the interaction of agile software delivery with its organizational context?

Section 7.1 to Section 7.3 elaborate on the patterns that are obtained from the cross-case analysis. The obtained patterns are further explained by linking the findings back to literature. Section 7.4 presents the conclusion of this chapter by giving an answer to the fourth sub-question.

7.1 Neglecting of adjustments around teams

A first pattern that is derived from the cross-case analysis is that multiple observations can be linked to the perception that organizations neglect to make organizational adjustments around the agile teams.

During the interviews all participants stated that agile was well implemented on team level. Friction was primarily perceived in the organization around the teams. In each case the development teams used Scrum as the agile method. Scrum clearly prescribes procedures, roles, responsibilities that should be applied by the team to work agile. Nevertheless, Scrum neglects to prescribe how the organization around the Scrum teams should be established.

Many of the difficulties that are perceived by the interviewees regarding the interaction between agile and the organization can be related to the lack of attention that is paid to how the preconditions should be arranged around the Scrum teams to effectively facilitate agile. As discussed on page 18, multiple frameworks do exist that can be used to scale agile within an organization. Nevertheless, not all frameworks are suitable for each organization, the frameworks are highly prescriptive and some critics argue that the frameworks diminish the level of agility.

Table 22 presents an overview of observations that are derived from the cross-case analysis, which can be related to an absence of adjustments that could have been done above team level to better facilitate agile software delivery. These observations are related to 9 of the 11 interview aspects, depicted at the left side of the table, and are subdivided into tasks and responsibilities on one-hand and governance mechanisms on the other hand. Governance mechanisms might entail several organizational measures, such as procedures, rules, agreements, preconditions, etc.










Aspect:	INTERVIEW OBSERVATIONS	
	TASKS AND RESPONSIBILITIES	GOVERNANCE MECHANISMS
Division of tasks and responsibilities 	<ul style="list-style-type: none"> - Lack of clarity - Missing of several roles, such as: architecture, implementation management, test management, configuration management, coaching, etc. 	
Budgeting and cost accounting 	<ul style="list-style-type: none"> - Budget responsibilities not clearly assigned 	<ul style="list-style-type: none"> - Not aligned with agile (fixed budgets) - Process of budget approval is bureaucratic, unclear or too slow - Different budgets causes ambiguity
Decision-making 	<ul style="list-style-type: none"> - Mandates or decision authority unclear 	<ul style="list-style-type: none"> - Flow of requirements unclear - Decision-making process unclear - User involvement
Mutual adjustment 	<ul style="list-style-type: none"> - Lack of central control or overview 	<ul style="list-style-type: none"> - Adjustment and coordination over teams is insufficiently - Teams work too isolated and not aware of interfaces
Resource management 	<ul style="list-style-type: none"> - Division between line management and functional management 	<ul style="list-style-type: none"> - Technical resources insufficiently arranged (i.e. infrastructure, architecture, test environment, etc.)
Planning and progress control 	<ul style="list-style-type: none"> + Responsibilities clearly assigned 	<ul style="list-style-type: none"> + Combining waterfall and agile + Progress report is efficient and transparent - Expectations regarding progress reports not aligned
Performance and reward systems 	<ul style="list-style-type: none"> - Line managers unable to sufficiently observe the performance of their personnel 	<ul style="list-style-type: none"> - Procedures not clear throughout whole organization
Staffing plan 		<ul style="list-style-type: none"> - No or out-dated job structure
Documentation 		<ul style="list-style-type: none"> - Current way of documenting is insufficient

Table 22- Interview observations that can be explained by the absence of adjustments above team level (own table)

A way to describe the required preconditions around software development is *IT governance*. In their book 'IT Governance' Weill and Ross describe IT governance as 'specifying the decision rights and accountability framework to encourage desirable behaviour in the use of IT' (2004, p. 8). The year of this publication shows that IT governance is nothing new, but as well necessary in case of agile software delivery. Moreover, Qumer and Henderson-Sellers already showed in their research (2008) that 77% of the participants underlined the importance of IT governance in large agile software development.

In this thesis, decision rights and accountability framework is interpreted as tasks and responsibilities, and governance mechanisms above the development teams. In this way the observations presented in Table 22 can be explained by the absence of a clear IT governance

above the teams; i.e. clear division of tasks and responsibilities above the teams and clear governance mechanisms within the organization that are aligned with the agile development teams.

A few of the aspects depicted in Table 22 are discussed below to elaborate on how organizations could improve their governance, based on suggestions made during the interviews and by referring to literature.



Division of tasks and responsibilities

As shown in Figure 25, the interviewees assigned most friction to the division of tasks and responsibilities. The interview results show that the division of tasks and responsibilities is described to be clear on team level, but problems are perceived on the upper level.

PRINCE2 distinguishes three levels within the Project Management Team; the *Direction level* is responsible for directing the project and is accountable for the success of the project, the *Management level* is responsible for the day-to-day management of the projects and the *Delivery level* is responsible for delivering the project’s products. Scrum is primarily focussed on the delivery level, but does not pay much attention to the tasks and responsibilities of the management and direction level.

Figure 27 depicts the three levels of project organization, according to PRINCE2 (Axelos, 2018a). The Scrum Guide describes three roles within Scrum that can be assigned to the delivery level, as shown in the picture. The Scrum Guide does not describe the roles on direction- and management level, represented in the figure by a question mark.

The exact roles on these levels differ per situation and organization, and cannot be standardized. Nevertheless some responsibilities are inherent to software delivery, such as IT infrastructure, IT architecture, implementation or release management, test management and configuration management (Kranenburg, Nelissen, & Brouwer, 2003). The roles belonging to project management are described in multiple different ways in literature. The ten knowledge areas described by PMBOK (PMI, 2013) are considered to be quite comprehensive and therefor shown as an example in the figure below.

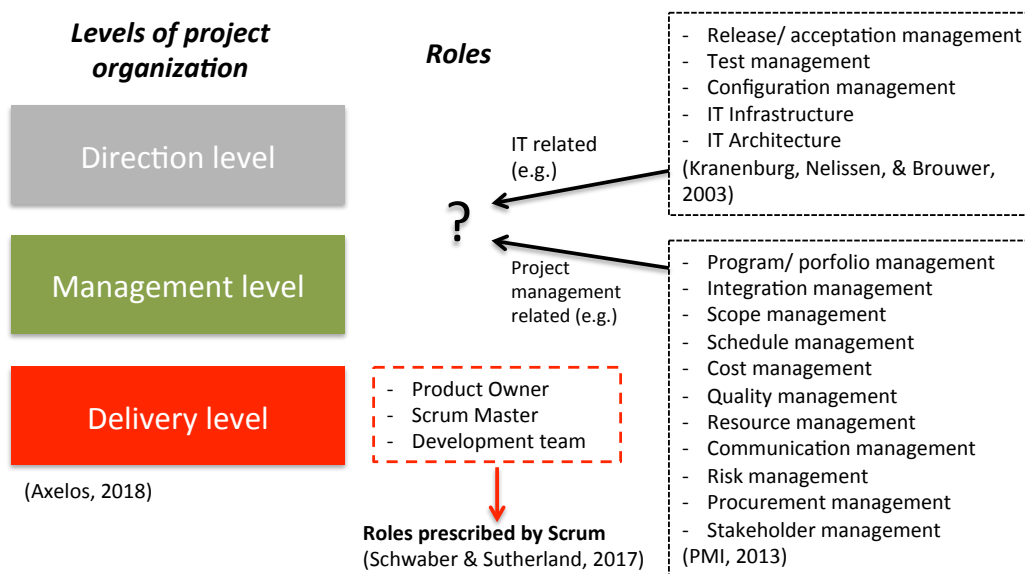


Figure 27- Levels of project organization related to roles (own illustration)

Although no standard division of roles can be applied to all situations, the scaled agile frameworks prescribe several roles that might be applicable for an organization when adapting the organizational context around an agile team. Table 23 gives an overview of some scaled agile frameworks and what roles these frameworks have defined above team level.

Framework:	Roles defined above team level
SAFe (SAFe®, 2017)	<ul style="list-style-type: none"> - Release train engineer (RTE), system train engineer (STE) - Architects, or engineers (system-, solution-, enterprise) - Business owners, epic owners - Portfolio management, product management
Nexus (Schwaber, 2018)	Nexus Integration team, exists to coordinate, coach, and supervise the application of Nexus so the best outcomes are derived. The team consists of the Product Owner, a Scrum Master, and Nexus Integration Team Members.
LeSS (The LeSS Company, 2018)	<ul style="list-style-type: none"> - Head of product group - Scaled product owner - Undone department- This department, ideally, does not exist <p>NB: LeSS assumes that the traditional control tasks are distributed between the feature teams and the product owner, such as project/management office tasks, configuration management, continuous integration support, etc.</p>
S@S (Sutherland & Scrum Inc, 2018)	<ul style="list-style-type: none"> - Executive Action Team - Chief Product Owner and MetaScrum - Knowledge and Infrastructure Teams - Customer Relations, Legal/ Compliance, and People Operations

Table 23- Overview of some scaled agile frameworks and what roles these frameworks have defined above team level (own table)

The findings from the multiple-case study show a shortcoming within organizations in relation to the distribution of tasks and responsibilities. Although no standard instruction does exist for an organization implementing agile in its software delivery teams, it is recommendable that organizations are aware of the necessity of clear roles above team level.



Decision-making

The interview results show that most friction regarding decision-making is perceived due to unclear or insufficient division of mandates, which is directly related with the decision-rights and accountability framework above the agile teams, i.e. IT governance. The work of the development teams is steered by the input from the business in terms of functional requirements, but also by preconditions from the IT organization in terms of technical requirements. Difficulties occur if this flow of requirements is not sufficiently funnelled and prioritized.

A few cases describe clear procedures and agreements about their decision-making process. In all these cases the flow of requirements is clearly covered and decision-rights are clearly assigned to the appropriate persons. The interviewees describe a situation in which the product owners, or managers, have sufficient authority to make the required decisions, and

particular persons within the business have given the authority to make the final decisions on behalf of the business.

Figure 28 depicts a schematic presentation of the decision-making process, in which the stakeholders from both the business and the IT department give input to the flow of requirements towards the agile team(s). The yellow circle with the blue arrow schematically presents the agile teams. Based on the interview results it seems that clear mandates or decision-rights above team level and a clear flow of requirements stimulates effective decision-making.

Literature can be consulted to optimize the decision-making mechanisms within an organization. For example, Weill and Ross use political archetypes (monarchy, feudal, federal, etc.) to describe the combinations of people who have either decision rights or input to IT decisions (2004, pp. 57–83). Next to that, most scaled frameworks pay attention to product management, which is described by SAFe (2017) as the responsibility for identifying customer needs, prioritizing features, guiding the work and developing the program vision and roadmap.

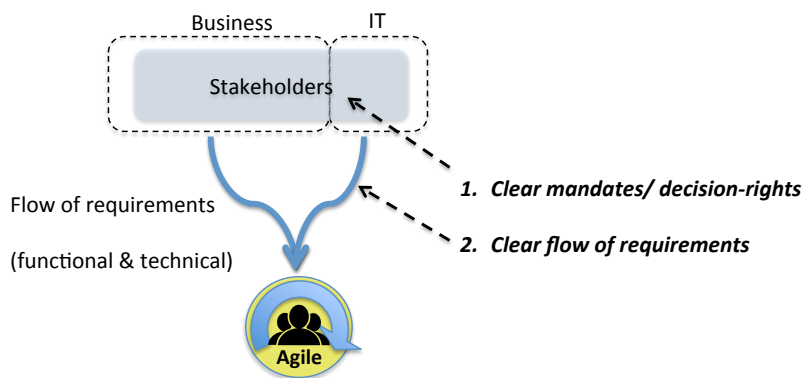


Figure 28- Schematic presentation of flow of requirements from stakeholders to agile team(s) (own illustration)



Mutual adjustment

Almost all interviewees notice that the adjustment and coordination between the development teams is insufficient. This can also be related to the assumption that the organizations primarily focus on team level and neglect the organization around the teams.

The optimization of cooperation and adjustment between different agile teams can be considered as the primary goal of the several scaled frameworks as discussed earlier. A few cases mention the implementation of scaled agile aspects in relation to mutual adjustment, such as for example the adjustment of the development cadence between teams and the business. Only one case mentions the full implementation of SAFe within their department. The interviewees of this case are very optimistic about the results of SAFe in relation to the mutual adjustment and describe how this has strongly improved the mutual adjustment, in contrast to the cooperation with other departments within the organization that did not adapt to SAFe.

Nevertheless, as mentioned earlier organizations seem to be restrained to fully implement a scaled framework. Therefore a few best practices are discussed that are described during the interviews, which aim to improve the mutual adjustment between the teams. Figure 29 depicts a schematic presentation of these mechanisms, in which the light-blue persons depict two agile teams and the dark-blue square depicts the development of a software solution. The red arrow illustrates the technical dependencies as described in the research ‘*Technical*

dependencies in practicing Agile in large-scale Software Development Organizations' (Sekitoleko & Evbota, 2013).

A first mechanism that is described during the interviews is the presence of overhead capacity. The agile team members are primarily focussed on the work of their own team and are unaware of the work of other teams. The appointment of a staff-member that is responsible to improve the integration and coordination between teams. Traditionally, this role used to be fulfilled by the project manager and several practitioners argue for the need of a project manager in an agile environment. The scaled frameworks have adopted these type of roles, as shown in Table 23.

Another suggestion derived from the interviews is the demarcation of responsibilities. A common statement made during the interviews is that teams are unaware of their dependencies with other teams. The case in which mutual adjustment above team level was assessed positively, describe 'moving the fence around a team'. This statement implies that the responsibilities of teams can be extended towards, for example, making a team responsible for the interface with other teams.

Thirdly, multiple cases stimulate the mutual adjustment by bringing people together to discuss the dependencies. This statement is broadly adopted by several scaled frameworks that prescribe daily meetings between different teams. A few interviewees refer to the 'Spotify Model' that is appreciated by the idea to bring people from different development teams together in so-called *Guilds*, or *knowledge centres* and in that way improve the coordination (Portman, 2017, pp. 105–116).

A last mechanism that is derived from the multiple-case study is *continuous integration and delivery*. Continuous integration enables developers to work on the same codebase simultaneously and immediately integrate changes. Continuous delivery is aimed at the automated transition of software to a test environment (Consultancy.nl, 2018). These rather technical measures are aimed at minimizing dependencies by the use of coupled architectures and compatible programming interfaces. This incorporates stage-gates, audits and operational controls into an automated software delivery pipeline (Narayan, 2015, pp. 15–16).

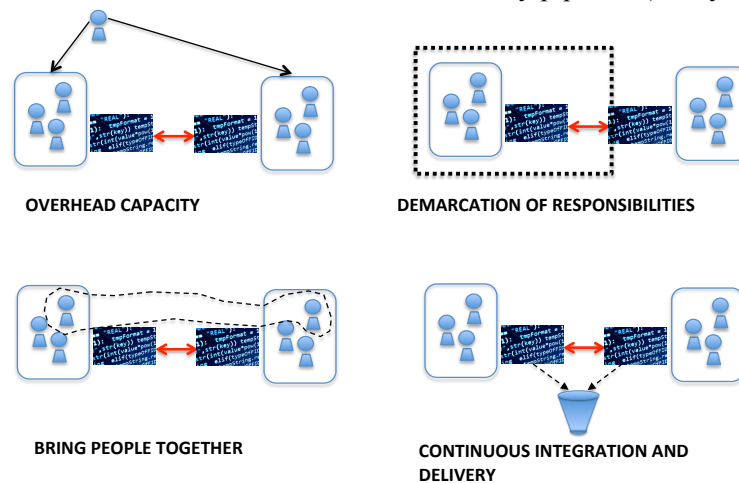


Figure 29- Schematic presentation of best practices derived from interviews regarding mutual adjustment between teams (own illustration)

The above reasoning shows a first pattern; several of the interview observations can be explained by to what extent the governance around the agile teams is adjusted. It seems that organizations should pay attention to how the exact roles above team level are assigned and whether the governance mechanisms within the organizations are aligned with the agile

software teams. Figure 30 is derived from Table 22 and schematically presents how the aspects division of tasks and responsibilities, documentation, budgeting and cost accounting, decision-making, planning and progress control, mutual adjustment, resource management, performance and reward systems and staffing plan are related to the adjustment of the IT governance around the agile teams. In this figure the yellow circle with the blue arrow schematically presents the agile teams.

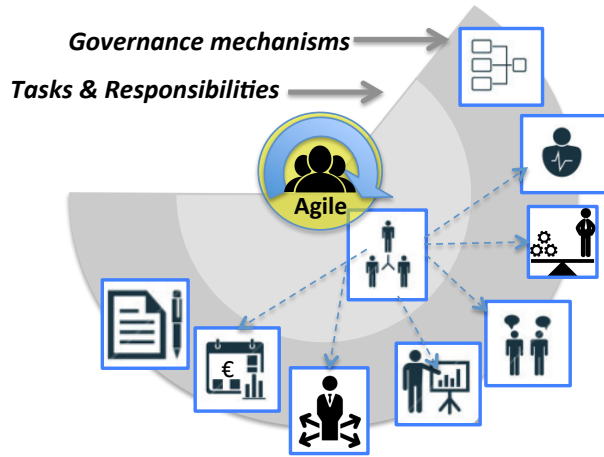

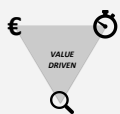



Figure 30- Schematic presentation of how nine interview aspects are directly related to the design of the IT governance above team level (own illustration)

7.2 Understanding agile

The previous section showed how some of the interview observations can be explained by the absence of a suitable IT governance above the agile teams. Based on this pattern, one can argue that organizations could reconsider their IT governance to optimally facilitate agile. When doing so, it is of importance that agile is well understood to make effective organizational adjustments. This section shows another pattern that can be abstracted from the cross-case analysis. Namely, how several interview results can be explained by the way agile is understood and interpreted in practice.

Section 3.1 explained that agile demands another way of thinking and differs from traditional management in multiple ways. If an organization makes adjustments to the IT governance to better facilitate agile, it is of importance that the agile theory is well understood and correctly interpreted. Table 24 shows how some of the interview observations can be explained by how agile is understood and interpreted in practice. These observations can be related to five different aspects, depicted at the left side, and can be linked to particular agile values or principles, depicted at the right side.

Aspect:	INTERVIEW OBSERVATIONS	AGILE VALUES OR PRINCIPLES
Budgeting and cost accounting 	- Budget is not fixed; coupled to changes in scope and expected work.	<i>Reversed triangle of constraints</i> 
Decision-making 	+ Most interviews emphasize an improved collaboration with the business - Nevertheless, mandates are unclear about the decision authority	CUSTOMER COLLABORATION OVER CONTRACT NEGOTIATION






Mutual adjustment 	+ Well established within the teams - Lack of interaction above the teams; working on islands	<i>INDIVIDUALS & INTERACTION OVER PROCESSES AND TOOLS</i>
Planning and progress control 	+ Combination between waterfall and agile, to find optimum of looking forward and being flexible	<i>RESPONDING TO CHANGE OVER FOLLOWING A PLAN</i>
Documentation 	- Current way of documenting is insufficient - Diverging perspectives	<i>WORKING SOFTWARE OVER COMPREHENSIVE DOCUMENTATION</i>
Management style 	- Lacking of general understanding or misinterpretation	All
Organizational culture 	- Lacking of general understanding or misinterpretation	All

Table 24- Overview showing how some of the interview observations can be related to the understanding of agile values or principles (own table)

The aspects depicted in Table 24 are discussed below to elaborate on how a misinterpretation or insufficient understanding of agile affects the interaction of agile with the organization. Firstly, the interview results regarding budgeting and cost accounting will be discussed and related to findings in literature. Thereafter, some of the interview results will be related to the agile values.



Budgeting and cost accounting

The friction regarding budgeting and cost accounting that is described during the interviews, corresponds with the problems that are addressed in the whitepaper of Sirkiä & Laanti (2017), which is later adopted by SAFe 4.0 (Knaster & Leffingwell, 2017). Sirkiä & Laanti argue that traditional budgeting and cost reporting is a system based on rigid-frames, which burdens agile working with unnecessary and counterproductive overhead and friction.

An important different point of view in agile is the reversed triangle of constraints, as discussed in Section 3.1.2. The reversed triangle implies that agile methods allow scope to vary within the fixed parameters of costs and time. Nevertheless, in most of the cases the participants describe a budget that is not fixed, but rather coupled to changes in scope and expected work to be done, causing a situation of unnecessary, bureaucratic budget approval processes.

Another interview observation is the use of different sources of money. Conventional cost accounting describes development as an investment, or capital expenditures, CapEx, and maintenance or operations are cost, i.e. operational expenditures, OpEx (Narayan, 2015, pp. 127–129). However, this distinction cannot always clearly be made as described during the interviews, especially in case of DevOps.

Based on the above reasoning it seems that the financial function could be adapted in response to agile. This requires a different financial governance structure around the teams, as discussed in the previous section. But moreover, adjustments to the financial structure require a good understanding of what agile entails and how to facilitate it.

Alternative approaches of accounting, budgeting and project finance that better suit agile software delivery are found in literature. SAFe recommends a different approach to budgeting

that reduces the overhead and cost accounting, while empowering decentralized decision-making (SAFe®, 2017). Other suggestions are for example made in the whitepaper of Sirkiä & Laanti (2017), or by Narayan in his book *Agile IT Organization Design* (2015, pp. 125–135).

Agile values

The essence of agile originates from the four agile values as discussed in Section 3.1.2. Each agile value will be related to an aspect, based on the interview findings.

Individuals and interaction	over	<i>processes and tools</i>
------------------------------------	------	----------------------------



Several interviewees clarify how this value is well effectuated within the Scrum teams. However, although this value seems to work on team-level, this doesn't automatically work on the parent level when an organization attempts to scale agile. Several interviewees indicate the necessity for more processes and tools above the teams to improve the **mutual adjustment** between teams, as discussed in the previous section. On the other hand, some interviewees clearly protest to this argument and argue that imposing more processes and tools will degrade the agility. Based on these observations it seems that organizations struggle to find an optimum balance between two extremes and moreover this balance differs within the organization.

Customer collaboration	over	<i>contract negotiation</i>
-------------------------------	------	-----------------------------



All interviewees indicate an improved collaboration with the business, i.e. their customers, as a result of the implementation of agile. On the other hand, several interviewees describe a wide diversity of stakeholders that is involved in the software development, with in many cases unclear mandates, which hampers the **decision-making** process. So in relation to the subjected cases it seems that this value indeed is applicable, but also brings along some organizational challenges in relation to the decision-making process, as discussed in the previous section.

Working software	over	<i>comprehensive documentation</i>
-------------------------	------	------------------------------------



The interview results show different perspectives in relation to this value, even within the same case. Some argue that their current way of **documenting** is minimal and lean, while others argue that it is insufficient and will lead to problems on the long run. Although this value seem to be familiar to all interviewees, the exact interpretation and implementation within an organization is unclear. Some interviewees suggest that they would like to have some guidelines from the organization regarding documentation, to stimulate clarity and consistency.

Responding to change	over	<i>following a plan</i>
-----------------------------	------	-------------------------



Most cases describe a combination between these opposites that are mentioned in the value, in relation to **planning and progress control**. Organizations use waterfall-like methods to look further ahead and establish a global planning to describe the contours. In the cases this was realized by the use of a roadmap, functional design, but also the 'traditional' Gantt charts. These global schedules finally result in a short-term planning specifying the details of the work-to-be-done, such as the sprint planning as prescribed by Scrum. Based on the multiple-case study, it seems that this value also require a subtle balance.

As discussed above, several of the interview observations can be explained by to what extent agile is understood and interpreted within an organization. As shown in Table 24, *management style* and *organizational culture* can be related to all agile values and principals, since these aspects are subdued to human behaviour and are affected by to what extent the managers and employees of an organization are aware of what agile exactly entails.

Figure 31 depicts a schematic presentation of the pattern as discussed above. Some of the adjustments within the IT governance are depending on how agile is understood and interpreted. The picture schematically presents five aspects around the agile teams and relates these to the four agile values and the reversed triangle. The right side of the picture shows the aspects *management style* and *organizational culture*, since these are considered to have an overarching role in the understanding of agile. The next section will further discuss these two aspects.

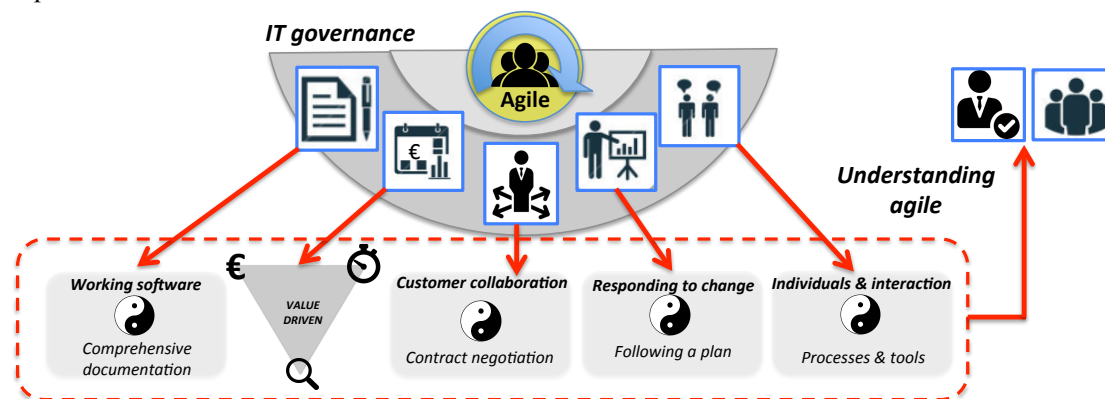


Figure 31- Schematic presentation of the adjustment of how the IT governance is related to the understanding of agile (own illustration)

7.3 Change needs time

Another pattern that can be found by interpreting the interview results is that change needs time. Although this seems to be an obvious statement, several observations in the case study can be related to the transition that an organization needs to undertake when agile is implemented.

As stated earlier, the implementation of agile has not been a focal point during the interviews and therefore the interview results regarding the organizational transformation are limited. Nevertheless, throughout the interviews a general picture did emerge about the transformation of an organization when implementing agile. For example, all cases describe a situation in which agile is implemented bottom up, i.e. initiated within the IT department and from thereon expanded within the organization.

The sequence in which the organization around the agile teams is adapting can be explained by two factors. Firstly, the time that is required to adapt. Secondly, the priority that is perceived to make changes. Table 25, on the next page, presents an overview of the interview aspects in relation to the time required and the priority to change, based on the interview findings and common sense.

All aspects, except for management style and organizational culture, are considered to require a relatively short amount of time to change. These aspects are considered as the *hard*, or more tangible aspects, and are mostly about adjustments in the tasks, responsibilities, rules and procedures within an organization. With good understanding of agile and effective change

management, these aspects can rapidly be adjusted to support the agile development. The *soft* elements, management style and organizational culture, are considered to take relatively more time, since this is about ‘changing people’, including perspectives, attitude and behaviour. A second distinction that is suggested is the priority to change an aspect. This scoring is derived from the interpretation of the interview findings, based on the urgency that the interviewees denoted to each aspect. The tangible aspects documentation, staffing plan, and performance and reward systems, were discussed during the interviews and although the interviewees indicated flaws and room for improvement to these aspects, they also perceived these flaws as non-urgent. Organizational culture is also considered to have a lower priority to change, based on the impression that is derived from the interviews that people accept that it requires more time to change people.












												
Time required to change aspect	Short											
	Long											
Priority to change aspect	High											
	Low											

Table 25- Time required and priority to change in relation to interview aspects (own table)

 **Management style &**  **Organizational culture**

As described above, management style and organizational culture are considered as different aspects compared to the others. This is for one, because these are considered as *soft aspects* since the aspects are about human behaviour, perspectives and habits, whereas the other aspects are more tangible and are more related to organizational mechanisms, rather than human aspects. Moreover, several scholars argue that changing an organizations culture is one of the biggest challenges. As for example, stated by Steve Denning (2011), arguing that an organization’s culture comprises an interlocking set of goals, values, communications practices, attitudes and assumptions.

Secondly, these aspects, especially management style, are considered to fulfil a key role in any organizational transition. Each transition requires active involvement of senior management to establish changes within the organization. How managers exactly should operate to improve organizational change, is discussed in dozens of books and literature, as for example debated by Jon Warner in his blog ‘Top 20 best books on managing change’ (2013). A commonality in this variety of literature is that a proactive participation of higher management is necessary to accomplish the correct organizational adjustments.

Figure 32 shows a schematisation of the above reasoning. As discussed in the previous two sections, many interview results can be related to the adjustment of the IT governance and how agile is understood within an organization. Both these factors describe a certain change within an organization that requires time to effectuate. Based on the interview findings a pattern is obtained in how these changes are made in time, divided in the priority to make

changes and the time required to make changes. In which the *soft* aspects are supposed to take more time than the *hard* aspects.

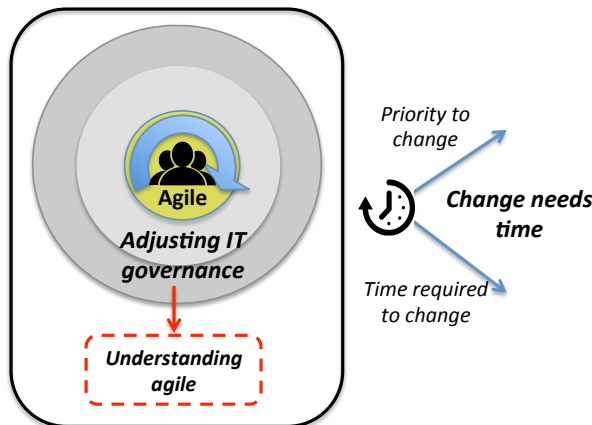


Figure 32- Schematic presentation of how the adjustment of the IT governance and understanding of agile are subjected to time (own illustration)

7.4 Conclusion of interpretation: Three patterns

Chapter 6 presented a wide variety of observations in relation to how agile software delivery interacts with its organizational context, derived from the interviews. This interaction is mainly discussed in terms of friction perceived by the interviewees. Although these results are of different nature and seem to have divergent causes, three patterns are obtained. In this way an answer can be given to the fourth sub-question:

What patterns can be obtained from the interaction of agile software delivery with its organizational context?

Figure 33 depicts a schematic illustration of the obtained patterns as discussed in the previous sections. The illustration includes the eleven aspects and links these aspects to three patterns. Most aspects are directly related to what extent an organization has scaled agile within the organization. Multiple shortcomings indicated by the interviewees can be related to the presumption that the studied cases did not pay enough attention to **adjustments** in the organization around the development teams, i.e. **the IT governance**. The most important aspect related to scaling is the division of tasks and responsibilities. Next to that, organizations need to adjust the governance mechanisms around the agile teams. Adjusting the governance around the teams, i.e. roles and procedures, is assumed to positively affect the aspects documentation, cost account, decision-making, planning, mutual adjustment, resource management, performance systems and staffing plan.

Making these adjustments also demands a good **understanding and interpretation** of what agile entails and requires. As shown, some of the adjustments in the IT governance can directly be related to the agile values and the reversed triangle. The *soft* aspects, management style and organizational culture, are assumed to play an overarching role in understanding.

Lastly, both *adjustments of the IT governance* and *understanding of agile* demand time. Several observations can be related to the transition of an organization when implementing agile; i.e. **change needs time**. In which adjustments within an organization are affected by the priority that is given to change and the time that is required to make a change.

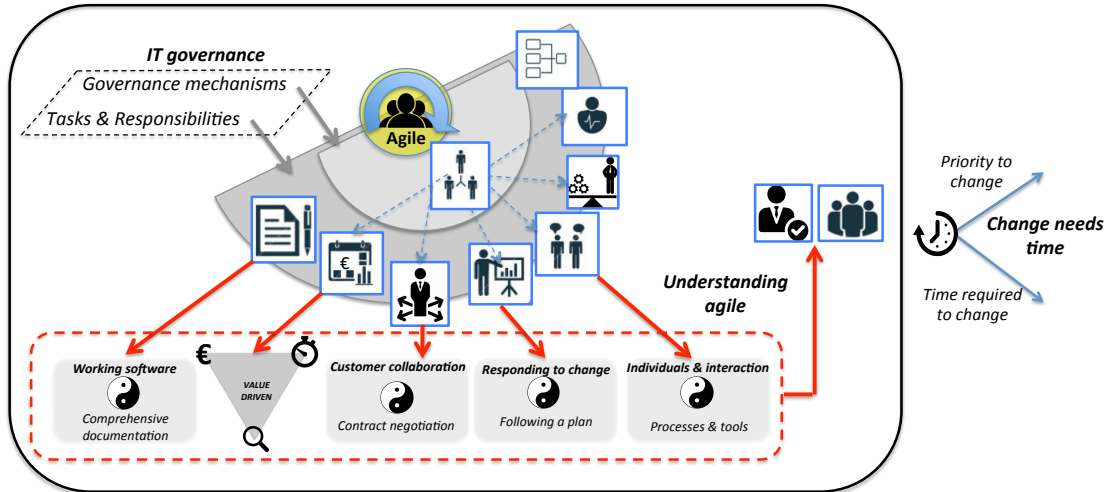


Figure 33- Schematic illustration of the patterns based on the interpretation of the interview findings. Linking the aspects to the IT governance, understanding agile and time (own illustration)

8. DISCUSSION

This research has explored the interaction of agile software delivery with its organizational context. The purpose of this chapter is to interpret the significance of the findings in light of what already is known about the research problem and to explain any new insights (Annesley, 2010).

Section 8.1 starts by bringing to mind the perceived problem and objective of this research, to subsequently discuss how the findings contribute to what is already known. Section 8.2 presents the limitations of the research.

8.1 Contribution of findings

The introduction of this research described a situation of a misalignment of agile management with the surrounding organization. Although several scholars mentioned this misalignment, there was no common, well-defined picture of this misalignment. This deficiency in understanding triggered the incentive for this research to explore the interaction between agile management and its organization. Ideally the findings should improve the knowledge about how organizations align with agile management. This section discusses the significance of the findings and what new insights are obtained.

Existing literature showed a scattered, incoherent image, with different abstraction levels. Some scholars used more indistinctive, abstractive terms (e.g. Nerur et al., 2005; Van Waardenburg & Van Vliet, 2013), where others were able to suggest very tangible, practical issues (e.g. B. Boehm & Turner, 2004; Chow & Cao, 2008). The in-depth investigation into this variety of literature resulted in a compilation of eleven aspects, with more or less the same level of abstraction. These aspects were validated during the interviews and expert meetings. In this way, this research contributes to existing literature by offering a complete list of relevant aspects that can be used to assess how the organizational context interacts around agile software delivery.

Another insight that this research portrays is that it reveals some focal points regarding the alignment of an organization with agile software delivery. The introduction of this research discusses how several researchers and practitioners suspect a misalignment between agile and its organization, but are unable to identify the focal areas of this misalignment. The findings of this research do identify several of these focal areas that are of interest when agile is implemented within a non-agile organization.

The findings of this research also reveal that an optimum needs to be found in how agile is interpreted. Multiple flaws in practice can be related to a one-sided interpretation of what agile entails and how it should be practiced. Agile offers a new way of thinking, but doesn't necessarily imply that former practices should be abandoned. Certain governance mechanisms that are used around project management and software delivery are also applicable in case of an agile method.

Lastly, an insight that is gained throughout this research is that the implementation of agile should not be considered detached from its organizational surrounding. This research indicates that agile is in general well established on team level, but organizations are facing difficulties above team level. Although several agile methods put a focus on team level,

organizations should be aware of adjustments that are required above team level. Changes made on team level should be aligned with its organizational surrounding.

8.2 Discussing the limitations

This section presents some of the limitations that should be taken into account. The limitations of the study are those characteristics of the research that influenced the results and include restraints on generalizability, applications to practice and the chosen research strategy.

Foremost, this research shows a rather one-sided perspective. The sixteen participants that were subjected to the interviews were all directly involved with agile software delivery. No people were involved from the business side of the organization, for example executive managers, although these perspectives are also relevant when investigating the interaction. The findings that are obtained in this research are in that way only based on how the interaction is perceived from one side and should be extended by including the business side of the organization.

It is difficult to pinpoint which of the observations from practice can directly be associated with agile and which observation can be related to other reasons, such as dysfunctional personnel, ineffective management, or for example the transition that an organization undertakes when implementing a new method and the time that is required to establish this. Although the interviewees were aware of the research goal, some of the organizational difficulties or perceived friction described by the interviewees don't naturally have a relation with agile, but can be related to other causes.

External validity deals with the problem of knowing whether the findings of this study are generalizable beyond the immediate study (Yin, 2014, p. 48). The multiple-case study included 8 different organizations and in total 16 interviewees have participated, but still offers a limited representation from practice. Moreover, all cases were conducted in the field of software delivery, which has specific characteristics, which makes the results less generalizable towards other fields. Next to that, although a wide variety of organizations is selected, this proxy does not represent all types of companies. The findings from this research should therefore always be considered in the light of this limited group of representatives. Future research is desired to establish the findings from this research.

Another limitation that should be taken into account is the level of subjectivity. Several elements of this research are affected by the subjectivity of the researcher. For example, the chosen research strategy, the aspects derived from literature and the interpretation of the findings. Also, the use of semi-structured interviews provides a high degree of freedom to the interviewees in answering the questions. The answers given by the interviewees are strongly depending on individual perspectives and interpretation, making the results rather subjective.

Lastly, the introduction of this research describes a suspicion of a misalignment between agile and its organizational context. Although this research suggests the need for improvement of the organization around agile software delivery, this is based on a qualitative analysis. This research doesn't show to what degree the organizational context affects the success of agile software delivery in any way. In this way the results of this research build on the situation that is described in the beginning, without proving the existence of an actual problem.

9. CONCLUSION AND RECOMMENDATIONS

The situation discussed in the introduction of this research described a misalignment of agile management with the surrounding organization. This led to the research objective to explore the organizational context around agile management. The main question to this research was formulated as follows:

“What kind of adjustments can an organization make to better facilitate agile software delivery?”

This chapter starts with presenting an answer to the four sub-questions in Section 9.1. Thereafter, Section 9.2 gives an answer to the research question as shown above. Recommendations for future research and practice are discussed in Section 9.3. Lastly, Section 9.4 presents a personal reflection on the graduation process.

9.1 Answering the sub-questions

The answers to the first two sub-questions are derived from literature. The last two sub-questions are answered by the performance of a multiple-case study.

1. How is agile software delivery described in literature?

This first sub-question was aimed to provide a deeper understanding of agile software delivery and in that way frame the context around the research subject. Traditional management of software delivery is based on a high level of predictability upfront, i.e. plan-driven. This method, many times referred to as *waterfall* method, tends to be well suited for projects with a low complexity, few scope changes and low risk. Agile management has revealed itself as a management approach that copes with an unclear product scope and fast-changing circumstances, making it better suited for software delivery.

Agile is an umbrella term for several iterative development methodologies, that are based on the same vision and core values. Although many different agile methodologies do exist, they can all be related to four values described in the Agile Manifesto. Additionally, a key difference between the agile mind-set and traditional project management is the agile triangle of constraints, which implies that agile methods allow scope to vary within the fixed parameters of cost and time. In other words, agile aims to deliver the most value it can before a set date and within a set budget, i.e. value-driven delivery. The agile values and triangle of constraints are depicted in Figure 34.

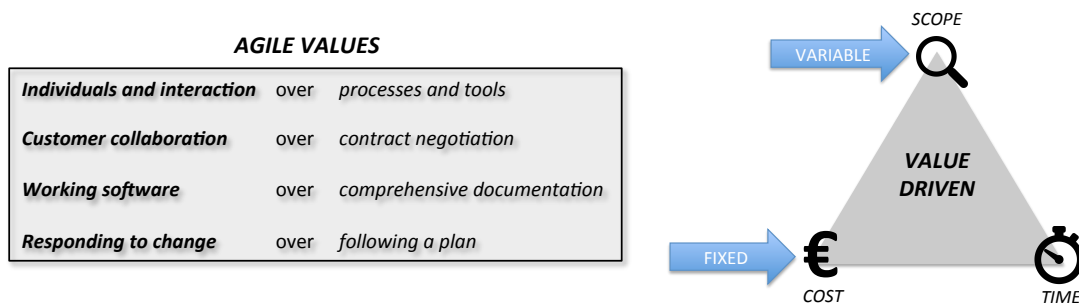


Figure 34- Agile values and agile triangle of constraints (own illustration)

Scrum is by far the most commonly used agile method and precisely describes the roles, tools and processes on team level. In case an agile approach, like Scrum, is extended towards multiple teams or the whole organization, one speaks of scaling agile. Approximately thirty frameworks do exist that can be used to scale agile within an organization, but these frameworks are criticized for being too prescriptive, leaving little room for customization and diminishing the level of agility.

A large variety of different organizations are involved in the development of software products. The commonality in how software delivery is organized in these organizations is by conceptualizing these organizations into two entities; the business and IT department. In this research the IT department is considered as the organizational entity responsible for the software delivery and in that way providing value to the business.

2. *Which aspects are described in literature as being relevant for the interaction between agile software delivery and the organizational context?*

Existing literature shows a wide variety of issues that can be related to the interaction between agile software delivery and the organizational context. These issues are described in different ways, varying from very specific and practical, towards more abstractly described arguments. The findings from literature are collected and where possible merged, resulting in a compilation of 11 aspects as shown in Table 26. These aspects are used to draw up the structure for the interviews.












1	Documentation		5	Budgeting and cost accounting		9	Staffing plan	
2	Decision-making		6	Mutual adjustment		10	Management style	
3	Planning and control systems		7	Division of tasks and responsibilities		11	Organizational culture	
4	Resource management		8	Performance and reward systems				

Table 26- Aspects obtained from literature that are relevant for the interaction between agile software delivery and the organizational context (own table)

3. *In what way does agile software delivery interact with its organizational context in practice?*

An answer to this question is found by conducting a multiple-case study at eight different organizations practicing agile software delivery. For each case, two participants were subjected to a semi-structured interview, which were each directly involved with the software delivery.

Although each case is unique in its own way, multiple commonalities were observed over the different cases in relation to the interaction between agile software delivery and its organizational context. For each aspect an impression was obtained about how it was related to embedding agile within the organization. In this way an overview was created of the friction that was perceived by the interviewees in relation to the different aspects, in that way creating an image of the misalignment of agile within the organization.

An aspect that was in general described negatively was the division of tasks and responsibilities, which was especially caused by an insufficient or unclear division of tasks and responsibilities above team level. Other aspects mentioned by the interviewees as being

insufficiently arranged around the agile teams were for example budgeting and mutual adjustment. Furthermore, the interviewees indicated room for improvement regarding organizational culture and management style, i.e. the *softer* aspects.

On the other hand, the interviewees also described much room for improvement in relation to documentation, staffing plan and performance- and reward systems, but were in general considered to be less urgent or relevant, and therefore were given a lower friction score. Lastly, some aspects, such as planning and control, were in general well adjusted within an organization practicing agile software delivery.

4. *What patterns can be obtained from the interaction of agile software delivery with its organizational context?*

Figure 35 depicts a schematic presentation of the most important patterns that are derived from the multiple-case study. The illustration includes the eleven aspects and relates these aspects to three patterns; the adjustment of the IT governance, understanding of agile and change needs time.

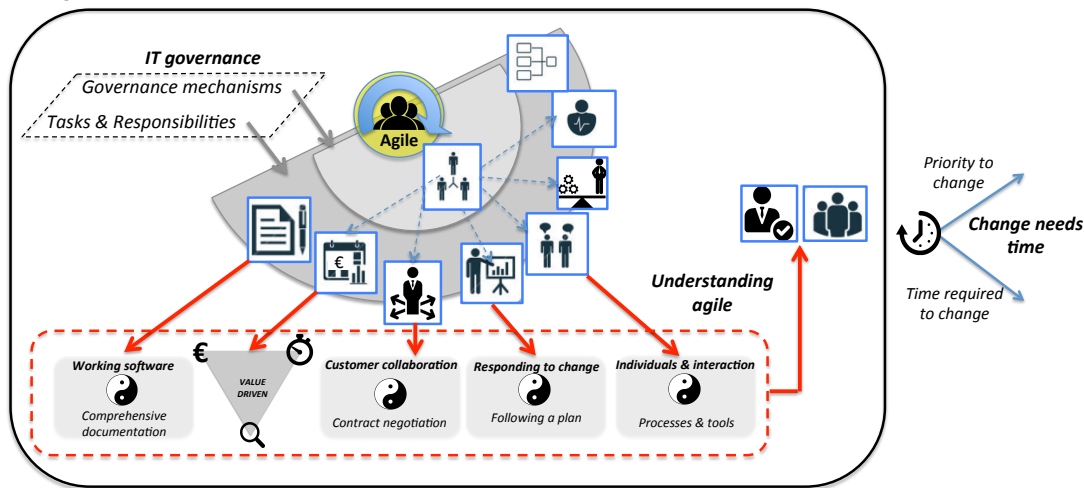


Figure 35- Schematic illustration of the patterns based on the interpretation of the interview findings. Linking the aspects to IT governance, understanding and time (own illustration)

Most of the observations that are made during the interviews can be explained by neglecting of organizational adjustments around the teams. In general, agile is well implemented within the delivery teams, but the organizational surrounding is insufficiently adjusted. This surrounding can be described as the *IT governance*, being the set of tasks, responsibilities and mechanisms around the teams. In addition, the division of tasks and responsibilities takes a central position, since it can be related to several aspects.

Another pattern that explains the interaction of agile with its organization is whether agile is *well understood and interpreted*. Some of the interview observations can be explained by an insufficient understanding of agile. Adjusting the governance structure around the agile teams, requires making adjustments that are aligned with the agile concepts, such as the four values and the reversed triangle.

The last pattern that is suggested to explain the observations of the multiple-case study is that *change needs time*. Every organization undertakes a transition when implementing agile software delivery. Alignment of agile with the organization depends on in which stage an organization is in this transition. Especially the softer aspects, organizational culture and management style, need more time to change. On the other hand, the management style also affects several other aspects, and the adjustment of these aspects.

9.2 Answering the research question

This research aimed to explore the interaction of agile software delivery with the organizational context. Based on this exploration, an impression is obtained about the alignment or misalignment of agile software delivery with its surrounding organization. In this way an answer can be given to the research question:

“What kind of adjustments can an organization make to better facilitate agile software delivery?”

This research brings the obtained patterns back to three factors that explain the interaction of agile and the organization. Relating this to the research question suggests that an organization could consider these factors to better facilitate agile software delivery.

As argued earlier, organizations tend to focus on team level when implementing agile and have the tendency to neglect the organizational surrounding, i.e. scaling of agile. This organizational surrounding is in this research described as the governance structure around the teams. This includes first of all a clear division of tasks and responsibilities around the team and next to that, other governance mechanisms, such as processes, tools, rules and techniques.

Therefore this research concludes that organizations could consider **adjusting** their **governance structure** to better facilitate agile software delivery. A clear division of tasks and responsibilities above team level is assumed to strongly improve the success of agile software delivery and is expected to decrease possible friction as described in the multiple-case study. Next to a clear division of tasks and responsibilities, other governance structures could be revised to assess whether these are optimally aligned. The findings of the cross-case analysis suggest a certain priority in aspects that require attention to be adjusted.

In addition, adjusting the governance structure around the teams also requires a good **understanding of agile**. Aspects as budgeting, decision-making and mutual adjustment demand a different interpretation of the governance structure in comparison with traditional software delivery.

Lastly, implementation of agile can be considered as a **transition** that needs to be managed over time. The required changes to the governance structure won't occur from one day to the next, but demand time. Pro-active management of this change could improve the success of agile software delivery within the organization.

Figure 36 depicts a schematic presentation of the conclusion; adjusting the governance structure around agile teams with a good understanding what agile is and requires, and a pro-active management of these adjustments.

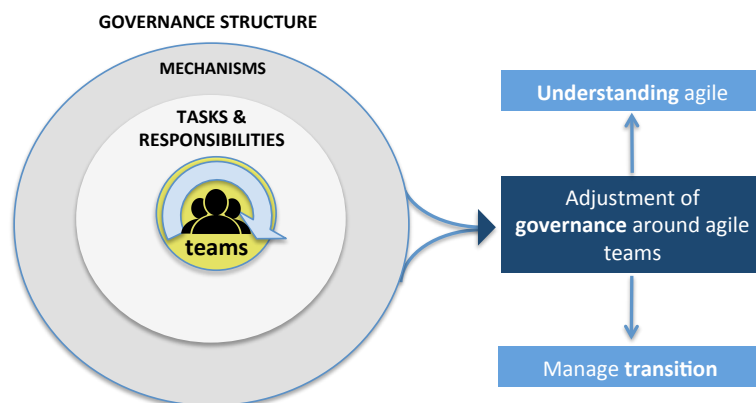


Figure 36- Schematic presentation of adjustment of governance structure around agile teams (own illustration)

9.3 Recommendations

This explorative research gained several new insights about the interaction of agile management and its organizations. Yet, many questions remain unanswered and therefore some suggestions can be made for future research. Next to that, some practical recommendations are obtained for organizations that are faced with challenges regarding fitting agile within the organization.

Recommendations for future research:

The first recommendation is to *extend the research field*. This research was focused on software delivery, but the same explorative research can be conducted within other branches where agile is practiced. Next to that, quantitative study can be conducted to confirm or reject the implied findings of this research.

Secondly, the interviewees that participated in the multiple-case study were all directly involved in the software delivery. *Including of other perspectives* might enrich the findings. For example, people from the business or senior management might offer different insights about agile software delivery.

A third recommendation is to conduct a *comparative research into the governance structure* around software delivery. Comparing cases of agile delivery with traditional, or non-agile, delivery, might identify success factors and tangible advice towards organizations about how to make adjustments to their governance structure.

Fourthly, a recommendation can be made to further investigate the *implementation of agile* within an organization. This includes the identification of steps that need to be taken to make adjustments in the organization to facilitate agile.

Fifthly, the answer to the research question suggests making adjustments to the governance structure around the agile teams, but this should be measured in practice to *investigate* how particular *adjustments affect the success* of agile software delivery.

The last recommendation for future research is to further investigate certain *elements of the governance structure in relation to agile*. Some aspects mentioned in this research demand further investigation about how this is best actualized around agile software delivery. The variety of scaled agile frameworks might offer suggestions that can be translated to general conclusions. The aspects of interest include:

- Division of tasks and responsibilities above team level;
- Coordination and integration of multiple agile teams;
- Agile budgeting and cost accounting;
- Decision-making, including collaboration with the business and user involvement.

Recommendations for practice:

The recommendations to organizations that practice agile software delivery or want to implement agile within their organizations can roughly be divided in two.

Firstly, organizations should consider to *adjust the governance structure* around the agile teams. Working with multiple agile teams within an organization requires more than is described in the Scrum Guide. On the other hand, implementing a full-blown scaled agile framework is not always immediately required, as long as the organization makes well

thought adjustments to the governance around the agile teams. Adjustments to the governance structure can be done in several ways, a few points of interest are mentioned below.

- Organizations should put effort in a *clear and complete division of tasks and responsibilities above team level*. Organisations should consider which responsibilities are applicable and whether these are covered. Roles described in scaled agile frameworks can come to help.
- Working with multiple agile teams requires active *coordination* over the teams and *integration* of work and products. This should be done with an agile mind-set, so maintaining sufficient autonomy on team level. Measures to improve coordination and integration are, amongst others; to make people responsible above the teams, bringing teams together, delegating interfaces and automation. Again, several scaled frameworks suggest measures that could come to help.
- Organizations should reconsider their *financial procedures* when working agile. Procedures that are already in place might hamper agile delivery, due to bureaucratic or slow budget allocation. Also cost accounting should be lean and simple.
- This research shows an improved collaboration between software delivery and the business. However, the *decision-making* that takes place and steers the work of the delivery teams, should be *clear and efficient*. This includes a clear division of mandates and decision-authority.
- Organizations should be aware of other procedures and rules that might need adjustment due to the implementation of agile. This might include rules regarding documentation, an updated staffing plan and agreements about how to report.

The second main recommendation towards organizations is to actively *manage the implementation of agile*. As mentioned above, agile software delivery requires several adjustments throughout the organization that take time.

The multiple-case study showed that agile is in many cases implemented bottom up. This bottom up implementation conceals a risk, since adjustments within an organization are limited to the lower, or operational level. By neglecting the adjustments that are required on a higher level throughout the organization, agile might be hampered by existing organizational mechanisms that are misaligned with agile.

Therefore it is of importance that implementation of agile requires active involvement of higher management and a full analysis is made of the adjustments that need to be done throughout the organization. The full, or holistic image that arises of the adjustments that need to be done must be translated to an implementation plan, which describes the specific steps to be taken in time.

9.4 Personal reflection

Doing this research concludes my MSc study, but has also been an enlightening journey in itself. Compared to my 7 years working as an officer in the army, doing this research was something totally new to me and offered me much food for thought about myself. This section describes a number of challenges and learning moments that I encountered during my research.

Foremost, the explorative character of the research brought a lot of challenges along the way. The exact goal and intended outcome of the research was roughly described when I started

the research and was taking more shape during the research process. The feeling that I was heading somewhere while not exactly knowing where it would end, was something new to me. For a person like myself, who favours to plan as much as possible upfront, this has been a challenging experience and has put my flexibility to the test.

This research took place in the field of software delivery and, more specific, was aimed at agile software delivery. The unfamiliarity with both the area of practice and the research topic challenged me to first obtain a basic understanding before proceeding. Especially the execution of the interviews with having limited knowledge caused a somewhat uncomfortable feeling at first. This situation forced me to quickly filter the relevant information during the conversations and think about follow-up questions at the same time. In hindsight this experience showed me how to rely on my common sense and personnel competences, without having all information available.

The multiple-case study gave me a unique opportunity to get an impression of eight totally different organizations. Since I only have worked within one organization myself, my frame of reference was rather limited. All interviews were very extensive and covered a wide range of different aspects within each organization, which enabled me to gain a quite complete impression of each organization. Many of the things I've witnessed in practice have extended my frame of reference and might come to use when I return working at the Ministry of Defence.

One of the most challenging factors during the graduation process was to handle the different expectations and opinions surrounding the research. This was partially caused by the different types of people within the graduating committee, whom had sometimes slightly diverging perspectives on my work. Next to that, many people from the graduation company showed interest in the research and were happy to share their expectations with me. The intensive involvement from different people, interesting discussions and diverging opinions was highly appreciated since it gave me new insights and enriched my analysis in many ways, but also challenged me to filter the feedback that was given to prevent wandering back and forth, and thus to stick to my own way of doing this research.

REFERENCES

- Abend, G. (2008). The Meaning of 'Theory.' *Sociological Theory*, 26(2), 173–199.
- Allisy-Roberts, P., Ambrosi, P., Bartlett, D. T., Coursey, B. M., DeWerd, L. A., Fantuzzi, E., & McDonald, J. C. (2016). 11th annual state of agile report. *Journal of the ICRU*, 6(2), 7–8.
- Annesley, T. M. (2010). The Discussion Section: Your Closing Argument. *Clinical Chemistry*, 56(November), 1671–1674. <https://doi.org/10.1373/clinchem.2010.155358>
- Aucoin, P. (1997). The design of public organizations for the 21st century: why bureaucracy will survive in public management. *Canadian Public Administration/Administration Publique Du Canada*, 40(2), 290–306. <https://doi.org/10.1111/j.1754-7121.1997.tb01511.x>
- Axelos. (2018a). PRINCE2 | Project Management | AXELOS. Retrieved February 12, 2018, from <https://www.axelos.com/best-practice-solutions/prince2>
- Axelos. (2018b). What is ITIL. Retrieved from <https://www.axelos.com/best-practice-solutions/itil/what-is-itil>
- Barton, B., Schwaber, K., Rawsthorne, D., Beauregard, F., McMichael, B., McAuliffe, J., & Szalay, V. (2005). Reporting Scrum Project Progress to Executive Management through Metrics.
- Bassellier, G., & Benbasat, I. (2004). Technology Professionals: Conceptual Development and Influence on it-business partnerships, 28(4), 673–694.
- Bodych, M. A. (2012). Integrated project management in the organization. *PMI Global Congress-EMEA*.
- Boehm, B., & Turner, R. (2004). Balancing agility and discipline: evaluating and integrating agile and plan-driven methods. *Proceedings. 26th International Conference on Software Engineering*, (July 2004), 718–719. <https://doi.org/10.1109/ICSE.2004.1317503>
- Boehm, B., & Turner, R. (2005). Management Challenges to Implementing Agile Processes in Traditional Development Organizations. *IEEE Software*, 22(5), 30–39.
- Boeije, H. (2014). *Analyseren in kwalitatief onderzoek- denken en doen* (2e druk). Boom uitgevers.
- Bozzuto, B. (2011). When Agile “Doesn’t Apply&” - SolutionsIQ. Retrieved February 12, 2018, from <https://www.solutionsiq.com/resource/blog-post/when-agile-doesnt-apply/>
- Braude, E. J., & Bernstein, M. E. (2016). *Software Engineering- Modern Approaches* (Second). Waveland Press, Inc.
- Cagara, B. (2017). Business Process Vs Project Management Process. Retrieved February 6, 2018, from <https://project-management.com/business-process-vs-project-management-process/>
- Chandler, A. D. (1962). *Strategy and Structure*. Cambridge: MIT Press.
- Charette, R. N. (2005). Why Software Fails. *IEEE Spectrum*, 42(9), 42–49.
- Chow, T., & Cao, D. B. (2008). A survey study of critical success factors in agile software projects. *Journal of Systems and Software*, 81(6), 961–971. <https://doi.org/10.1016/j.jss.2007.08.020>
- Cockburn, A., & Highsmith, J. (2001). Agile software development: The people factor. *Computer*, 34(11), 131–133.
- Cohn, M., & Ford, D. (2003). Introducing an agile process to an organization. *Computer*, 36(6), 74–78.
- Consultancy.nl. (2018). Continuous Integration, Continuous Delivery: dé stap na agile. Retrieved June 20, 2018, from <https://www.consultancy.nl/nieuws/16755/continuous-integration-continuous-delivery-de-stap-na-agile>
- Daft, R. L. (2004). *Organization Theory and Design* (8th ed.). Thomson South-Western.
- De Haes, S., & Van Grembergen, W. (2009). An exploratory study into IT governance implementations and its impact on business/ IT alignment. *Information Systems Management*, 26(2), 123–137.

- Denning, S. (2011). How Do You Change An Organizational Culture? Retrieved June 21, 2018, from <https://www.forbes.com/sites/stevedenning/2011/07/23/how-do-you-change-an-organizational-culture/#44db3ce339dc>
- Denning, S. (2015). How to make the whole organization agile.
- Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software*, 85(6), 1213–1221.
- Eisenhardt, K. M. (1989). The Academy of Management Review Building Theories from Case Study Research. *C Academy of Management Review*, 14(4), 532–550. Retrieved from <http://www.jstor.org/stable/258557>
- Engwall, M. (2003). No project is an island: Linking projects to history and context. *Research Policy*, 32(5), 789–808. [https://doi.org/10.1016/S0048-7333\(02\)00088-4](https://doi.org/10.1016/S0048-7333(02)00088-4)
- Fowler, M., & Highsmith, J. (2001). The Agile Manifesto. *Software Development*, August.
- Graziano, A. M., & Raulin, M. L. (2007). *Research Methods* (6th ed.). Pearson International Edition.
- Griffiths, M. (2015). *PMI-ACP exam prep : rapid learning to pass the PMI Agile Certified Practitioner (PMI-ACP) exam* (Second). RMC Publications.
- Hall, D. J., & Saias, M. A. (1980). Strategy Follows Structure! *Source: Strategic Management Journal*, 1(2), 149–163.
- Hastle, S., & Wojewoda, S. (2015). Standish Group 2015 Chaos Report - Q&A. Retrieved February 6, 2018, from <https://www.infoq.com/articles/standish-chaos-2015>
- Heidenberg, J., Matinlassi, M., Pikkarainen, M., Hirkman, P., & Partanen, J. (2010). Systematic Piloting of Agile Methods in the Large: Two Cases in Embedded Systems Development. *LNCS*, 6156, 47–61.
- Highsmith, J., & Boston Consulting Group. (2012). Adaptive Leadership, 1–10.
- Hobday, M. (2000). The project-based organisation: an ideal form for managing complex products and systems? *Research Policy*, 29(7–8), 871–893.
- IEEE Standard. (1990). IEEE Standard Glossary of Software Engineering Terminology, 610(12).
- Karlström, D., & Runeson, P. (2006). Integrating agile software development into stage-gate managed product development. *Empirical Software Engineering*, 11, 203–225.
- Knaster, R., & Leffingwell, D. (2017). *SAFe 4.0 Distilled*. Addison-Wesley Professional.
- Kranenburg, K., Nelissen, F., & Brouwer, J. (2003). *De moderne softwarefabriek* (Tweede druk). Den Haag: Ten Hagen en Stam Uitgevers.
- Leffingwell, D. (2011). *Agile Software Requirements*. Pearson Education (US).
- Luftman, J. Y. (2003). Assessing IT/Business alignment. *Information Systems Management*, 20(4), 9–15.
- Maylor, H., Brady, T., Cooke-Davies, T., & Hodgson, D. (2006). From projectification to programmification. *International Journal of Project Management*, 24(8), 663–674.
- Mintzberg, H. (1983). *Structure in Fives: Designing effective organizations*. Prentice-Hall, Inc.
- Narayan, S. (2015). *Agile IT Organization Design*. Pearson Education (US).
- Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of Migrating to Agile Methodologies. *Communications of the ACM*, 48(5).
- Nicholas, J. M., & Steyn, H. (2017). *Project Management for Engineering, Business and Technology* (5th ed.). Routledge.
- O’Grady, S. (2014). What is a Software Company Today? *Tecosystems*. Retrieved from <http://redmonk.com/sogrady/2014/03/19/software-company/>
- Owen, R., Koskela, L., Henrich, G., & Codinhoto, R. (2006). Is agile project management applicable to construction? *Salford Centre for Research and Innovation*, 51–66. <https://doi.org/10.1111/j.1467->

9302.2008.00617.x

- Ozcan-Top, O., & Demirörs, O. (2013). Assessment of Agile Maturity Models: A Multiple Case Study. *CCIS*, 349, 130–141. Retrieved from https://link.springer.com/content/pdf/10.1007/978-3-642-38833-0_12.pdf
- PMI. (2013). *A guide to the project management body of knowledge (PMBOK guide)* (5th ed.). Project Management Institute, Inc (PMI).
- Portman, H. (2017). *Scaling Agile in Organizations*. Van Haren Publishing.
- Powell-Morse, A. (2017). Scaled Agile Framework: What is it and how do you use it?
- Qumer, A., & Henderson-Sellers, B. (2008). A framework to support the evaluation, adoption and improvement of agile methods in practice. *Journal of Systems and Software*, 81(11), 1899–1919. <https://doi.org/10.1016/j.jss.2007.12.806>
- Ravanfar, M. M. (2015). Analyzing Organizational Structure based on 7s model of McKinsey. *International Journal of Academic Research in Business and Social Sciences*, 5(5).
- Robbins, S. P., & Barnwell, N. (2006). *Organisation Theory, Concepts and Cases* (5th ed.). Pearson Prentice Hall.
- Royce, W. W. (1970). Managing the Development of Large Software Systems: Concepts and Techniques. *IEEE WESCON, August 197*, 1–9.
- Rubin, H. J., & Rubin, I. S. (2005). *Qualitative Interviewing- The art of hearing data*. (2nd, Ed.). Sage Publications, Inc.
- SAFe ®. (2017). Scaled Agile Framework. Retrieved February 21, 2018, from <http://www.scaledagileframework.com/>
- Salem, K. A. (2009). *Software Engineering - Kassem A. Saleh - Google Boeken*. Ft. Lauderdale, FL: J. Ross Pub.
- Schmidt, R. F. (2013). *Software Engineering- Architecture-driven Software Development*. *Software Engineering*. Elsevier Inc. Retrieved from <http://linkinghub.elsevier.com/retrieve/pii/B978012407768300001X>
- Schwaber, K. (2018). Online Nexus Guide. Retrieved July 18, 2018, from <https://www.scrum.org/resources/online-nexus-guide>
- Schwaber, K., & Sutherland, J. (2017). The Scrum Guide. *Scrum.Org and ScrumInc*, (July), 17. Retrieved from <http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf>
- Sekitoleko, N., & Evbota, F. (2013). Technical dependencies in practicing Agile in large-scale Software Development Organizations. Retrieved from https://gupea.ub.gu.se/bitstream/2077/38524/1/gupea_2077_38524_1.pdf
- Shehzad, B., Awan, K. M., Lali, M. I.-U., & Aslam, W. (2017). Identification of Patterns in Failure of Software Projects. *Journal of Information Science and Engineering*, 33, 1465–1479.
- Sirkiä, R., & Laanti, M. (2017). Lean and agile financial planning. Retrieved from <https://www.nitor.com/application/files/8314/6737/4669/Whitepaper-24-Dec-2013-Lean-and-Agile-Financial-Planning.pdf>
- Stacey, R. D. (1993). *Strategic Management and Organisational Dynamics*. London: Pitman.
- Surbhi, S. (2015). Difference between public sector and private sector.
- Sutherland, J. (2001). Agile Can Scale: Inventing and Reinventing SCRUM in Five Companies. *Cutter IT Journal*, 14(12), 5–11.
- Sutherland, J., & Scrum Inc. (2018). *The Scrum@Scale Guide*. Retrieved from <https://www.scrumatscale.com/wp-content/uploads/Scrum@Scale-Guide.pdf>
- The LeSS Company. (2018). LeSS Works. Retrieved June 18, 2018, from <https://less.works/>
- Tolfo, C., & Wazlawick, R. S. (2008). The influence of organizational culture on the adoption of extreme programming. *The Journal of Systems and Software*, 81, 1955–1967.
- University of Leeds. (2018). project versusbusiness-as-usual work. Retrieved May 16, 2018, from

https://it.leeds.ac.uk/info/205/project_management/968/project_vs_business-as-usual_work

- Van der Wal, Z., De Graaf, G., & Lasthuizen, K. (2008). What's valued most? Similarities and differences between the organizational values of the public and private sector. *The Authors. Journal Compilation*, 86(2), 465–482. <https://doi.org/10.1111/j.1467-9299.2008.00719.x>
- Van Vliet, H. (2008). *Software engineering : principles and practice* (Third). John Wiley & Sons.
- Van Waardenburg, G., & Van Vliet, H. (2013). When agile meets the enterprise. *Information and Software Technology*, 55(12), 2154–2171.
- Verbruggen, A. (2017). *A project manager's journey towards agile project management*. TU Delft.
- Verschuren, P., & Doorewaard, H. (2010). *Designing a Research Project* (2nd ed.). The Hague, Netherlands: Eleven International Publishing.
- Vinekar, V., Slinkman, C. W., & Nerur, S. (2006). Can Agile and Traditional Systems Development Approaches Coexist? An Ambidextrous View. *Information Systems Management*, 23(3), 31–42.
- Warner, J. (2013). Top 20 Best Books on Managing Change. Retrieved June 21, 2018, from <http://blog.readytomanage.com/top-20-best-books-on-managing-change/>
- Weill, P., & Ross, J. W. (2004). *IT Governance*. Harvard Business School Press.
- Wijnen, G., Weggeman, M., & Kor, R. (1989). *Verbeteren en vernieuwen van organisaties*. Kluwer.
- Wysocki, R. K. (2014). *Effective project management : traditional, agile, extreme*. Wiley.
- Xia, W., & Lee, G. (2004). Grasping the complexity. *Communicatin of the ACM*, 47(5).
- Yilmaz, M., O 'connor, R. V., & Clarke, P. (2015). Software Development Roles: A Multi-Project Empirical Investigation. *ACM SIGSOFT*, 40(1), 1–5.
- Yin, R. K. (2014). *Case Study Research: Design and Methods* (5th ed.). Sage Publications, Inc.

APPENDICES

A. Interview structure

Inleiding

Deze bijlage beschrijft het gedachte verloop van het interview. Figure 37 geeft een schematische weergave van de structuur van het interview. Gezien de interviews gekwalificeerd zijn als semigestructureerd, zal deze structuur als leidraad dienen in de uitvoering, maar ook ruimte bieden om er vanaf te wijken. De inhoud van het interview is grotendeels afgeleid van de aspecten gedefinieerd in paragraaf 3.3.

De interviewstructuur is grofweg in vijf delen opgesplitst. De vet gedrukte tekst in de schematische weergave heeft een directe relatie met de hoofdvragen. Deel 1 betreft de introductie en heeft als doel een beeld te krijgen van de geïnterviewde, de organisatie en de software delivery. Dit moet onder andere resulteren in een visualisatie van de organisatie rondom de software delivery.

Deel 2 sluit aan op de visualisatie van de organisatie en verdieping zoeken omtrent zes aspecten die een directe relatie hebben met de organisatiestructuur. Deel 3 en 4, respectievelijk uitvoering en personeel, hebben geen directe relatie met de organisatiestructuur en worden daarom als aparte delen behandeld. In deze delen komen de overige aspecten aan bod. Het interview wordt afgesloten door deel 5, waarin een compilatie van de aspecten aan de geïnterviewde wordt voorgelegd om deze te vergelijken en eventueel aan te vullen.

In de onderstaande tekst wordt de opzet van het interview chronologisch beschreven, inclusief een tijdindicatie, en zal tijdens de uitvoering van het interview als leidraad dienen.

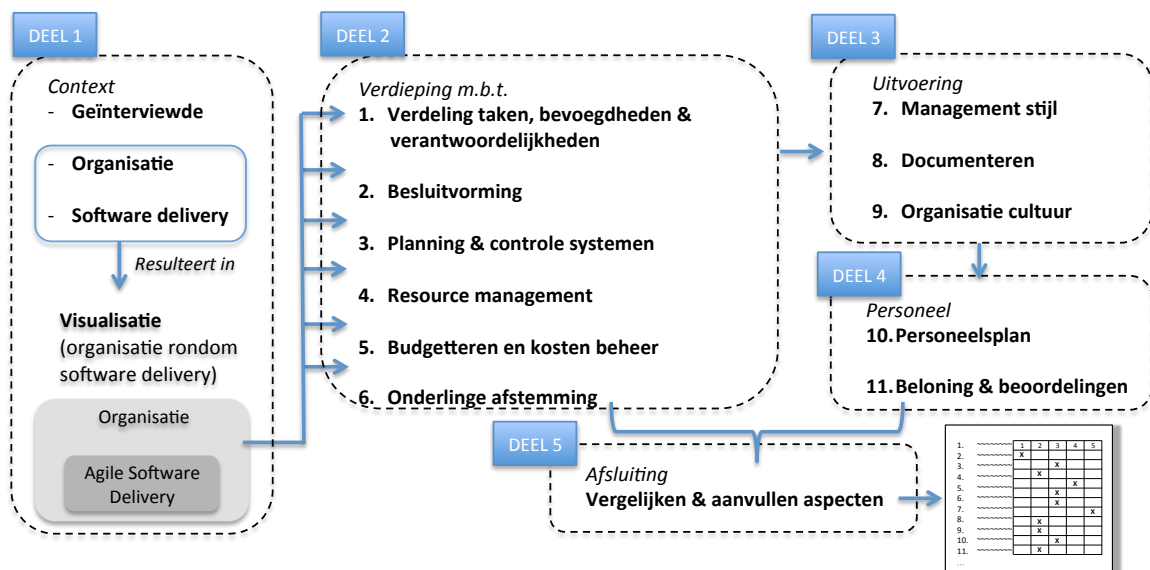


Figure 37- Schematische weergave opzet interview (eigen afbeelding)

Vorbereidingen op interview:

Email gestuurd met daarin:

- Bevestiging tijd en locatie interview;
- Aanleveren algemene informatie.

Introductie (5 min)

Onderzoek:

- Onderzoek vindt plaats i.h.k.v. afstudeeronderzoek aan TU Delft i.s.m. KWD;
- Richt zich op software ontwikkelingsprojecten die agile worden uitgevoerd, maar verschillende soorten organisaties;
- O.b.v. literatuuronderzoek een aantal aspecten onderkent die van invloed zijn. Deze zullen als leidraad dienen tijdens interview.

Praktische zaken:

- De uitkomst van het interview zal vertrouwelijk worden behandeld en anoniem in het onderzoeksrapport worden opgenomen;
- Tijdens onderzoek worden geluidsopname gemaakt, deze zal na het onderzoek worden verwijderd;
- Na het interview wordt een samenvatting opgestuurd, met het verzoek om deze te valideren;
- Het interview zal circa 1,5 tot 2 uur in beslag nemen.

Verloop interview:

- Informatie aangeleverd voorafgaand aan het interview, kan deels terugkeren tijdens het interview ter verduidelijking of verdieping.
- Interview zal globaal volgende opzet hebben:
 - Algemene informatie over de geïnterviewde, de organisatie en de software delivery;
 - Visualisatie organisatie rondom software delivery;
 - Interactie tussen organisatie en software delivery o.b.v. een elftal aspecten.

DEEL 1- Context (20 min)

GEINTERVIEWDE	
Vragen	Doel
<p>Huidige functie?</p> <ul style="list-style-type: none"> - <i>Wat is de formele titel?</i> - <i>Toelichting van functie(inhoud)?</i> - <i>Taken, bevoegdheden en verantwoordelijkheden?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - <i>Indruk krijgen over ervaring en achtergrond van geïnterviewde</i>
<p>Hoeveel ervaring?</p> <ul style="list-style-type: none"> - <i>Binnen organisatie?</i> - <i>Binnen functie?</i> 	
<p>Rol m.b.t. software delivery?</p> <ul style="list-style-type: none"> - <i>Omschrijving rol?</i> - <i>Nog met andere projecten/ delivery's te maken?</i> - <i>Tijdsverdeling?</i> 	
<p>Wat is je ervaring met agile? (bijv. Scum, XP, FDD, etc)</p> <ul style="list-style-type: none"> - <i>Hoeveel jaar ervaring?</i> - <i>Met welke methodes ervaring?</i> - <i>Certificaten gehaald/ trainingen gedaan?</i> 	
	<div style="display: flex; justify-content: space-around;"> <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> <p><u>Agile methods</u></p> <ul style="list-style-type: none"> ○ Scrum/ Scrumban ○ XP ○ Kanban ○ FDD ○ DSDM ○ Lean dev ○ DevOps ○ Hybrid ... ○ Other ... </div> <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> <p><u>Scaled agile methods</u></p> <ul style="list-style-type: none"> ○ SAFe ○ LeSS ○ Nexus ○ S@S ○ Spotify ○ PRINCE2 Agile ○ Other... </div> </div>

ORGANISATIE	
<i>Vragen</i>	<i>Doel</i>
<p>Waar zit ik hier (bijv. afdeling) binnen de organisatie?</p> <ul style="list-style-type: none"> - <i>Visualiseren?</i> - <i>Wat doet dit deel van de organisatie?</i> 	<p>Uitkomst informatie:</p> <p>Beeld krijgen van organisatie waar geïnterviewde zich begeeft (visueel)</p>
<p>Omschrijving van organisatie(onderdeel/ entiteit)?</p> <ul style="list-style-type: none"> - <i>Missie/ doel/ visie</i> - <i>Grootte</i> - <i>Aantal werknemers</i> - <i>Type/ aantal klanten</i> <p><i>Etc</i></p>	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Verzamelen relevante informatie m.b.t. organisatie in het algemeen
<p>Ervaring organisatie(onderdeel/ entiteit) met agile?</p> <ul style="list-style-type: none"> - <i>Wat voor methodieken worden gebruikt?</i> - <i>Waar in organisatie? (evt. scaled agile?)</i> - <i>Hoelang in gebruik? Hoe organisatie binnen gekomen?</i> - <i>Ervaring?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in doorvoering agile binnen organisatie - Inzicht in volwassenheidsniveau organisatie m.b.t. agile <div style="display: flex; justify-content: space-around; margin-top: 10px;"> <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> <p><u>Agile methods</u></p> <ul style="list-style-type: none"> ○ Scrum/ Scrumban ○ XP ○ Kanban ○ FDD ○ DSDM ○ Lean dev ○ DevOps ○ Hybrid ... ○ Other ... </div> <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> <p><u>Scaled agile methods</u></p> <ul style="list-style-type: none"> ○ SAFe ○ LeSS ○ Nexus ○ S@S ○ Spotify ○ PRINCE2 Agile ○ Other... </div> </div>

SOFTWARE DELIVERY	
<i>Vragen</i>	<i>Doel</i>
<p>Wat voor soort software product wordt er ontwikkeld? (in dit deel v.d. organisatie)</p> <ul style="list-style-type: none"> - <i>Scope</i> - <i>Deliverables</i> - <i>Budget</i> - <i>Sprake van project? (start/ geplande einddatum)</i> - <i>Andere producten waar dit deel v.d. organisatie aan werkt?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht krijgen in software product - Inzicht krijgen in toegevoegde waarde voor organisatie
<p>Hoe is de software delivery gepositioneerd en georganiseerd binnen de organisatie?</p> <ul style="list-style-type: none"> - <i>Welke afdelingen betrokken?</i> - <i>Aantal + grootte teams?</i> - <i>Co-locatie teams?</i> - <i>Fasering/ verloop in tijd?</i> - <i>Wordt er werk uitbesteed? (resultaat-/ inspanningsverplichting)</i> <p><i>Hoe loopt de scope?</i> <i>(pad van 'idee', of 'business case' tot 'realisatie')</i></p>	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Compleet beeld (visueel) krijgen van hoe software delivery is gepositioneerd en georganiseerd/
<p>Wat voor soort management-/ ontwikkel methode</p>	<p>Uitkomst informatie:</p>

<p>wordt er gebruikt?</p> <ul style="list-style-type: none"> - <i>Agile? (Scrum, XP, FDD, etc)</i> - <i>Waarom is er voor deze methode gekozen?</i> - <i>Welke aspecten werken wel/ niet binnen deze methode?</i> 	<p>- Inzicht krijgen in (project) management methode</p> <div style="display: flex; justify-content: space-around;"> <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> <p><u>Agile methods</u></p> <ul style="list-style-type: none"> ○ Scrum/ Scrumban ○ XP ○ Kanban ○ FDD ○ DSDM ○ Lean dev ○ DevOps ○ Hybrid ... ○ Other ... </div> <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> <p><u>Scaled agile methods</u></p> <ul style="list-style-type: none"> ○ SAFe ○ LeSS ○ Nexus ○ S@S ○ Spotify ○ PRINCE2 Agile ○ Other... </div> </div>
<p>Hoe verloopt de software ontwikkeling?</p> <ul style="list-style-type: none"> - <i>Worden resultaten behaald binnen randvoorwaarden (tijd/ geld)?</i> - <i>Is opdrachtgever tevreden?</i> - <i>Hoe is sfeer/ succesbeleving binnen de betrokken teams?</i> <p><i>Hoe kan dat?</i></p>	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Indruk krijgen van succes van software delivery - Eerste indruk verkrijgen van frictie punten binnen software delivery

DEEL 2- Verdieping organisatiestructuur o.b.v. zes aspecten (25 min)

1. Hoe zijn de taken, bevoegdheden en verantwoordelijkheden verdeeld rondom de software delivery?	
<i>Vragen</i>	<i>Doel</i>
<p>Hoe zijn de management taken verdeeld?</p> <p><i>Zie schema volgende blz.</i></p>	
<p>Hoe worden de werkzaamheden verdeeld binnen de geschetste software delivery? (open vraag → doorvragen)</p> <ul style="list-style-type: none"> - <i>Welke functionarissen te onderscheiden?</i> - <i>Beschrijving per rol</i> - <i>Mis je ergens bepaalde taken, verantwoordelijkheden of bevoegdheden?</i> 	<p>Uitkomst informatie:</p> <p>Compleet beeld krijgen van de betrokken functionarissen en begrip van desbetreffende taken, bevoegdheden en</p>
<p>Hoe verloopt deze verdeling van TBV in praktijk?</p> <ul style="list-style-type: none"> - <i>Is het voor iedereen duidelijk?</i> - <i>Zijn alle TBV goed ondergebracht, of missen er nog TBV?</i> - - <i>Veranderen rollen gedurende de tijd?</i> - <i>Hoe kan dat?</i> 	<p>Uitkomst informatie:</p> <p>Indruk krijgen van de mate waarin er met taken, bevoegdheden en verantwoordelijkheden wordt omgegaan</p>
<p>Is er een project manager benodigd voor deze software delivery?</p>	<p>Project manager wel/ niet benodigd, omdat...</p>
<p>M.b.t. huidige inrichting van taken en verantwoordelijkheden:</p> <ul style="list-style-type: none"> - <i>Is dit goed ingericht?</i> - <i>Doorvragen → hoe kan dat? Evt. voorbeelden?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - <i>Mate bepalen waarin aspect van invloed is op het verloop van de software delivery.</i>

	TEAM LEVEL		UPPER LEVEL		
	Scrummaster	Product Owner	Proj Manager
Scope mgmt*					
Cost mgmt*					
Schedule mgmt*					
Quality mgmt*					
Risk mgmt*					
Stakeholder mgmt*					
(Human) resource mgmt*					
Process mgmt*					
Portfolio mgmt*					
Architecture**					
Release/ implementation mgmt**					
<i>Additions...</i>					

* General project management task, based on the ‘*knowledge areas of project management*’ according to PMBOK (Nicholas & Steyn, 2017, pp. 12–13)

** Additional tasks for IT projects, based on interpretation of processes and tasks described by ITIL (Axelos, 2018b), ‘*De modern softwarefabriek*’ (Kranenburg et al., 2003) and SAFe (SAFe®, 2017)

2. Hoe vindt de besluitvorming plaats binnen de software delivery?			
Vragen	Doel		
<p>Welke formele beslissingsbevoegdheden zijn er te onderscheiden?</p> <p>Waar in de organisatie (zie schema) vinden de besluiten plaats m.b.t. -----></p> <ul style="list-style-type: none"> - <i>Welke functionarissen?</i> - <i>Welke bevoegdheden?</i> - <i>Wat is de lijn van besluitvorming?</i> - <i>Is het duidelijk wie welke besluiten maakt?</i> <ul style="list-style-type: none"> - <i>Hoe verloopt de escalatie van een issue impediment?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Compleet beeld m.b.t. bevoegdheden van functionarissen; formeel en informeel <div style="background-color: #e0e0e0; padding: 5px; margin-top: 10px;"> <p><u>Voorbeelden</u> (wijzigingen m.b.t.):</p> <table style="width: 100%; border: none;"> <tr> <td style="width: 50%; vertical-align: top;"> <ul style="list-style-type: none"> ○ Scope ○ Budget ○ Tijd/ planning ○ Kwaliteit ○ Risico's </td> <td style="width: 50%; vertical-align: top;"> <ul style="list-style-type: none"> ○ Stakeholders ○ Informatie ○ HR/ personeel ○ Process ○ Etc... </td> </tr> </table> </div>	<ul style="list-style-type: none"> ○ Scope ○ Budget ○ Tijd/ planning ○ Kwaliteit ○ Risico's 	<ul style="list-style-type: none"> ○ Stakeholders ○ Informatie ○ HR/ personeel ○ Process ○ Etc...
<ul style="list-style-type: none"> ○ Scope ○ Budget ○ Tijd/ planning ○ Kwaliteit ○ Risico's 	<ul style="list-style-type: none"> ○ Stakeholders ○ Informatie ○ HR/ personeel ○ Process ○ Etc... 		
<p>Hoe vinden besluiten plaats?</p> <ul style="list-style-type: none"> - <i>Top down/ bottum up?</i> - <i>Formeel/ informeel?</i> - <i>Wel/ niet in overleg?</i> - <i>Is er ruimte voor inbreng/ input van derden?</i> <ul style="list-style-type: none"> - <i>Worden besluiten toegelicht?</i> - <i>Hoe gecommuniceerd?</i> <ul style="list-style-type: none"> - <i>Snelheid van besluitvorming?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Indruk krijgen van hoe besluiten tot stand komen en hoe deze worden doorgevoerd binnen de organisatie 		
<p>Zijn er verschillen per niveau?</p> <ul style="list-style-type: none"> - <i>Delivery/ management/ stuurgroep</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in de verschillende niveaus 		
<p>M.b.t. huidige wijze van besluitvorming:</p> <ul style="list-style-type: none"> - Is dit goed ingericht? - Doorvragen → hoe kan dat? Evt. voorbeelden? 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Mate bepalen waarin aspect van invloed is op het verloop van de software delivery. 		

3. Hoe is de planning en voortgangsrapportage ingericht m.b.t. de software delivery?	
Vragen	Doel
<p>Hoe wordt er gepland?</p> <ul style="list-style-type: none"> - <i>Methodiek/ technieken?</i> - <i>Value driven versus plan driven?</i> - <i>Planningshorizon/ -interval?</i> - <i>Gebruik milestones?</i> - <i>Hoe flexibel/ star is planning?</i> - <i>Hoe wordt er omgegaan met verandering?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Compleet beeld m.b.t. wijze van plannen <div style="border: 1px solid black; background-color: #e0e0e0; padding: 5px; display: inline-block;"> <p>Agile value Responding to change over following a plan</p> </div>
<p>Hoe wordt voortgang gerapporteerd?</p> <ul style="list-style-type: none"> - <i>Mondeling/ rapportages?</i> - <i>Frequentie?</i> - <i>Bijsturing o.b.v. voortgang?</i> - <i>Aan wie wordt er gerapporteerd en wat wordt er mee gedaan?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Compleet beeld m.b.t. wijze van voortgangscontrole
<p>Zijn er verschillen per niveau?</p> <ul style="list-style-type: none"> - <i>Delivery/ management/ stuurgroep</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in de verschillende niveaus
<p>M.b.t. huidige inrichting/ gebruik planning en voortgangsrapportage:</p> <ul style="list-style-type: none"> - Is dit goed ingericht? - Doorvragen → hoe kan dat? Evt. voorbeelden? 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Mate bepalen waarin aspect van invloed is op het verloop van de software delivery.

4. Hoe worden de resources gemanaged m.b.t. de software delivery?	
<i>Vragen</i>	<i>Doel</i>
Soort van resources, anders dan geld? <ul style="list-style-type: none"> - <i>Personeel?</i> - <i>Specialisten?</i> - <i>Anders?.... (bijv. hardware, werkplekken, etc)</i> <i>Welke resources zijn kritiek?</i>	Uitkomst informatie: <ul style="list-style-type: none"> - Beeld van benodigde resources
Hoe wordt personeelstoewijzing gemanaged? <ul style="list-style-type: none"> - <i>Permanente indeling personeel/ teams?</i> <i>(werk naar teams versus teams naar werk)</i> - <i>Verandering in teams?</i> - <i>Voldoende personeel beschikbaar?</i> <i>(kwalitatief& kwantitatief)</i> 	Uitkomst informatie: <ul style="list-style-type: none"> - Compleet beeld m.b.t. wijze van resource management
Hoe worden andere resources gemanaged? <i>(zoals hardware, werkplekken, etc.)</i>	
Zijn er verschillen per niveau? <ul style="list-style-type: none"> - <i>Delivery/ management/ stuurgroep</i> 	Uitkomst informatie: <ul style="list-style-type: none"> - Inzicht in de verschillende niveaus
M.b.t. huidige resource management: <ul style="list-style-type: none"> - Is dit goed ingericht? - Doorvragen → hoe kan dat? Evt. voorbeelden? 	Uitkomst informatie: <ul style="list-style-type: none"> - Mate bepalen waarin aspect van invloed is op het verloop van de software delivery.

5. Hoe vindt budgetteren en kosten beheer plaats m.b.t. de software delivery?	
Vragen	Doel
<p>Hoe loopt het geld door de organisatie?</p> <ul style="list-style-type: none"> - <i>Waar zijn welke financiële bevoegdheden en verantwoordelijkheden belegd?</i> - <i>Budget en scope management afgestemd?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Beeld van waar financiële bevoegdheden en verantwoordelijkheden zijn belegd.
<p>Hoe wordt er om gegaan met toewijzing van budgetten en beheer van kosten?</p> <ul style="list-style-type: none"> - <i>Hoe worden budgetten toegewezen?</i> - <i>Ruimte voor inbreng m.b.t. bepalen budgetten?</i> - <i>Hoe worden kosten beheerst? (rapportages, frequentie van rapporteren, etc)</i> - <i>Wat als budgetten niet voldoen of worden overschreden?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Compleet beeld m.b.t. toewijzing budgetten en beheer van kosten <div style="border: 1px solid gray; background-color: #f0f0f0; padding: 5px; margin-top: 10px;"> <p>Wat voor entiteit? Bijvoorbeeld Projecten/ producten/ functionaliteiten/ teams, etc Hoe worden uren geschreven? Bijv. - Op project/ werkpakket/ geen uren registratie, etc</p> </div>
<p>Zijn er verschillen per niveau?</p> <ul style="list-style-type: none"> - <i>Delivery/ management/ stuurgroep</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in de verschillende niveaus
<p>M.b.t. huidige wijze van budgetteren en kosten beheer:</p> <ul style="list-style-type: none"> - Is dit goed ingericht? - Doorvragen → hoe kan dat? Evt. voorbeelden? 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Mate bepalen waarin aspect van invloed is op het verloop van de software delivery.

6. Hoe vindt de onderlinge afstemming plaats rondom de software delivery?	
Vragen	Doel
<p>Welke teams-/ afdelingen rondom software delivery hebben een afhankelijkheid?</p> <ul style="list-style-type: none"> - <i>Andere software deliveries?</i> - <i>Binnen de huidige software delivery?</i> <p>Wat voor afhankelijkheden?</p> <ul style="list-style-type: none"> - <i>M.b.t. (deel)producten, werkspraken, etc</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Beeld van onderling afhankelijkheden
<p>Hoe vindt onderlinge afstemming plaats?</p> <ul style="list-style-type: none"> - <i>Afspraken/ coördinatie maatregelen?</i> - <i>Onderlinge samenwerking en coördinatie?</i> - <i>Invloed inbreng van hoger niveau?</i> - <i>Is er sprake van zelf organiserende teams? Zo ja, tot waar?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in hoe onderling wordt afgestemd
<p>Is er iemand verantwoordelijk voor integratie/ onderlinge afstemming?</p> <ul style="list-style-type: none"> - <i>Verschil per niveau?</i> - <i>Wie grijpt erin, als het niet loopt zoals het zou moeten?</i> 	
<p>M.b.t. huidige onderlinge afstemming:</p> <ul style="list-style-type: none"> - Is dit goed ingericht? - Doorvragen → hoe kan dat? Evt. voorbeelden? 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Mate bepalen waarin aspect van invloed is op het verloop van de software delivery.

DEEL 3- Aspecten m.b.t. uitvoering (15 min)

7. Hoe kan de management stijl rondom de software delivery worden geduid?	
Vragen	Doel
<p>Waar liggen bepaalde stukken mandaat gegroepeerd? Wat zijn invloedrijke functionarissen binnen de organisatie?</p> <ul style="list-style-type: none"> - <i>Wat voor functionarissen?</i> - <i>Wat voor invloed?</i> - <i>Hoe is dat duidelijk?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht krijgen in management functionarissen/ niveaus binnen organisatie?
<p>Hoe oefenen zij hun invloed uit?</p> <ul style="list-style-type: none"> - <i>Informeel/ formeel?</i> - <i>Sociaal/ betrokken/ afstandelijk?</i> - <i>Mens-/ resultaat gericht?</i> - <i>Hoge mate van controle?</i> - <i>Flexibiliteit?</i> - <i>Goed geïnformeerd?</i> - ... - <i>Past dit bij de manier van werken? (situationeel leiderschap)</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht krijgen in de wijze van leidinggeven - Indruk krijgen in hoeverre
<p>Heb jij een leidinggevende rol?</p> <ul style="list-style-type: none"> - <i>Hoe zou jij de managementstijl duiden? (zie hierboven)</i> 	
<p>Zijn er verschillen per niveau?</p> <ul style="list-style-type: none"> - <i>Delivery/ management/ stuurgroep</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in de verschillende niveaus
<p>M.b.t. huidige management stijl:</p> <ul style="list-style-type: none"> - Is dit goed ingericht? - Doorvragen → hoe kan dat? Evt. voorbeelden? 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Mate bepalen waarin aspect van invloed is op het verloop van de software delivery.

8. Op welke wijze wordt er gedocumenteerd?	
<i>Vragen</i>	<i>Doel</i>
<p>Hoe wordt er gedocumenteerd omtrent de software delivery?</p> <ul style="list-style-type: none"> - Wat voor documentatie? - Hoe frequent? - Formele/ informele documenten? - Etc 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in wijze van documenteren en hoe dit wordt ervaren <div style="border: 1px solid black; background-color: #e0e0e0; padding: 5px; margin-top: 10px;"> <p><u>Agile value</u> Working product <i>over</i> comprehensive documentation?</p> </div>
<p>Wat vindt jij van de wijze van documenteren?</p> <ul style="list-style-type: none"> - Te veel/ te weinig/ precies goed - Omdat... 	
<p>Zijn er verschillen per niveau?</p> <ul style="list-style-type: none"> - <i>Delivery/ management/ stuurgroep</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in de verschillende niveaus
<p>M.b.t. huidige wijze van documenteren:</p> <ul style="list-style-type: none"> - Is dit goed ingericht? - Doorvragen → hoe kan dat? Evt. voorbeelden? 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Mate bepalen waarin aspect van invloed is op het verloop van de software delivery.

9. Hoe kan de organisatie cultuur binnen en rondom de software delivery worden beschreven?	
Vragen	Doel
<p>Hoe zou je de werk cultuur binnen de organisatie willen beschrijven?</p> <ul style="list-style-type: none"> - Formeel/ informeel - Structuur/ chaos - Hiërarchisch/ plat - Saamhorigheidsgevoel - Omgangsvormen - Waardes <ul style="list-style-type: none"> a. Integriteit, eerlijkheid b. Collegialiteit c. Loyaliteit - Geschreven en ongeschreven regels? <p>Past de huidige organisatie cultuur bij de wijze van werken? (agile) -----></p>	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht krijgen in de organisatie cultuur <div style="border: 1px solid gray; background-color: #f0f0f0; padding: 5px; margin-top: 10px;"> <p><u>Agile values</u></p> <ul style="list-style-type: none"> - Individuals & interaction <i>over</i> processes & tools - Customer collaboration <i>over</i> contract negotiation </div>
<p>Zijn er verschillen per niveau?</p> <ul style="list-style-type: none"> - <i>Delivery/ management/ stuurgroep</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in de verschillende niveaus
<p>M.b.t. huidige organisatiecultuur:</p> <ul style="list-style-type: none"> - Is dit goed ingericht? - Doorvragen → hoe kan dat? Evt. voorbeelden? 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Mate bepalen waarin aspect van invloed is op het verloop van de software delivery.

DEEL 4- Aspecten m.b.t. uitvoering (10 min)

10. Hoe is het personeelsplan ingericht rondom de software delivery?	
Vragen	Doel
Sluit het personeelsplan goed aan op de software delivery? <ul style="list-style-type: none"> - <i>Juiste mensen, op de juiste plaats?</i> - <i>Zou je dingen anders willen zien?</i> 	Uitkomst informatie: <ul style="list-style-type: none"> - Inzicht in personeelsplan in relatie met software delivery - Inzicht in functiehuis in relatie met software delivery
Ben je bekend met het <i>functiehuis</i> van de organisatie? <ul style="list-style-type: none"> - <i>Komen de functiebeschrijvingen overeen met de huidige wijze van werken?</i> - <i>Zijn de benodigde competenties goed beschreven?</i> 	
Zijn er verschillen per niveau? <ul style="list-style-type: none"> - <i>Delivery/ management/ stuurgroep</i> 	Uitkomst informatie: <ul style="list-style-type: none"> - Inzicht in de verschillende niveaus
M.b.t. huidige personeelsplan: <ul style="list-style-type: none"> - Is dit goed ingericht? - Doorvragen → hoe kan dat? Evt. voorbeelden? 	Uitkomst informatie: <ul style="list-style-type: none"> - Mate bepalen waarin aspect van invloed is op het verloop van de software delivery.

11. Hoe wordt personeel beloond en beoordeeld?	
Vragen	Doel
<p>Hoe werkt het functionerings- en beoordelingssysteem?</p> <ul style="list-style-type: none"> - <i>Wie beoordeelt wie?</i> - <i>Hoe frequent?</i> - <i>Gebeurt dit voldoende?</i> - <i>Hoe wordt er omgegaan met externen?</i> - <i>Denk je dat dit goed is ingeregeld?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in de wijze van beoordelen
<p>Hoe wordt er beloond?</p> <ul style="list-style-type: none"> - <i>Zijn er beloningssystemen?</i> - <i>Mogelijkheid tot differentiatie?</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in de wijze van belonen
<p>Zijn er verschillen per niveau?</p> <ul style="list-style-type: none"> - <i>Delivery/ management/ stuurgroep</i> 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Inzicht in de verschillende niveaus
<p>M.b.t. huidige wijze van belonen en beoordelen:</p> <ul style="list-style-type: none"> - Is dit goed ingericht? - Doorvragen → hoe kan dat? Evt. voorbeelden? 	<p>Uitkomst informatie:</p> <ul style="list-style-type: none"> - Mate bepalen waarin aspect van invloed is op het verloop van de software delivery.

DEEL 5- Afsluiting (10 min)

Questionnaire
Elke geïnterviewde is gevraagd om de questionnaire in bijlage B in te vullen. Hierbij een score toe te kennen aan de mate van frictie die men ervaart omtrent de aspecten die zijn besproken.
Afronding/ overige zaken
Afsluiting interview. Overige opmerkingen, vragen of toevoegingen vanuit geïnterviewde? Toelichting verdere procedure. Interview rapport wordt opgesteld en zal binnen maximaal 2 werkdagen naar geïnterviewde worden gestuurd ter verificatie en validatie. Verzoek aan geïnterviewde om interview rapport binnen 1 werkweek van commentaar te voorzien.

B. Questionnaire to score aspects

Omtrent agile software delivery kan frictie ervaren worden met de organisatie daaromheen.
Tijdens het interview zijn onderstaande elf aspecten besproken.

Geef per aspect aan in hoeverre u frictie ervaart omtrent dit aspect.

1 = Geen frictie, 5 = veel frictie

Stelling:	<i>1</i>	<i>2</i>	<i>3</i>	<i>4</i>	<i>5</i>	<i>Weet niet/ geen mening</i>
1. Verdeling taken, bevoegdheden en verantwoordelijkheden						
2. Besluitvorming						
3. Planning & voortgangsrapportage						
4. Resource management						
5. Budgetteren en kosten beheer						
6. Onderlinge afstemming						
7. Management stijl						
8. Documenteren						
9. Organisatie cultuur						
10. Personeelsplan						
11. Beloning & beoordelingen						

C. Scoring of organizational aspects

1. Verdeling TBV		1	2	3	4	5	Geen Mening	Score	52,5	3,3
1	PM							4		
1	SM							2		
2	PM							5		
2	PO							5		
3	PM							3,5		
3	SM							2		
4	PM							3		
4	SM							2		
5	SM							4		
5	PO							4		
6	PO							1		
6	PL							2		
7	AC							4		
7	PM							4		
8	HA							3		
8	PO							4		
2. Besluitvorming		1	2	3	4	5	Geen Mening	Score	33,5	2,1
1	PM							3		
1	SM							1,5		
2	PM							3		
2	PO							3		
3	PM							2		
3	SM							2		
4	PM							3		
4	SM							2		
5	SM							3		
5	PO							4		
6	PO							1		
6	PL							2		
7	AC							1		
7	PM							2		
8	HA							2		
8	PO							2		
3. Planning & voortgangscntrole		1	2	3	4	5	Geen Mening	Score	32,5	2,0
1	PM							1		
1	SM							3		
2	PM							2		
2	PO							3		
3	PM							1		
3	SM							2		
4	PM							2		
4	SM							2		
5	SM							2		
5	PO							3		
6	PO							2		
6	PL							3		
7	AC							2		
7	PM							2		
8	HA							1		
8	PO							2,5		
4. Resource mgmt		1	2	3	4	5	Geen Mening	Score	31,5	2,1
1	PM							4		
1	SM							3		
2	PM							3		
2	PO							1		
3	PM							4		
3	SM							1,5		
4	PM							1		
4	SM							1		
5	SM							2		
5	PO						X			
6	PO							3		
6	PL							1		
7	AC							4		
7	PM							3		
8	HA							2		
8	PO							2		
5. Budgetteren en kosten beheer		1	2	3	4	5	Geen Mening	Score	38,5	2,8
1	PM							1		
1	SM						X			
2	PM							2		
2	PO							4		
3	PM							3		
3	SM							1,5		
4	PM							1		
4	SM							1		
5	SM							4		
5	PO						X			
6	PO							4		
6	PL							3		
7	AC							5		
7	PM							4		
8	HA							3		
8	PO							3		

	1	2	3	4	5	Geen Mening		
6. Onderlinge afstemming								41,5
1 PM								4
1 SM								1,5
2 PM								5
2 PO								5
3 PM								3
3 SM								2
4 PM								1
4 SM								2
5 SM								2
5 PO								3
6 PO								3
6 PL								2
7 AC								4
7 PM								4
8 HA								2
8 PO								2
7. Management stijl								36,5
1 PM								3
1 SM								2
2 PM								2
2 PO								3
3 PM								1,5
3 SM								2
4 PM								1
4 SM								4
5 SM								4
5 PO								4
6 PO								2
6 PL								1
7 AC								2
7 PM								3
8 HA								3
8 PO								2
8. Documenteren								33,5
1 PM								1
1 SM								3
2 PM								2
2 PO								2
3 PM								1
3 SM								2
4 PM								1
4 SM								2,5
5 SM								5
5 PO								1
6 PO								3
6 PL								1
7 AC								2
7 PM								2
8 HA								2
8 PO								4
9. Organisatie cultuur								39,0
1 PM								4
1 SM								2
2 PM								2
2 PO								3
3 PM								3
3 SM								2
4 PM								2
4 SM								1
5 SM								4
5 PO								5
6 PO								2
6 PL								3
7 AC								1
7 PM								2
8 HA								3
8 PO								4
10. Personeelsplan								18,5
1 PM								1
1 SM								1,5
2 PM								2
2 PO								1
3 PM	-	-	-	-	-	-	X	
3 SM	-	-	-	-	-	-	X	
4 PM								1
4 SM								2
5 SM								3
5 PO								2
6 PO								2
6 PL								1
7 AC	-	-	-	-	-	-	X	
7 PM								2
8 HA								1
8 PO	-	-	-	-	-	-	X	
11. Beloning & beoordelingen								21,0
1 PM								5
1 SM								1
2 PM								2
2 PO								1
3 PM	-	-	-	-	-	-	X	
3 SM	-	-	-	-	-	-	X	
4 PM								1
4 SM								3
5 SM								4
5 PO								2
6 PO								2
6 PL								1
7 AC	-	-	-	-	-	-	X	
7 PM								2
8 HA								2
8 PO	-	-	-	-	-	-	X	

