Vertically Scaling Agile
- A Multiple-Case Study

Rasmus Lindström
Nicklas Östman

Supervisors: Ola Leifler & Mats Jegebo
Examiner: Simin Nadjm-Tehrani
Upphovsrätt


Copyright

The publishers will keep this document online on the Internet – or its possible replacement – for a period of 25 years starting from the date of publication barring exceptional circumstances. The online availability of the document implies permanent permission for anyone to read, to download, or to print out single copies for his/her own use and to use it unchanged for non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional upon the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility. According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement. For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its www home page: http://www.ep.liu.se/.

© Rasmus Lindström
© Nicklas Östman
Abstract

The conceptual framework of agile software development is an ever-growing movement in the software industry. However, recent studies have shown that large, less software-focused companies, where software development is primarily used for in-house IT-solutions, struggle with giving up traditional command-control type of management. This hits hard on some of the most important principles of agile software development and in many cases this phenomenon has inevitably led to large gaps between development teams and more managerial parts of the organization. This thesis has aimed to study this gap and investigate how it affects software development teams’ ability to carry out their work.

By comparing three software teams that were internally highly similar but with varying external conditions, impact on the teams’ behaviour based on their different environments was studied. The study was carried out using a multiple-case study approach with primary data sources consisting of survey gathered data from all team members and interviews with a subset of the team members. The results gathered from this study suggest that agile development teams are extremely dependent on a well-functioning interface to business related parts of an organization. Regarding teams’ ability to make decisions and being agile in their way of working, the results primarily isolate impediments with roots in an unwillingness to adhere to and lack of understanding of agile principles.

In this thesis, our gathered results were also correlated with a modern framework called Flow in order to confirm its relevance regarding analyzing software development teams in large-scale environments.
Acknowledgments

First of all, we would like to thank our examiner Simin Nadjm-Tehrani for valuable input both before and during this thesis. We would also like to thank both our supervisors Ola Leifler at Linköping University and Mats Jegebo at mPeira. Ola for his valuable input all throughout the writing of this thesis. Without your input a lot of important information would not have made it within the thesis, and some other less important information would have. Thank you also for the kick in the butt in the later parts of this thesis and the moral support throughout the work. Mats for the long evening meetings, helping us concretize a topic for this thesis that interests us both as well as the great help in finding teams that we were allowed to study. Thank you also for the hospitality, always trying to help and last but not least sharing all your wisdom and inspiring thoughts around the topic of agile software development.

Also, huge thanks to our parents for supporting us whenever we hit a road bump along the way. Last but not least a big thank you to Josefine (Nicklas’s girlfriend) for letting us spend so much time on this thesis, and always prioritizing the finalization of this report above her own plans.

The forthcoming book of Flow was made available in manuscript form for this thesis by Ted Kallman and Andrew Kallman. The book itself is scheduled to be released second half of 2017.

Rasmus Lindström & Nicklas Östman  
Stockholm, January 2017
List of Figures

1.1 Challenges in achieving flow ........................................ 4

2.1 Simplified illustration of horizontally scaling agile methods

   ASDT - Agile Software Development Team .......................... 8

2.2 Simplified illustration of vertically scaling agile methods ........ 9

5.1 Project organization team A ........................................... 31

5.2 Years of software development experience ........................... 34

5.3 Years of agile software development experience ....................... 34

5.4 Project organization team B ........................................... 35

5.5 Years of software development experience ........................... 37

5.6 Years of agile software development experience ....................... 37

5.7 Project organization team C ........................................... 39

5.8 Years of software development experience ........................... 41

5.9 Years of agile software development experience ....................... 41

5.10 Perceived ability for self-organizing in teams ....................... 42

5.11 Perceived ability for High-bandwidth communication in team .......... 44

5.12 Perceived ability for Customer collaboration in teams ................ 45

5.13 Perceived ability to prioritize in teams .............................. 47

5.14 Perceived uncertainties and conflicts in all teams .................. 48

5.15 Perceived ability for shared decision-making in teams .............. 50

6.1 Perceived uncertainties and conflicts in all teams .................. 65
List of Tables

2.1 Table of indicators ......................................................... 16
4.1 Survey demographics ..................................................... 26
4.2 Interviewee codes used for referencing interview citations ........ 27
4.3 Interview processing codes ............................................. 29
Introduction

Effective software development is a matter that has been going through numerous revolutions during the last three to four decades. The matter is driven by the goal of higher software quality, innovation and shorter lead times. This process has produced one of today’s most popular frameworks; Agile software development.

Agile software development is a conceptual framework first presented in the 2001 Agile Manifesto [40], which has more or less been the base for all the different practices commonly seen in software development such as Scrum, Kanban or Extreme programming. The manifesto however, has little to tell about scalability or adaption to business characteristics.

The claimed benefits and organizational structures linked with agile development, compared to other development methods, have boosted its popularity and agile development has been correlated to both higher project success rate [33] and higher productivity [39]. This has arguably given the framework the status as the most prominent choice. However, the growth of the method has been, in some perspectives, faster than its evolution and companies sometimes adopt agile methods purely based on the fact that it is "the most prominent choice", and not that it necessarily suits the business needs.

Large-scale agile development can be interpreted as the presence of agile development methods in large organizations with complex infrastructures, where large-scale refers to the size or complexity of the software development context[11]. Large-scale agile development can also be referred to as large projects (many teams) holistically embracing agile development and therefore the scaling of agile practices themselves[11]. Practices which originally were aimed for projects with a single-team setup. A large-scale application of agile development often clashes with fundamental agile principles such as self-management, since this principle in many ways can hinder much needed communication between teams and the rest of a project organization[11]. On the other hand, many large organizations adopting agile methodologies see the practices as misfits in the current infrastructure and rather than adapting the business for agile development, agile development practicing parts of the organizations have to adapt to the waterfall type of management thus causing a noxious effect to the adaptability and flexibility (i.e. agility) sought out by agile methodologies[43].
The focus of this thesis will be on the gap caused by the isolation of teams when agile software development meets enterprise. An isolation that has been found hindering teams’ ability to perform, both as an entity in an agile way and making decisions in alignment with business goals [2]. This disconnection of teams and business has become an area of interest for research [10]. Additionally, frameworks for inter-communication of agile software development teams and surrounding business are rapidly emerging on the commercial market.

Through a multiple-case study of three different software development teams with a broad survey and in depth interviews, this thesis ultimately aims to investigate the relevance of one of these emerging frameworks. This framework is called Flow.

1.1 Flow

Flow is a framework described in a book [17] with the same name, written by Andrew Kallman along with his brother Ted Kallman. Both brothers have experience working as management consultants, and the framework is based on lessons made during their careers. The framework has gotten wide support within the software community and is believed by many to increase the chances of successful projects.

The term *flow* is described, by the authors, as a *hyper performing state in which all organizations want to be*. There are infinite ideas, probably one for each management consultant that exists, on how an organization should act to become better at what they do. In the book, they use the following words to define Flow:

"The state of optimal performance achieved by applying a clear, consistent, persistent and unified vision at all levels of an organization".

The concept the authors describe is simple, achieving it is however not as simple. Vision, according to the authors, is key for any project, independent from what the project is trying to achieve. Without a clear and well defined vision the project will probably end up with anarchy and a lack of order. By cascading the organization’s vision down the hierarchy levels, from the board of directors down to the individual teams, Flow is claimed to facilitate achieving a *hyper productive state*.

To achieve *flow*, according to the authors, a set of criteria for six parameters is vital. The six parameters are: **align Vision, Right People, Define, Distill, Deliver and Drive**. Below is an explanation of why the parameters are needed to achieve a hyper performing state.

1. **Align Vision**: If the team is not led, pushed and pulled by vision the team will be driven in multiple directions, all at once, depending on circumstances. The goals have to align with the overall company goals.

2. **Right People**: Without the right competences within the team, the possibility to reach a hyperactive state is slim to none.

3. **Define**: Prioritize what is going to be done and minimize the amount of noise from external parts.

4. **Distill**: Let the group make decisions in order to reduce incorrect assumptions. Seek consensus in the group. Consensus does not equal unanimity.

5. **Deliver**: Vision without execution does not provide any results. When there is an agreement on what needs to be done, execute it.

---

1 Generic term for external factors negatively impacting the team members ability to carry out daily work
6. **Drive:** Continue driving the project towards its goal. There might be a need to reiterate step 1-3 in order to get a sustainable success.

![Figure 1.1: Challenges in achieving flow](image)

The figure above is used by Kallman and Kallman[17] in their book to clarify what the different outcomes are when one of the six parameters is not fulfilled. When vision is missing, there is a risk for anarchy within the project. When the project does not have the right resources there’s a high risk for anxiety etc. According to the authors, flow is not achievable if not all of the parameters are met.

The topic of shared vision management is not a new thing within business. Wang and Rafiq[42] express the need of a shared-vision to get the most out of a workforce:

> "Without a shared vision, the reality of a firm would be characterized by highly enthusiastic and committed individuals pulling the organization toward different directions."

Christenson and Walker[5] are arguing for the need of a clear project vision within the project in order to motivate the project members:

> "We argue that the key to developing an effective project vision is to make objectives and purpose clearly understood, to inspire motivation, and to ensure that the project vision is credible and challenging."

Kallman and Kallman[17] goes even further regarding the role of shared vision. They argue that creating a shared-vision is top priority for management in an agile software development team:

> "Vision is always the key Driver for what a project is trying to achieve for its product, service or result (i.e. increase revenues, decrease costs and/or get rid of or mitigate risk). If there is no project vision, your job as a leader is to define and distill agreement on what the vision really is (or, should be), with the appropriate stakeholders. Understand and articulate the “why” for your team."

### 1.2 Motivation and Problem statement

People within the Flow community, and early adopters of Flow, have expectations that software development teams in businesses adopting the Flow framework are functioning better
than other agile software development teams and that the concept in many ways eases problems connected with the gap unveiling when deploying agile methodologies in large organizations. These are problems primarily occurring when higher expectations are set for developers regarding communication, self-management and customer collaboration; the human-centric parts of agile software development. There are also expectations that an adaption to the concepts of Flow will strengthen individual team members’ ability to make quick, justified, decisions and because of that be able to deliver the same work quicker than before and in a larger degree aligned with business intentions. The purpose of this thesis is thus to answer the following questions:

1. Is Flow a valid framework for analyzing human-centric agility of software development teams in large-scale environments?

2. Is Flow a valid framework for analyzing individual developers ability to make justified and/or tactical decisions in large-scale environments?

To answer these questions, found impediments and success factors for agility and decision-making will be mapped against constructs presented in Flow and validity of the analyzing capabilities in the Flow framework is considered as the overlap of these parameters. With a considerable overlap the framework should be treated as valid in an analytic perspective.

1.3 Delimitations

The study will partially be conducted in a qualitative way which means that external factors may have impact on the result. These factors could for example include culture within the company, competence and experience. The study is only going to be conducted in three teams which will, in combination with the qualitative nature, entail a high risk that the result won’t be generalizable. There will however be a thorough reflection about this within the thesis.

Aligning teams with business strategies and vision is of course, in many ways, a managerial issue. This thesis will, however, primarily map correlating effects in team members’ way of working in different projects/settings. Management practices and other parts of the infrastructure will therefore not be a unit of analysis in the case study, but will naturally be reflected on as a part of the case context.

1.4 Research Context

The common denominator of the three studied teams is that they are all in a single-team project setup, meaning that they are not working in cooperation with other teams and at the moment only involved in one project. All three projects are also running under a government owned business. Thirdly, all the teams have adopted agile methodologies, but to different extents. What differs is the surrounding infrastructure, which varies from a very strict traditional organization to an environment where teams are more self-managed.

1.4.1 Team A

Team A is located in Södertälje, Sweden and works within an energy producing company called Söderenergi. The company itself is not, in any way, connected with software development or solutions. However, in an ongoing project at Söderenergi, a large number of internal IT-solutions are to be integrated. This project has been undertaken by Team A and the team has been active in its current form since February 2015. The team consists of 11 members which consider themselves as highly cross-functional in their roles and tasks.
1.4.2 Team B

Team B is an in-house development team working for Trafikverket at their headquarters located in Borlänge, Sweden. There are currently 10 members in the team. The project they are working on is a migration project, migrating a system from an old platform to a newer edition of the same platform. The team itself has adopted agile development, however the surrounding infrastructure has not. An issue the team is currently struggling with is that the surrounding infrastructure is lagging behind in channeling requirements for the project, due to its waterfall nature. Another issue is that some stakeholders of the project are not familiar with agile software development, and therefore have no trust that the method will work. The project has been ongoing for 1.5 years, however small progress has been made as the collection of requirements is not yet finalized, and therefore not given to the team. The project is supposed to have delivered all the basic features before end of 2016.

1.4.3 Team C

Team C is a team consisting of IT consultants from CGI, currently working on a project for Trafikverket. The team is located in Östersund, Sweden, geographically far from where Trafikverket headquarters are located. The team is not employed by Trafikverket, but are by contract supposed to work as if they were employed directly by Trafikverket, a so-called co-sourcing team. Team B is currently working on developing a new driver’s license theory test. The team consists of 10 members and Scrum is the development method used. Team B has worked with Scrum since the project was launched in September 2015. The project is estimated to be finished before the end of 2017. This project is governed by a management-team counterpart situated at Trafikverket, which also have adopted an agile way of working in order to mitigate some of the managerial issues connected with agile development.

1.5 Thesis structure

The first chapter of this thesis includes a short introduction, both to why scaling agile software development is often an issue, and to Flow, the framework being investigated. The introduction also includes a motivation to the chosen research questions, as well as a short description of the study context.

The second chapter includes theory from different areas and will act as the theoretical reference in our case study. The literature presented will cover topics such as agility measurement, shared decision-making and the importance of vision within organizations. It will also cover results of related work done within the topic, and other related work with significant value to our research.

The third chapter includes a thorough description of similar or related studies and specifically how these studies have been carried out. The purpose of this chapter is primarily to justify the choice of methods in this thesis but also to provide the reader with a better understanding of approaches used in the research area.

The fourth chapter describes in great detail how the work in the study has been carried out. The main focus is on how data has been collected throughout the thesis. It also covers theory on how to conduct a case study, a survey, interviews and best practices for making observations.

In the fifth chapter, the results of the data collected throughout the case study is presented.
1.5. Thesis structure

The **sixth** contains a cross-case analysis of the results.

The **seventh** includes a discussion part which highlights areas as why the methods used were chosen, what limitations they may have had in the study and what the found results might indicate. This chapter also includes discussions regarding the wider context of the study such as ethical and societal aspects.

In the **eighth** and final chapter, the conclusions made are presented. The chapter will also mention future research of interest.
The topics in the theoretical chapter of this thesis are focused on issues that often occur when agile software development methodology is used in large-scale environments and how the authors of Flow claim their framework could mitigate these issues. It will also cover literature on how to measure agility, different decision making theories and what obstacles software development teams often encounter regarding these concepts.

2.1 Agile in a Large Context

As briefly mentioned in the introduction, there are essentially two broader views on agile development in large organizations. The first one is the horizontal scaling, where large projects are classified in portfolios consisting of several teams jointly working towards the same end product. In practice, this often means that different teams are responsible for different parts or features of an end product, which will be integrated at a later stage. This kind of scaling is simply done from the original single-team structure of agile methodologies to a multiple-team approach (figure 2.1). This is without a doubt the most popular research topic for agile practices in large organizations [11].

Figure 2.1: Simplified illustration of horizontally scaling agile methods

ASDT - Agile Software Development Team
2.1. Agile in a Large Context

However, with agile development becoming more and more popular \[43\] and slowly moving towards an industrial standard for software development, the second view of agile in a large organization has become more evident; the vertical scaling. Vertical scaling is referred to, throughout this report, as the adoption of agile principles and values in higher hierarchic layers of an organization (figure 2.2).

![Figure 2.2: Simplified illustration of vertically scaling agile methods](image)

This type of scaling is becoming more evident since rapid changes in market forces organizations to holistically adopt more flexible project methods\[41\]. However, Large organizations do tend to move slowly and a transition to agile methods has often resulted in problems in the borderland between agile software development and more traditional sequential management methods.

The founders of the Flow framework recognize this gap and mean that it existed even before the concept of agile methodologies became popular. They further mean that the demands on self-management and individual decision-making, that come with agile development, have made the problem more evident than ever before\[17\].

As briefly mentioned in the introduction, there are many other frameworks aimed at bridging this gap such as SAFe\[1\] or Management 3.0\[2\]. Kallman and Kallman\[17\] mean that these tools are too dependent on a successful, by-the-book implementation of agile methodologies at the team-level. According to the them, those conditions rarely exist in large organizations where many problems with agile methodologies can occur.

Isolation from organization

Agile development methodologies encourage self-management and more informal communication. Team isolation from surrounding infrastructure can therefore become evident\[9\]. This type of problem can appear when organizations implement agile by the book\[9\] and therefore give teams a great deal of autonomy. The autonomous nature can result in teams diverting from organizational goals and prioritizing their own goals\[3\].

This can obviously become a serious issue in a larger context and as reported by Cockburn et al\[6\], plan-altering decisions made by team members based on interpretations of customer requests or due to technology platform changes could often be admonished by management. In fear of these turns of events, a counterproductive need of control from management can become evident\[3\].

\[1\]http://www.scaledagileframework.com/

\[2\]https://management30.com/about/
Water-scrum-fall

What many less software-focused companies (where software development is used primarily for in-house IT-solutions) often struggle with, is giving up the traditional command-control type of management. Management parts of an organization are simply reluctant to accept the change needed to successfully adopt agile methods.

For example, as pointed out relatively early in the agile evolution by Boehm et al., life-cycles of agile approaches in teams and traditional waterfall approaches in management often clash. The clash often appears as agile teams strive for informal requirements and early iterative releases of software, while the surrounding management is used to an extensive design phase and a complete, concrete list of requirements before the project goes into production.

The work done by West et al. and Bannink confirms the existence of this problem and how it often resulted in a “water-scrum-fall” infrastructure, where agile practices that fit with underlying processes are the only ones getting implemented. An example is the “water-scrum-fall” infrastructure where a complete set of detailed requirements and tasks is handed over up front to an agile development team which are expected to produce deliverables in chunk releases. Their research further elaborates on how this phenomenon causes impediments for team-level agility. They also explain how these impediments are foremost caused by the lack of acceptance and adoption to the agile ways of working from management levels in the organization.

Kallman and Kallman also explain that many advocates of agile methodologies suggest that the problem mentioned above has its roots in managements unwillingness to adapt to agile methodologies and that the solution simply is to overcome that unwillingness. They suggest that the sequential waterfall-like management is needed for structure in large organizations and therefore argue that mixing agile methodologies with more traditional project management is a problem that goes both ways. They further elaborate on the issue and mean that an organization being “truly” agile does not imply practicing “team-level agile” (scrum or similar framework) throughout a project organization, but rather to embrace the core values of agile thinking. This ultimately means that practitioners of agile methods must be open to understanding and to synchronize with more traditional project management methods. As described in the book:

“...we have observed that some Agile Coaches wear this type of removal (of traditional management methods) as a badge of honor. Every time that happens, it drives another nail in the coffin for any one attempting to scale Agile to higher levels of an organization”

Managing projects in an agile manner

The practices connected with the agile approach for software development have however started to spread into higher layers of organizations hierarchies. Agile management is simply the adherence of management to agile values and practices. For example: supporting short, iterative evolution of projects, less granular requirements and reduced formality in processes.

The research done by Serrador and Pinto elaborates on the need for and potential of management alignment with agile values and principles. In their study, they found that project management holistically embracing an agile type of management were correlated to higher project success rate. They also argue that the quality of vision and goals is one of the most important moderators for a high success rate in managing agile projects. To reach the potential of agile development and aligned management, Bannink argues that speaking the same language between hierarchic layers in the organization is another
important success factor. As Bannink and Serrador et al, the authors of Flow point out these moderators as important and especially vision as the most important factor for successfully managing agile projects.

There are a lot of challenges facing the traditional type of project management in dynamic environments. Collyer et al [7] isolated some of these challenges and classified them according to three types of changes in a project which creates the challenges. The three types are; changing goals, changing resources or changing dependencies on related projects or services.

To sum up, a common language and a common vision are claimed to be two of the most important tools for successfully mitigating these challenges.

2.2 Vision based management

Kouzes et al [21], Walker [5] and Wang et al [42] all agree that a shared vision is vital for a company’s success. From their study regarding key success factors in leadership, with almost one million respondents, Kouzes et al [21] conclude that:

“what leaders struggle with most is communicating an image of the future that draws others in — that speaks to what others see and feel.”

Walker [5] in his article about why vision is important for project success states that:

“The most significant success factor for project teams is that they have a common and shared idea of what difference they are trying to make as a result of the project. Such a vision can be built up by exploring questions with stakeholders and project team members”

Wang et al [42] has a similar understanding and states that the existence of a shared vision helps guide the developers in which creative ideas to pursue, and which to let go of, all depending on the overall organizational objectives.

Serrador and Pinto [33] has a similar view of the importance of a clear vision within projects:

“We further found evidence that the quality of the vision and goals for the project can serve as a significant moderator of the relationship between Agile methods and project success”.

He also argues that the main benefits of a shared vision is to make purpose and objectives clear, to motivate, and to challenge the employees.

Wilson [45] use another approach when he describes what a vision has to include. In his article, he states that the vision has to cover the following:

1. Must be coherent and integrate goals
2. Must be powerful
3. Must include how the organization could be
4. Must include how the organization should be

A precise vision including the above parameters, will translate, according to Wilson, into improved customer satisfaction, sustained competitive advantage and higher employee commitment. He also states that since we are living in a rapidly changing environment with low predictability, the need for a shared vision within companies is even greater. Along with the rapidly changing environment, the flexibility and speed in individual decision-making
has become a more crucial part of business.

As mentioned, the authors of Flow mean that the vision will always be the key driver for any project and that it is very typical for large organizations to believe that strategies and ultimately vision is "automatically" linked to people and their actions \cite{17}. To reach a setting where the vision works as a backbone in a project, Kallman and Kallman mean that project organization needs to work through a "define and distill" process. This process, simply put, means that the whole project organization, from a development team to project management and executive roles, defines a common language. They further mean that this process eliminates the previously mentioned state of isolation and therefore facilitates a better functioning vertical scaling of agile principles.

Formulated in the Flow framework, a failed communication of a project’s vision will always result in the work being driven by circumstance.

2.3 Naturalistic decision making

Naturalistic decision making is a theory which is used to understand how humans make decisions in their lives that are both meaningful and familiar to them. Naturalistic decision making, as a theory really shifted the concept of human decision making\cite{27}:

"from a domain independent general approach to a knowledge-based approach exemplified by decision makers with substantial experience"

The theory of naturalistic decision making in many ways expanded what many researchers see as the decision-making process. According to this theory the decision-making process includes perception and recognition of situations and a base for decision making, not a choice from different options based on information \cite{20}.

2.4 Rational & non-rational decision making

In software development both routine and non-routine activities occur frequently. Therefore, rational decision making cannot always be used. In rational decision making the decision maker is assumed to have complete information about the problem before making the decision. After the problem has been thoroughly mapped the options that will maximize the outcome gets chosen. However, this is not always viable in software development, as the information is not always complete when the decision has to be made\cite{27}.

Because of the transition to agile, more of the activities that occur within a software development project can be classified as non-routine ones \cite{27}. When making a decision about a non-routine activity the bounded rationality model is often used. The model itself assumes that the decision makers’ rationality is based on the knowledge the decision maker possesses. When uncertainty increases, it becomes more important to control the output. Controlling output can be done by setting up goals and targets, and by relying on continuous feedback in order to be able to adjust accordingly.

2.5 Decision making in agile development teams

The more self-managed a team (or organization) gets, the more responsibility is put on the individual member of the organization to make decisions and priorities in their work life. A software development team can be incredibly efficient at their work, writing brilliant code with no errors at an incredible speed. However, if the individual or team are not able to make well justified decisions and/or priorities, the team will never be effective in the meaning of
2.5. Decision making in agile development teams

Thus, a problem almost every agile software development team struggles with can be concluded to: how can the developers make sure that they are working on the right things? While the use of an agile development method has many benefits such as being able to respond to new customer requirements in a rapid manner, the usage of agile development can also bring a lot of confusion to the team. The issue gets even more obvious in large agile environments. The risk of decisions made by individuals, not aligning with the organization’s goals increases with every hierarchy layer added between the decision makers (developers) and top management.

The example above is just one example of all the decision-making obstacles that will occur when using agile software development. The rest of this section will aim to explain this and other decision making obstacles when working with an agile development method.

Takeuchi et al [15] argue that self-management within teams can directly influence the team’s effectiveness, as the decision-making authority is given to employees further down in the organization hierarchy. Letting the employees working closer to the problem make the decisions related to the problem, increases the speed and accuracy of problem solving [15].

Regarding decision-making in agile development teams Moe et al [27] has the following to say:

"In agile software development the Product owner is responsible for strategic decisions. However, the project team should also be involved, because the team possesses the relevant technical knowledge. A self-managing team is responsible for shared decision-making on the tactical and operational levels, however the Product owner should also be involved because he or she possesses the business perspective."

Kallman and Kallman [17] argue that there are positive benefits with decisions being made at a lower hierarchic layer as they are in agile software development. However, they also mean that unless people know and understand what the highest priority is, it is impossible for them to direct their focus to the right things and therefore will not be able to execute well and deliver results. They also claim that:

"Continuous prioritization against the Vision allows for quick adaptation to changing circumstances, while maintaining proper concentration on delivering the true value."

2.5.1 Strategic, operational and tactical decisions

Decisions that are taken within a software development projects can be divided into three categories; strategic decisions, operational decisions and tactical decisions. The boundaries between these three categories are not always clear, but they do differ in the information requirements needed to make a justified decision.

**Strategic Decision Making**

Moe et al [27] describe strategic decision making as decisions that relate to the organizational goals and objectives. The information needed for such a decision is often incomplete and therefore the process of making such decisions often extends over a longer period of time. Further back in history these decisions were usually made by middle or senior managers, but as developing teams have become more self-managed over the last years, the decision
2.5. Decision making in agile development teams

Makers of this kind of decision have gone further down into the organizations hierarchy \[23\].

Both Moe et al \[27\] and Drury et al \[12\] agree that a major obstacle for decision making in agile development is lack of clear prioritization. Both studies show that the more customers involved, the harder it gets for the development team to get their prioritization correct. The consequence of having multiple customers involved in the software development process is that there will be conflicting needs, making it hard for both the product owner and the software development team to make decisions about what needs to be implemented when. The multiple customer involvement problem could also have the effect that long term quality requirements get put aside, since it was more important for the product owners of the studied teams to have new features than to have high quality code. Another consequence that changes of plans have on development teams is that the level of commitment to made decisions decreases. Instead of implementing the decision, team members wait for the requirement to change again, before they commit to implementing it.

Tactical Decision Making

Tactical decisions involve the project management view and how to implement strategic decisions the best way. How resources should be allocated is one of the major tactical decisions. Both Moe et al \[27\] and Drury et al \[12\] highlight major issues regarding the allocation of resources. Both studies point out the importance of a static team within each sprint. In both studies team members were removed, either permanently or temporarily, during sprints. This causes agile development teams to have major issues while planning the sprint. If the resources available differ between the planning of the sprint and the implementation of a sprint an already tough challenge (time estimations), becomes an impossible one. Another issue that arises from the allocation of resources problem, is that the teams’ agreed definition of done become unclear. The uncertainty make teams more likely to rush through tasks, not completing them entirely, in order to meet the sprint goals \[12\].

Operational Decision Making

Moe et al \[27\] describe the operational decision as the decisions that “deal with ensuring effectiveness of day-to-day operations within the organization.” Operational decisions in agile software development usually regard the process of assuring that certain tasks are done efficiently and effectively. The development teams usually have the information needed to make decisions regarding this level.

Moe et al \[27\] highlight one major issue regarding decision making on the operational level, challenges about performing tasks together. In both Moe et al \[27\] and Drury et al \[12\] studies the fear of conflicts is a major issue. Without a space to raise problems within the teams the problems that arise become more of a personal issue than one of the team. Another major issue in shared decision making at the operational level is the lack of knowledge regarding certain topics of discussion. While specialization can have its benefits, it is said to hinder shared decision making in agile development teams.

The authors of Flow agree that customer involvement (with a big number of stakeholders) often causes the team members to get confused. They mean that aligning a couple of hundred stakeholders is an impossible task and that it becomes even worse when all stakeholders are trying to agree upon definitions that should be used within the project \[17\]. They also highlight the importance of having a product owner within an agile environment and state that:

"The role of a Product Owner is critical for Scrum or Agile teams to succeed."
2.6 Measuring agility

With little empirical examination of core concepts in the agile practices, implementations have often been supported by anecdotal promises which have resulted in a lack of understanding of the actual implementation objectives. Lee et al. also argue that this lack of understanding has made it hard for organizations to measure and control factors of agile development in order to influence improvement of agility.

2.6.1 Traditional methods for benchmarking team capabilities

There will always be a need for measuring efficacy and efficiency of projects and teams, especially for software projects with relatively short time-frames and a fierce, competitive market. Tools for doing this kind of measurements have been around long before any official declaration of agile development methods. One of the most popular tools for measuring, and ultimately improving, capability of entities (teams, projects, organizations) is the Capability Maturity Model Integration (CMMI). This method is primarily aimed at traditional software development and argues that agility is hardly visible in the CMMI model (and similar improvement models), since core values of agile methods do not fit the assumptions made about processes and organization in the model. The CMMI-model emphasizes the need for formally defined and controlled processes and gives guidance on how to increase capability maturity by optimizing processes.

2.6.2 Methods for benchmarking team-level agility

As previously presented, there are a lot of core values and principles for the agile way of working that can suffer in large organizations. Primarily values such as self-management, high-bandwidth communication and customer collaboration are in the risk zone for losing their potential in highly regulated environments. Reaching higher potential for an agile software development team is arguably believed to be achieved by applying a set of new practices and processes, such as sprints and user stories. However, achieving true adaptability and flexibility in the way of working cannot be measured by the existence or non-existence of certain processes. This correlates with the earlier mentions of "Scrum-but" and "water-scrum-fall", where practices are implemented but the objectives of those implementations are lost.

Assessing agility in teams is a research topic that is still immature and many, sometimes contradicting, models have been created in order to fill that gap in research. The rest of this section will provide a brief review of some of the most popular models mentioned in the literature and present indicators for agility from these models in order to anchor indicators for agility used in this thesis.

The Objectives, Principles and Strategies framework (OPS) is a tool used to assess how good different agile methodologies are. The framework essentially breaks down the core values of agile development (responding to change etc.) into objectives and principles to support those objectives. Specific practices, such as daily stand-up meetings, are then listed in order to achieve principles and ultimately the objectives of the method. In order to assess to which extent an organization has successfully implemented the practices to support the objectives and principles a set of indicators are proposed.

Sidkys Agile Measurement Index (SAMI) is one of the core components of Sidkys Agile Adoption Framework, which is a direct response and counterpart to the arguable inadequacy of traditional measurement and process improvement tools, such as the CMMI. However, much like CMMI, the tool proposes a level-indicated rate of so called "agile matu-
2.6. Measuring agility

The different levels of maturity are achieved by satisfying core agile principles through applying practices of agile development methods.

The Comparative Agility (CA) assessment tool created by Williams et al \[44\] is a commercial tool used to effectively compare team agility to competitors in the market. To this date, the tool has more than 17 000 users. CA assesses agility in seven different dimensions: teamwork, requirements, planning, technical practices, quality, culture and knowledge creating. The assessment of these dimensions is done by team members answering a Likert scale survey with a set of indicators.

The perceptive agile measurement tool (PAM) \[36\] is developed with a social-psychological perspective. The creators of the tool argue that there is much more to agile development than technical and process issues, hence the need for a tool measuring human perceptions of the vast area of human and social aspects connected with agile development. The tool itself is divided by well-known practices of agile development, setting the dimensions for the assessment, namely: iteration planning, iterative development, continuous integration and testing, co-location, stand-up meetings, customer access, customer acceptance tests and retrospectives. Indicators are listed under these dimensions and Likert scaled questions are then applied to answer them.

2.6.3 Indicators

Seen in table 2.1 are the indicators for human-centric agility used throughout this report, divided in three main areas Self-Organizing, High-bandwidth communication and Customer collaboration. These indicators are distilled from the tools described in previous section.

<table>
<thead>
<tr>
<th>Construct</th>
<th>Unit</th>
<th>Indicator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Organizing</td>
<td>Individual governance</td>
<td>Members choose which tasks to work on</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Members plan their own work</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Authority to make plan-altering decisions</td>
</tr>
<tr>
<td></td>
<td>Team governance</td>
<td>Time estimations is done by the team</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Language used in teams is standardized</td>
</tr>
<tr>
<td></td>
<td>Cross-functional team</td>
<td>Workflow maintains a constant productive level</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Workflow is not affected if a team member is absent</td>
</tr>
<tr>
<td></td>
<td>Members are willing to work outside their specialties</td>
<td></td>
</tr>
<tr>
<td>High-bandwidth communication</td>
<td>Team</td>
<td>Synchronizing work is done through regular informal meetings</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Concerns about reaching goals openly communicated</td>
</tr>
<tr>
<td></td>
<td>External communication</td>
<td>Clear communication channels with business/management</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Changes requested by customer are accommodated for</td>
</tr>
<tr>
<td>Customer collaboration</td>
<td>External communication</td>
<td>Customer easily reachable</td>
</tr>
</tbody>
</table>

| Definition of done is decided by customer |
3 Related Work

This chapter includes a description of related work done within the research area with a similar approach as this thesis. In order to further justify method choices, the applied methods in the studies are compared to the approach of this thesis. A very short summary of the results of the studies can be found at the end of every subsection.

3.1 Data collection

This paper, written by Gwanhoo Lee and Weidong Xia [24] is, much like ours, a response to the lack of empirical foundation in the practices of agile development. A matter which is referred to as a big problem since organizations, according to the authors, implement agile practices without really knowing under which circumstances the methods work or do not work. However, with many principles of agile development preventing traditional ways of measuring project success, this article specifically aims to create empirical evidence for possible changes in software development performance. This is done by finding relationships between core agile constructs and a couple of dimensions for performance, primarily time and budget accuracy.

Although this study is a lot larger than ours, the approach and procedure of the study does not differ a whole lot. Just like the study presented in this thesis, the Lee and Xia study is based on both qualitative and quantitative empirical data. The qualitative data is based on interviews and focus groups, while the quantitative data is based on an online survey sent out to the potential respondents. The study included over fifty 90-minutes interviews and a survey going out to almost 2000 respondents and having an effective response rate of 29%.

Just like our study, the authors of this study used factor analysis in order to analyze their quantitative data. Although both studies used factor analysis for dimension reduction, the choice of factor analysis method differed. Lee and Xia used PLS (Partial least squares) while we used PCA (Principal component analysis).

Unlike our study, the study done by Lee and Xia covers a big part of the software development area. The teams in their study work for very different organizations, ranging in both size and industry sectors. Both private and public sector companies were included in the
study, while our study only looks at teams working for companies in the public sector.

The study found that maximizing agility is not optimal in every software development project. Instead, the teams should focus on cost, time and functional requirements, which in turn will determine how much autonomy and diversity is required for the specific project.

3.2 Decision making

This study, done by Moe et al. [27], was conducted on four project teams. Two of the teams worked at one company and the other two worked for another company. Like our study, this study was meant to "Explore and provide insight into the shared decision-making in agile development". The study is, like ours, of a multiple case study nature. However, differing from ours, this study relied on qualitative data collected through interviews with team members as well as documentation coming from meetings, projects plans and schedules. Although many interviews were conducted, the study relies on so-called triangulation of data sources, meaning that the data was collected through multiple data sources. In our study, we also conducted interviews, red documentation but also decided to add a survey, to further triangulate the results. The survey, that was conducted before the interviews, gave us insights to what subjects were of most interest to discuss during the interviews.

Another difference between our study and this particular study is when the studies were carried out. Our studies were conducted when the projects were already partly completed, while Moe et al. conducted their studies right from the start when agile development was introduced in the projects. This could be seen as an obstacle for our study, as the different projects had worked with agile development for different periods of time. However, we decided to decrease the size of this obstacle by not only ask the questions in regard to how things were at the exact moment of the data collection but also ask about the past and how it had been working historically.

Moe et al. interviewed close to everyone within the different projects in their study, while we decided to only interview three team members out of each team. Although interviewing all team members would be preferable in our study as well, we decided to only interview a set of team members due to timing issues (team members being on vacation during the interview period etc.) and because of the size of the study. Conducting additional interviews and transcribing them would make it impossible for us to stick to our time schedule.

Just like we did with our data, Moe et al. transcribed all their interviews. This was done partly in order to achieve traceability but also in order to code words, expressions and issues of interest into an analyzing application. This was done in order to discover patterns or themes within the data. In our study, we used the same approach to analyze our data. Although most questions were very similar during the interviews, the data analysis yielded different areas of interest.

The study found that the ability to self-management was highly affected by the implementation of shared decision-making. The study also found that unrealistic goals on a team level often, usually makes individual goals more important than team goals. The study also found that the "how" shared decision-making is implemented is of lesser importance. The most important factor of shared decision-making is that all decisions are communicated to the concerned parties.
3.3 Transition from Waterfall to Scrum

To better understand recognized challenges and to identify new ones, Bannink[2] did a case-study which primarily aims to investigate challenges in a Waterfall to Scrum transition from team managers’ perspectives. In addition to correlating literature findings with the findings in the study, Bannink lists recommendations, based on the findings, for anyone about to do a transition from Waterfall to Scrum.

The case-study is, like ours, based on the findings of semi-structured interviews. However, differing from our study is that the study was isolated to only one company with an ongoing transition to agile methods. Another difference from our study is that Bannink primarily compared his studied teams on the basis of already existing research in challenges of the transition to agile, investigating the existence or non-existence of these challenges in the teams, whereas our approach was based on measuring a desired behavior for teams.

Bannink found that the biggest challenges in an ongoing transition from Waterfall to Scrum is for managing parts of an organization to move from a traditional command-and-control type of management to a more facilitating role. A change that is needed in order to empower team members to be more self-managed.

3.4 The impact on communication

Pikkarainen et al [29] aim to investigate agile practices effect on internal and external communication for development teams. In the study, two larger agile projects at a company in Finland were compared on the basis of a rigid conceptual framework covering communication in agile environments. The research is of a case-study nature with interviews as the primary resource for analysis. Unlike our study, the authors also incorporated group interviews with the whole team. During the analysis of the projects, much like our study, both projects were analyzed as isolated entities and then cross-case analysis were carried out to identify differences and similarities.

The researchers found that the use of agile practices both improved and facilitated a more profound and well-functioning communication within the team. When it comes to external communication for the team, the researchers did however find that a lack of adequate communication mechanisms could cause great impediments for communication and they mean that certain agile frameworks (e.g Scrum) cannot offer these mechanisms “out of the box”. The authors further mean that their results suggest that the missing parts of popular agile frameworks could be filled with processes from more plan-driven methods of software development.

3.5 Impact of agile principles and practices on large-scale software development projects

Many of the studies done within the area of agile software development and its legitimacy has a qualitative approach and claimed advantages with agile methods often lack rigid empirical evidence. Lagerberg and Skude [22] perform a quantitative study within two projects, one agile and one with a more plan-driven approach, measuring claimed benefits with the use of agile practices.

The main source for analysis in the work by Lagerberg and Skude was a survey sent out to all project members in both projects measuring employees’ take on different aspects of their daily work. A significant similarity between this study and ours is that the projects are
measured on the basis of a desired behaviour for the teams. However, in our study all teams work according to agile practices which enabled a greater possibility of analysis of external factors. Lagerberg and Skude primarily aimed to measure the effectiveness in the practices themselves and did not, to the same extent, incorporate effects of external factors.

The authors found that practicing agile development could, for example, increase productivity, facilitate knowledge sharing and that it could possibly increase software quality. The authors did also highlight the importance for software documentation in the agile project studied, a practice that otherwise often has low priority in agile practices.

### 3.6 Enterprise landscape

Waardenburg et al. [41] investigate the challenges that occur when agile methods co-exist with more traditional plan-driven development and how organizations deal with these challenges.

Because of the nature of the projects studied, the researchers were not able to participate during the teams’ daily work. Instead, their data was collected during 21 semi-structured interviews with different roles within software development. Most roles were covered such as developers, product owners, release managers, project managers etc.

The researchers found that there are two main challenges that emerged when the projects transitioned to agile. The first challenge was due to the complexity of the IT landscape that many companies have. The landscape had in many cases evolved during a long period increasing the complexity, making it a hard to add or edit functionality. This was found to be an obvious impediment for agile practitioners. The other challenge presented was the lack of continuous business involvement from stakeholders. They found that in order to successfully transition to agile, the mindset of stakeholders had to be changed.

### 3.7 People vs Process

A common perception in reports about agile software development in large organizations is that the organizations get more productivity in their departments that develop software. Conboy et al [8], set out to investigate if a transition to agile always comes together with happy, engaged and more productive employees, even if the transition to agile is not chosen by the team themselves.

Just like our study this study was divided into two separate parts. The first part was conducting focus group discussions with employees from 17 different organizations. The second part included conducting in depth interviews with employees from all 17 organizations.

The research found that not all companies benefit from a transition to agile and that in many cases puts a lot of extra pressure on the development team. The challenges that are presented include reliance on social skills, need for developers to have knowledge about the core business as well as having to be cross functional in their daily work.

### 3.8 Scrum in a Globally Distributed Project

Passivara et al [28] aim to find out how agile practices can be used in large agile software development projects. The report also pinpoints challenges that arose during the use of agile methods in a distributed project (meaning the team is not co-located).
3.8. Scrum in a Globally Distributed Project

The study is a single-case study. During the study, data was collected from semi-structured interviews that were recorded and later transcribed. The seven interviews conducted were with different roles within the project, ranging from the scrum master to developers and product owners. The transcripts were later analyzed and coded into eight main groups that included 77 sub-groups.

The main findings of the study is the importance of scrum education, making sure face to face meetings do occur, frequent communication exists and, at least, partially overlapping working hours.
This chapter will describe how we developed the chain of evidence in order to answer our research questions. Our intention is to provide the reader with a thorough walk-through of our work in order to assure replicability of the study.

4.1 Research Method

The reasoning for having a clear and well defined research plan is that there is a clear view of what is going to be achieved in the study, as well as to avoid confusion during the study [30].

4.1.1 Research Design

The research design is the plan that the researchers will use for getting from the research questions to the conclusions. In Rowley’s [30] words a research design is:

”...the logic that links the data to be collected and the conclusions to be drawn to the initial questions of a study; it ensures coherence.”

Rowley [30] also states the following as the five ground pillars of a research design:

- The study’s questions
- The study’s propositions
- The study’s units of analysis
- The logic linking the data to the propositions
- The criteria for interpreting findings

How we’ve collected the data, including a subdivision of our research questions and a presentation of the units of analysis, and the logic connected to either confirm or refute propositions will be presented in the following sections.
4.1.2 Case Study

To gather the data needed to answer the stated research questions a multiple-case study has been conducted. A multiple-case study is, according to Yin [46], a research method used to:

“investigate a contemporary phenomenon in depth within its real-life context”

This is a definition which is highly descriptive of the nature of our study, where people are the studied unit and the phenomenon, functionality of agile development in large organizations, is highly contemporary.

The case study is a so called observational study or field study approach where an increased knowledge about certain individuals, groups or organizations is the prime objective [31]. The design of a case study also permits vague boundaries between units of study and their contexts, contrary to e.g. controlled experiments [31].

A case study must use multiple sources of data, which can alter from more qualitative data such as interviews and observations to more quantitative data such as surveys. However, this is not what defines a multiple-case study. A multiple-case study is a case study that has been conducted in multiple environments, so called units [30].

Runeson & Höst [31] mean that although case studies have been criticized for being impossible to generalize from and being biased by the researchers, they can still be well suited for some software engineering research:

“This critique can be met by applying proper research methodology practices as well as reconsidering that knowledge is more than statistical significance”.

4.1.3 Rationale for chosen method

Runeson & Höst [31] have the following to say regarding studies on real world issues:

“Conducting research on real world issues implies a trade-off between level of control and degree of realism”

A case-study was chosen as the research method for this thesis because of the very complex nature of a realistic situation. The complexity can hinder the understanding of what and why things are happening, which is a problem for studies with a more explanatory purpose [31] such as the study conducted in this thesis. However, as stated by Runeson and Höst [31], as a study gets more controlled, and easier to understand, the degree of realism decreases.

4.2 Pre-study

Our pre-study process consisted of a thorough literature review. In a quite narrow research area such as this, we saw the need for using a more systematic approach to find literature and used techniques inspired by Kitchenhams [19] guidelines on systematic literature review. These guidelines are primarily aimed for more rigorous research projects and standalone second hand studies. Therefore, a full application of Kitchenhams guidelines has not been done in this project since it would either be too time-consuming for a master thesis or be at risk for quality issues due to stressed execution. However, the guidelines affected our literature review in positive ways. Clear objectives of the literature search were set up beforehand. Second hand studies were only used to find primary sources. The importance of selection criteria is heavily emphasized by Kitchenham [19] and influenced us to be very observant regarding factors as number of citations and year of publication.
In addition, we performed informal interviews with key representatives of the studied units. These interviews were primarily aimed to confirm the relevance of the scope in the study and prior assumptions made by the researchers.

4.3 Research questions

To answer the overall research questions of this thesis the authors of this report decided to split them into smaller more specific categorized questions as suggested by [31]. The lower granularity questions stated below are all extracted from the study’s literature review.

The research design of this study is based on investigating the team members’ perspective and how they feel that they are affected by their surroundings, a large part of the following questions is therefore only dependent on subjects’ own perspective of their reality.

The first research question from section 1.2 in the thesis, namely: “Is Flow a valid framework for analyzing human-centric agility in large-scale software development teams?” is refined as follows:

To what extent are the team-members agile in their way of working?

- How well does the team self-organize?
- To what extent does high bandwidth communication exist in the teams?
- To what extent is customer cooperation incorporated?

Are the concepts of Flow aligned with reality in terms of team’s ability...

- to self-organize?
- to communicate in a high bandwidth manner?
- to accommodate for changes proposed by customer?

The second research question from section 1.2 in the thesis, namely: “Is Flow a valid framework for analyzing individual developers ability to make well justified and/or tactical decisions in large-scale software development teams?” is refined as follows:

How are decisions made within the teams?

- To what extent do external factors affect decision-making in the teams?
- To what extent do team members feel confident in the decisions made by the team?
- To what extent is the decision-making shared in the teams?
- How are tasks prioritized within the team?

Are the concepts of Flow aligned with reality in terms of...

- the impact of external factors?
- how shared decision-making is used?
- the ability to prioritize within the team?
4.4 Data Collection

A big part of the data collected within case-studies is normally qualitative. This data is both broader and richer than quantitative data which can be considered as more precise, but often lacks credibility and validity. This is one of the reasons why data-triangulation is of high importance within a case-study. What triangulation means is that the researchers take multiple angles towards the studied units, and can therefore provide a broader picture of the units. According to Stake [38], four different types of triangulation can be applied:

- Data (source) triangulation—using more than one data source or collecting the same data at different occasions.
- Observer triangulation—using more than one observer in the study.
- Methodological triangulation—combining different types of data collection methods, e.g. qualitative and quantitative methods
- Theory triangulation—using alternative theories or viewpoints.

In this thesis, three different units have been under study in order to strengthen credible and precise results, i.e. data triangulation. Methodological triangulation was applied through an on-line survey sent out to all team members and interviews with project managers and team members. Results from these interviews were also complemented by existing internal documentation from the different projects/teams.

4.4.1 Surveys

In a study of a comparative nature as this one, it is important to get a wide empirical base for data collection in order to be able to investigate differences and commonalities in the studied units. As stated by Lekvall et al [25], surveys are suitable for this purpose.

The surveys were used as a first part of the incremental nature of doing a case study and in order to get a wider knowledge of the context. The questions formulated in the survey were directly connected with the stated study research questions and based on findings of the initial literature review.

Construction of the survey and logical partitioning of the questions were done according to the International Code on Market and Social Research (ICC/Esomar) and recommendations by Lekvall et al [25]. This included confidentiality aspects, clarification of ambiguous questions and separation of questions in different dimensions.

The survey itself was constructed with a set of questions regarding team members’ perception of their daily work within the different projects and the questions answered on a nine-point Likert scale. The questions (around 40) were grouped by subject in to four groups as suggested by Kelley et al [18].
4.4. Data Collection

Surveys were sent to our supervisor for proof-reading and then to the project leaders of the studied projects in order to secure the legitimacy of the questions as well as avoiding any risk for offensive or ambiguous questions, which is also a necessary step according to Runeson & Höst [31].

As the goal was to get a mapping of different perspectives of the daily work in the different teams, the survey was sent out to all members of each team. It was made clear in the introduction that the study’s aim was to investigate their work as they experience it, rather than how they think it should be. The survey, in its full form can be found in appendix A.

Table 4.1 refers to the experience of software development, experience of agile software development, and how long the team members have been active in their current project. The cells right of "Team A", "Team B" and "Team C" refers to the number of team members that answered the specific questions.

<table>
<thead>
<tr>
<th>Team</th>
<th>Experience in software development (years)</th>
<th>Experience in agile software development (years)</th>
<th>Experience in current project</th>
<th>Answer rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0-2</td>
<td>3-5</td>
<td>5+</td>
<td>0-2</td>
</tr>
<tr>
<td>Team A</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Team B</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>Team C</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

4.4.2 Semi-structured interviews

Although surveys are beneficial to investigate relationships between studied units and the questions at hand, they seldom provide any real support for understanding and explaining those relationships [14]. With a predefined set of topics and the aim to explain the relationships found in the survey, semi-structured interviews were considered the most suitable for the purpose of our study.

To assure consistency in this part of the data collection, two interview guides were constructed [31], one for interviews with developers and one for interviews with project managers (Appendix B & C). The guide, in similarity with the survey, starts off with an introduction of why the study is conducted as well as assuring the confidentiality of the interviewees participation. The questions in the interview guide were constructed based on recommendations made by Bryman et al [4]. These recommendations and guidelines were used to ensure that the researchers were not asking leading questions or in any other way steered the interview. This was done in order to avoid the risk of biased answers and ambiguous questions. To avoid language barriers, all interviews were held in Swedish and therefore the guides were also formulated in Swedish.

The main interview itself was divided in three parts for members of the team and two parts for the team managers. For team members, the interview initially covered background information such as a brief description of their role in the team and the project as a whole. The next part of the interview covered the two main themes of the study: team agility and decision-making. Questions asked were directly derived from the stated research questions and were asked in an order based on the funnel model, as suggested by Runeson et al [32], where very open questions were asked first and more specific questions followed. The specific questions that followed were both read directly from our interview guide and as responses to interviewees answers to the more open-ended questions. This was another measure taken to ensure as little influence in interviewees’ answers as possible. The last part of the interview was based on patterns and abnormalities found in survey responses. These
questions were one part of the outcome in the survey data analysis and can be read more about in subsection 4.5.

For team managers, the interview strategy and characteristics were very similar to the interviews with team members. The themes of the interviews, however, differed and now aimed to cover the team’s communication with surrounding organization and the organizations compatibility with agile methods. Questions here were largely based on previous findings and theories found in our literature review.

All the interviews were recorded and later transcribed. The transcriptions were then summarized and the summary, containing the most relevant points made during an interview were sent out for confirmation to each interviewee. As argued by Runeson & Höst [31] this is a largely contributing factor in assuring validity in the study, since falsely made interpretations by the researchers can, to a large degree, be eliminated.

One could argue that probability sampling [4] would be suitable in a study of this nature, since answers from the interview were partially used to explain phenomenons found in the survey (which covers the whole population of each unit under analysis). The reason for not using a probability sampling method is that they all include an element of randomness in the "chosen" objects. Since there were objects in the sampling groups that were not available at the time of the schedule interviews another method had to be chosen. Interview subjects were instead, to a large degree, chosen in accordance with the convenience sample method [4], simply choosing subjects on their availability. The goal when finding subjects was to ensure that they had deep understanding of the way of working in the teams as well as finding subjects who had understanding of the interface between the team and the business.

Table 4.2: Interviewee codes used for referencing interview citations

<table>
<thead>
<tr>
<th>Team</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A1</td>
</tr>
<tr>
<td>B</td>
<td>B1</td>
</tr>
<tr>
<td>C</td>
<td>C1</td>
</tr>
</tbody>
</table>

4.4.3 Internal documentation

The aim and purpose for this study heavily depends on the organizational structure surrounding the teams under study. Internal documentation, such as project plans and maps of infrastructure, have been provided and used for context description but foremost as a means of making valid conclusions connected to previously found challenges in research. This type of data-collection is also described by [46] as a source of evidence in a case study.

4.5 Data analysis

The data analysis that has been conducted during this study can be divided into two separate parts. One analysis was conducted on the data collected from surveys and one on the data collected through interviews.

4.5.1 Survey analysis

When all responses from the survey had been collected an analysis of the data was made. One of the intended purposes of this analysis was to find key areas of interest to discuss during the interviews. By doing so, the researchers could avoid asking about less important
4.5. Data analysis

topics during the fairly short interviews. It was also possible to ask questions about subjects that differed between the subject groups, and therefore get a better understanding of how these factors affected the teams. Also, this first analysis widened our eyes by pointing out relations that were not obvious at first glance.

Descriptive statistics

One part of the descriptive statistics that was collected in the survey included for how long the subjects had worked with software development in general, for how long they had worked with agile development and for how long they had been working in their current project. This data was mainly collected in order to verify that the three subject groups were comparable. Also, when going through other parts of the descriptive data, the researchers were looking for patterns and abnormalities in the data such as:

- Distinct differences in mean values between teams
- Dispersion of answer within teams (i.e. common picture)
- Variance between teams

Some basic bar charts were also created in order to see if there were any quickly visible differences between the different subject groups. This was used to compare the different groups to each other and to find data that stood out from the other teams. Also, all questions were looked at per group in order to discover variance in between members of each subject group.

Multi-variable analysis

Factor statistics is a method used to describe the variance between different observed variables. It is used in many sciences to deal with data sets where there are a large number of variables observed and to highlight underlying, so called latent, factors that affect the variables. In this thesis, a multiple factor analysis has been used since the survey had multiple continuous variables. The specific type of multiple factoring chosen was principal component analysis.

Principal Component Analysis

Principal component analysis (PCA) was used during the data analysis in order to transform variables in to a smaller number of variables called principal components. It can also be used to spot trends, patterns and outliers that the data holds in an easier way than by looking at the raw data collected in the study. Abdi et al. argue that there are four goals for using PCA:

1. Extract the most important information from the data table
2. Compress the size of the data set by keeping only this important information
3. Simplify the description of the data set
4. Analyze the structure of the observations and the variables

PCA reaches these goals by computing new principal components which are linear combinations of the variables. The first principal component holds the largest possible variance, the second must explain the second largest variance and so on. The second component is computed the same way as the first but must be orthogonal to the first one. This is then repeated (the third must be orthogonal to the first and second etc.) depending on how many
4.5. Data analysis

components that are found during the analysis [1].

In this thesis PCA was used prior to interviews to understand latent factors causing variances in each individual team. Underlying factors were modeled for each category of the survey (decisions, organizing work etc.).

4.5.2 Interview analysis

The raw data from the interviews was recorded on the authors phones, which were also used as recorders throughout all interviews. In order to organize the data and to structure it in a more comprehensive way, a processing of the data was done according to the recommendations made by Runesson et al [32]. The transcriptions were processed in a software called NVivo[1] where all raw text was coded according to the codes presented in table 4.3, based on the theoretical reference, the purpose of the study and the stated research questions. Having coded the interviews, finding themes in interviews was possible in a much more systematic way.

<table>
<thead>
<tr>
<th>General code</th>
<th>Specific code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agility</td>
<td>Customer collaboration</td>
</tr>
<tr>
<td></td>
<td>High-bandwidth communication</td>
</tr>
<tr>
<td></td>
<td>Self-organizing</td>
</tr>
<tr>
<td>Decision-making</td>
<td>Shared decision-making</td>
</tr>
<tr>
<td></td>
<td>Prioritization</td>
</tr>
<tr>
<td></td>
<td>Conflicts and uncertainties about directions</td>
</tr>
<tr>
<td>Project</td>
<td>Agile acceptance &amp; compatibility</td>
</tr>
<tr>
<td></td>
<td>Organization</td>
</tr>
<tr>
<td>Vision &amp; Goals</td>
<td>Definition</td>
</tr>
<tr>
<td></td>
<td>Importance</td>
</tr>
<tr>
<td></td>
<td>Usage</td>
</tr>
</tbody>
</table>

Table 4.3: Interview processing codes

This chapter will present the results collected during the case study. The chapter will be divided into three parts, one for each of the studied teams. Each of those parts will thereafter be divided into three parts presenting the context of the teams, the results collected from surveys and the results collected through interviews with team members. The two later parts mainly focus on the teams perceived ability to work in an agile manner and their perceived ability to make decisions. Also, within each team’s section there will be a brief description of both the survey respondents and the interview subjects.

5.1 Team A: Söderenergi

The project studied at Söderenergi has evolved out of a process mapping (i.e. mapping the way of working in order to find impediments) that started 2012. The mapping was meant to find processes within the company that could either be more effective, and be improved, or processes that were considered to be redundant and could therefore be removed. The mapping, as well as the new project were both ordered by the management of Söderenergi. The project mainly aimed at improving the IT structure within the organization and to automate as much as possible. By implementing an integration platform, linking all the systems used at Söderenergi the team hopes to reach this particular goal as this will decrease the amount of manual labor required from the workforce. The work required today could be manually finding, handling and adding information (such as equipment stocks and outgoing orders) into several separate systems in order to keep the information up to date in all systems. A major issue within the handling of information was that the different departments within the company did not share processes around this area.

Although saving money is one of the main goals with the project, it is not the only one. Another important goal for the project is to concretize how processes should be carried out. The reason for this is to make it easier for new employees to learn their new role. [A2]

This project is the first agile software development project that has been carried out at the company. Although the project is moving forward, it has been a big change for the whole organization to work together with an agile software development team. Developing software
with an agile way of working within a non-agile organization was pointed out as a huge challenge by one interviewee:

“One of the biggest problems with agile software development is to work out the issues that arise when working with a non-agile organization. If one could find a way that wouldn’t force the organization to work in a different way, but rather find a link between agile software development and a plan-driven organization, that would be superb” [A1]

5.1.1 Organizational structure

Almost the whole team working with the project at Söderenergi has been hired as external consultants. Some of these consultants have been hired directly based on their competence in systems that are currently used in the intranet or will be used in the new version. Therefore, almost all team members have specific roles in the team, ranging from integration developer and test-specialist to specific application specialists. Some of the team members are, since before the project, hired by Söderenergi and are currently members in the team in part-time roles.

The organization itself is very flat and the project is managed directly under the CEO. The project is fully undertaken by the team, meaning that no other actors in the business are involved in the project organization.

![Figure 5.1: Project organization team A](image)

5.1.2 Vertical scaling of agile principles

Before this project, Söderenergi have not worked with an agile software development team. This has been a challenge, both for the project team and team management as well as for the rest of the organization. One of the early challenges that occurred was linked to a period of time when the team had nothing to show the organization. The organization expected to see progress, made early on in the project and the team being able to demonstrate improvements straight away. When the team was not able to do these demonstrations the involvement of the organization decreased. As a consequence, the team worked without getting proper feedback from the organization, and developed part of a product that did not meet the organization’s requirements.

Parts of the organization have also had issues with not getting a long-term plan from the development team. While the team plans three to six weeks at a time, some units in
the organization want a plan for delivery a couple of months in advance. The reasoning for wanting these plans is to secure resources for the launch and testing phase of the product. Because of their agile work style the software development team at Söderenergi do not want to plan that far ahead since it would impede their agility.

One of the interviewees was doubting if the organization was ready and mature enough to work with an agile software development team. It was said that:

"It takes blood, sweat and tears, also from the organization. Maybe they did not understand how much the transformation to agile software development was going to cost in time and man hours in the end"[A3]

5.1.3 Development process

The software development at Söderenergi started after the mapping of processes that was conducted at the company. During the mapping 14 projects were planned. These 14 projects have since then decreased to 9 projects, all covering processes in different business units. The set of these 9 projects have later started to be called the VI-program. In the mapping, some overall requirements were stated. These requirements have continuously been specified by the team. The team at Söderenergi has been focused on ensuring that all requirements are well rooted in the company’s business interests. Although this has been a primary focus, lacking knowledge and insights from the business of how a potential system could work have, from time to time, forced the development team to look at external systems for inspiration.

The requirements are prioritized by the team during the sprint planning meetings that occur every three weeks. During these meetings, the upcoming sprint is planned and the tasks that are being implemented during the sprint are handed out to the developers.

When the task reaches the team members’ desk the purpose of the task is clearly specified. However, the actual design of the solution is left open for the developers to specify.

When the team finishes a product, a workshop demonstration of the product is usually carried out by the team. The team invites stakeholders and end users to see the new product in order to get additional feedback. This additional feedback is important for the team in order to make final changes before the launch of the product.

5.1.4 Project management model

The project management model used at Söderenergi is called PPS (Praktisk ProjektStyrning). PPS was developed by the Swedish company Tieto and is used widely in Sweden. The model is not based on theory, but rather built upon experience of real life situations. The core of the model is 8 decision points (DP). These decisions do not have to be made sequentially, but most are. The decision points are:

- DP1: Decision about starting the preparations for the project
- DP2: Decision to continue or cancel the preparations
- DP3: Preparations ends. Agree upon project definition, requirement description and solution description
- DP4: Decision about starting the execution of the project
- DP5: The execution continues, changes or is canceled depending on risk analysis and verification of partial goals
5.1. Team A: Söderenergi

- DP6: Decision to launch the project to the end users
- DP7: Decision to make a hand over of the administration of the project
- DP8: Decision to consider the project to be done

This model is used within each project of the bigger program. However, in order to make an agile software development approach viable within the program, the projects are worked at in parallel so iterative integration of each project is possible. This enables the team to work agile although the project is managed in a more plan-driven manner.

5.1.5 Clarity of goals and vision

The interviewees at Söderenergi all agreed that the vision of the project is to eliminate "double work". The term was used to describe the additional work that had to be done in order to make sure information was available in different systems used at the company. One of the interviewees described this vision along with, what he described as a project slogan:

"It should be easy to do things the right way" [A1]

It was mentioned in the interviews that having a vision and goals that everyone in the team could relate to their own role is important. Because of the role based team setup, this was seen as a tool to unify the team and to give the team members an overall goal to reach for, which they could all relate their individual roles to. One of the interviewees related back to previous experience in software development and stated that:

"You sat there and were a small cog, not knowing why you did things. In the long run, I think that kills creativity and involvement. Involving people and making it clear why things are done, I think, increases the work performance and the ambition of the employee" [A1]

One of the interviewees said that the project manager worked quite a lot with ensuring that everyone in the team understood the goal, usefulness and purpose of their own work, instead of fixating on their current tasks [A1]. According to two of the interviewees at Söderenergi, the importance of clear vision and goals is based on the individuals in the team [A1][A3]. One of the interviewees also mentioned that he thought that some team members could be less controlled and still be efficient, while others needed a certain level of control and structure in their work to function well. [A1]

5.1.6 Description of survey respondents and interview subjects in team A

The survey went out to 10 possible respondents. Out of the 10 respondents 9 answered the survey which gives an answer rate of 90%. The respondents had various roles within the team such as application specialists, application admins, integration developers and integration architect. The subject for interviews also had different roles. One interview was held with the product owner, who had the title of business analyst within the organization. The other two interviewees had the title integration developer and application admin/development manager. Presented in the graphs below is some further information about the respondents’ experience within software development.
5.2 Team B: Trafikverket

Currently at Trafikverket, a portal called PPI (project portal investment) is used to handle and store the documentation regarding their investments. The platform currently used is called PPI5 and is built on SharePoint 2010. The platform has since the launch been custom fitted to suit all users around the company. All this customization has led to quite a code debt, which affects the possibility to make even more customizations. It has come to a point where the project administration is scared of making more changes, since they are not sure what effects the new changes in the code would have on the existing portal. This is one, and probably the biggest reason for Trafikverket to develop a new portal, PPI6. PPI6 is built on SharePoint 2013, which is a newer version of the existing platform.

After a slow start, mainly due to requirement issues within the project, the project management have had to make some changes in priorities. Instead of trying to accommodate the organizations every request of new functions in the new portal the team instead have to focus on meeting the deadline of the project with only the minimum requirement of new functions delivered. Since early 2016 the development team has a new part project leader, who has made some changes to the way the team is supposed to work.

5.2.1 Organizational structure

The project is currently divided into three sub-projects: requirements, development and migration (not shown in figure below as this sub-project is yet to start). The requirement team is responsible that the requirements gathered meet the organization’s needs. To their help with this task they have nine so called “work groups”. Each of the nine groups is responsible for setting up requirements regarding one part of the portal. These requirements are later “translated” by the requirement group into requirements which are understandable for the developers. Not until then are the requirements forwarded to the developing team. [B2]

All three sub-project teams are working under the same head and organizational project leaders. Under the two project leaders there are three part-project leaders, one for each project part. In the requirement team, there is a wide range of backgrounds of the members. One part of the members come from an IT requirements background and one part is end users, using the portal in their daily work. A third part is the stakeholders that ordered the new portal. These three groups (requirements, end users and stakeholders) collaborate within so called “work groups” in order to give the development team a clear view on what has to be done in the project. The requirement team are the ones taking on the customer role within the project, and are the team that the development team have the most interactions.
with. However, since the requirement team is quite large, consisting of nine work groups all containing at least five people there is no declared customer working as a product owner. This was pointed out as a problem by the interviewees as they found it very challenging to find the right person to put their specific questions to. [B2]

The development team is responsible for developing the portal to meet the requirements sent from the requirement group. This is the team that has been studied during this thesis. The team uses an agile way of working with stand-up meetings three times a week and product demos after each three-week sprint [B1]. The team had a slow start in the project, partly because of them not getting requirements ready to work with. This has led to parts of the organization doubting if the team will be able to deliver on time. [B2]

The third and final part of the project is the migration of all the documents stored in PPI5. This is expected to be done with minimal interference with the daily work at Trafikverket. This sub-project had not, at the time when this thesis was written, been started (and is therefore not represented in the figure below).

![Project organization team B](image)

**Figure 5.4: Project organization team B**

### 5.2.2 Vertical scaling of agile principles

As mentioned earlier Trafikverket is not an organization that is particularly used to working with agile development. One interviewee pointed out that the organization believed that agile development only impacted the development team and not its surroundings. It was also believed that the organization did not want to be part of the development process (as a customer usually is in agile development) and not willing to spend the resources to be so. [B1]

One interviewee pointed out that there were a lot of "us and them" thinking in the organization. The development team thought that the organization were poor at creating requirements while the business side did not have an understanding for how the developing team worked or how resources were allocated within the team [B2]. All the interviewees pointed out that the collaboration with the business had gotten better after the work groups were implemented. However, it was also pointed out that requirements coming from separate groups sometimes collided, making it hard for the development team to know which requirement they should proceed with [B3]. As mentioned earlier, the organization got a better understanding of the agile way of working as time went by. This lowered the pressure on
the development team, but one interviewee pointed out that instead of worrying about the
development team, the organization switched focus and started worrying about the testers
and how they were working instead [B2].

5.2.3 Development process
The software development team in PPI6 does not take part in collection of requirements. The
requirements are, as previously described, delivered by the requirements team. The work-
groups were created in the beginning of 2016, and have a diverse experience in creating
requirements. The different experience can be seen in the level of details the requirements
include. Some of the requirements are very abstract, while some are as detailed as a mock-up.
Before these work-groups were created PPI6 mostly worked with recreating PPI5 but on the
newer platform that has been implemented.

An issue that has arisen after the work-groups were created is the constant contact be-
tween the work-groups and the development team. The work-groups are eager to see their
specific requirements prioritized over others and therefore irregularly contact the developers
to ensure this.

Too ensure compatibility with the rest of the platform and to increase the possibility to
re-use code the IT department at Trafikverket have set up some rules for the developers to
follow when writing code. However, the developers themselves do not see this as an issue.

5.2.4 Project management model
Trafikverket is a Swedish government owned authority that historically has been a plan-
driven organization. The project management model used within the organization is still
plan-driven, although a refined version called XLPM is the one currently used. XLPM uses
six tollgate decisions in order to guarantee economic governance [B2]. The six tollgate deci-
sions that have to be made is:

1. Decision to start project analysis
2. Decision to start project planning
3. Decision to start establishing the project and the execution of it
4. Decision to continue the execution, either following the existing plan or with a new
   refined plan
5. Decision to handover the final product to the customer
6. Decision to start project shut down

These decisions are all taken sequentially, one after another. For example, the project
planning will not start before the project analysis is finished. This is quite the opposite of
how agile development is expected to work, and combining the project management model
with agile development has shown to be a challenge at Trafikverket.

5.2.5 Clarity of goals and vision
Team B does not use a clear vision as a management tool. However, since the new part project
leader was included in the team, the team to a larger extent has started using the project’s
goals in their daily work.
Although all interviewees mentioned that having a clear vision for the project is of great importance, none of them felt that vision had a big impact within team B [B1][B2][B3]. One interviewee even said that he thought that the team didn’t have a vision [B1]. They all agreed that having a vision would possibly help make the team strive against a common goal, which in turn could increase the team spirit.

One interviewee felt that the project goals were so abstract that they were hard to use for anything. The same interviewee also stated that the goals are formulated in a way that makes them too easy to fulfill. The interviewee pointed out that this was a good thing, as they would always be able to argue that the project had been successful, but the interviewee also thought that the lack of clear goals was damaging for the actual outcome of the project. Without stricter, more measurable goals the interviewee thought that the team’s effort decreased[B2].

Another interviewee mentioned that the vision of the project started off on a good level, but budget cuts coming in had the effect that the vision plummeted. The interviewee said that the vision went from:

“'creating a superb portal, with good user experience, which everyone is happy with’ to ‘let’s get something together so we can shut down the old portal’”[A3]

It was also mentioned in an interview that the lack of vision within the project was potentially making the product less user friendly and more technically advanced [B1].

It was pointed out in the interviews that the project was lacking a way of thinking on how to receive a requirement from the requirements team and turn that requirement in to a usable function. A lack of structure within the project was also pointed out, making it hard for the developing team to know which tasks to prioritize, and what functionality that was really required in order to deliver an acceptable product.

### 5.2.6 Description of survey respondents and interview subjects in team B

The survey conducted in team B went out to 8 possible respondents. 8 out of the possible 8 respondents submitted their answers, which gives a response rate of 100%. The respondents were mostly developers within the team, but also one software architect, one software tester and one software requirement person answered the survey. The interviews were held with one software developer working purely with development, one software architect/developer and the project manager of the team. Presented in the graphs below is some further information about the respondents’ experience within software development.

![Figure 5.5: Years of software development experience](image)

![Figure 5.6: Years of agile software development experience](image)
5.3 Team C: CGI

The current software infrastructure for construction and practicing of driver license theory tests has turned obsolete and combined with a lot of continuous modifications to the system, the Swedish Transport Administration (Trafikverket) understood that the system would not withstand further development and therefore had to be replaced with a new system. Initially Trafikverket were out to find an existing, standardized system, which could be object for minor customization before being deployed. However, no system fitting the existing needs were found. Trafikverket therefore decided to hire a software developing team from the consulting firm CGI, located in Östersund. The project was decided to be co-sourced, meaning a very close collaboration between Trafikverket and CGI rather than a typical customer/vendor relationship [C1]. The application is developed from scratch and is written in .NET.

The project undertaken by CGI has the overall goal to develop a highly adaptable, ready for the future, system which handles its purpose at least as well as the existing system. An existing system that is currently divided in three subsystems:

- **ConstructIT** - construction of new questions for tests.
- **ProvIT** - construction of actual driver license theory tests.
- **TranslateIT** - the subsystem which is used for translating questions in tests, making tests adaptable for citizens who do not have Swedish as their mother tongue.

All these subsystems are, in the present system, administrated by Transportstyrelsen, a Swedish government agency which regulates all transportation in Sweden and therefore the holders of criteria needed for passing theory tests for the different driver’s licenses issued in Sweden. The actual execution of the tests, where students either pass or fail, are administrated by Trafikverket. Along with the construction of the new system, all administration of tests (including questions, test construction and translation) will be administrated by Trafikverket and Transportstyrelsen will solely administrate the backbone regulations.

5.3.1 Organizational structure

Co-sourcing, as opposed to outsourcing, basically means a team is hired to carry through a project in a much more collaborative manner. Team members become integrated in the project organization and act accordingly. This structure is used to open up for a much more reliable communication when it comes to continuous feedback and channeling requirements.

In this case CGI is hired by Trafikverket according to the above described conditions. In order to effectively maintain such a setup, a managing team has been formed at Trafikverket. Besides regular responsibilities of managing a project, including budget and time planning, the managing team includes key roles associated with agile development such as a product owner, acceptance test leader and reference groups.

The development team, in which all members are located in Östersund, have a single-unit setup. This means that all members involved in the development of the system (developers, test engineers, architects etc.) are working as one unit, sharing office-space and work environments [C1]. The method of development in the CGI team is heavily based on Scrum practices with influences from many other agile models which combined represent a tailor-made solution for agile software development at CGI.
5.3.2 Vertical scaling of agile principles

To ensure that the co-sourcing relationship with CGI is functioning well from Trafikverket’s side, the managing team has also decided to apply agile practices in their work. By using a synced iterative way of working with CGI, the teams are enabled to plan, overview and follow-up on the same platforms. This effectively means that the managing team works with packages which primarily contain user-stories from existing systems and in some cases, very general requirements. These packages are scheduled to a rigid time-plan according to XLPM. However, as the actual content (specific requirements and tasks) are produced continuously in collaboration with the development team, this makes the two teams able to share backlogs and regular feedback-meetings which sync with sprints.

However, one of the interviewees explained that he does not believe that the management’s compliance to agile principles plays any crucial role in a healthy relationship between customer and vendor [C1]. The health of that relationship, as pointed out by the same interviewee, is rather based on a trust in the way of working, a trust in the development team and a compliance to the need of presence due to the feedback nature of agile development. Factors that all interviewees of the CGI team feel exist in this project. One of the developers also explained that, besides understanding the utility of agile development, the management team understands possible obstacles that can erupt during the process and how these are mitigated [C3].

5.3.3 Development process

The project was initiated with a set of project goals and impact goals, where the project goals refer to actual outcomes of the project and impact goals refer to measurable, business beneficial, effects of that outcome [C1]. With these goals as a backbone for the project, CGI made a mapping of the current system along with many suggestions for improvement in
5.3. Team C: CGI

order to create user-stories for the new system. When the mapping of the system was done, these user-stories were divided in packages that were to be developed in a chronological order.

Deciding the contents of each package, which is specified as general requirements, is a job done continuously in a collaboration between stakeholders, the management team and the CGI team. Planning the development order of the contents in each package is usually made three sprints ahead and requirements specified by the CGI team are demonstrated to stakeholders every two weeks using mock-ups. The specification of actual, detailed, tasks are then made on a day to day basis by the developers. The developers are therefore entitled to define their own work, as long as it does not diverge from overhead requirements. The feedback on developed solutions is then continuously administered by the product owner and key representatives from Trafikverket (such as requirement and acceptance test specialists) as they move from initial requirement to tested deliverables.

One interviewee explained that this has not been the case all along and that the project, in the beginning, had a problem where requirements specified, as the outcome from the regular two-week demonstration, were too detailed. This left little room for opinions and suggestions from developers. This problem has been mitigated as the project has reached further into the implementation stage [C2].

5.3.4 Project management model

As previously mentioned, project management within Trafikverket is heavily based on plan driven methods. This case makes no exception and management of this project also follows the XLPM model. The model is however slightly adapted for this project in order to strengthen compatibility with the agile development method of the team. What primarily differentiates the usual execution of the XLPM toll-gate model to the adapted one used in this case, is the initial level of detail of the requirements. The definition of the requirements is formulated in a very general away and is, in many cases, only defined as user-stories. The requirements are then continuously reworked, gaining more and more detail, as the more general requirements and project goals move closer in time [C1]. The other big difference is the fact that both Trafikverket and CGI is in a mutual, well defined, agreement regarding communication, transparency and feedback, which also could be considered unusual in a project management model of the plan driven nature.

5.3.5 Clarity of goals and vision

The definition of the project vision, according to all interviewees in the CGI team, is to build a working future-proof system but more specifically to eliminate any possibility of cheating in order to pass tests. Even though a clear vision for the project was shared among the interviewees, the applicability or possibility to utilize that vision in their daily work was not as clear. One of the developers explains that the vision does not affect his daily work since his tasks are on such a detailed level and that active work against vision only occurs on a higher level of abstraction in the project. This view is also shared by the team manager. He elaborated even further on the importance of the team’s vision and means that the shared vision is very important for the team to make sure they move in the same direction and to trigger engagement in the project.

5.3.6 Description of survey respondents and interview subjects in team C

The survey went out to 8 possible respondents at CGI. Out of the 8 possible respondents 6 answered the survey giving a response rate of 75%. The survey was sent out mainly to
software developers but it was also sent to a requirements collector, an architect and two software testers. The interviews held at CGI was with the project manager of the team and with two of the software developers, one with more expertise in front end development and one with expertise within back end development. Presented in the graphs below is some further information about the respondents’ experience within software development.

![Figure 5.8: Years of software development experience](image1)

![Figure 5.9: Years of agile software development experience](image2)
5.4 Result from survey

In order to better illustrate and comprehend the answers from the survey, questions have been clustered into themes. An explanation for the clusters is presented alongside each illustration. The illustrations show combined averages for all questions included in each cluster. As mentioned in the method chapter, all questions of the survey were answered on a 1-9 Likert-scale. The full list of questions included in each cluster can be found in Appendix D.

To better comprehend the spread of the results in the survey, standard deviation will be denoted as high spread if standard deviation of the answers is above 1.5 and low spread if it is lower than 0.8.

5.4.1 Perceived agility in teams

The following sections will present the perceived agility in the teams regarding the ability to self-organize, communicate in a high-bandwidth manner and collaborate with the customer.

Figure 5.10: Perceived ability for self-organizing in teams

- **Maintaining productivity** - Represents team members’ view on how well members of team can keep a constant productive level, without being overworked.
- **Team governance** - This cluster represents team members’ view on how independently the team can estimate and carry out their work.
• **Individual governance** - Represents team members’ ability to freely choose which tasks to work on.

• **Cross-functionality** - Represents team members’ view on how comfortable team members are with working outside their specialties and how well the team handles members becoming unavailable.

**Self-organizing in team A**

The first thing that stands out in the survey done by team A regarding self-organizing is the perceived ability for the team members to carry out their work without interference from management. All survey respondents were unanimous, with a low spread in the results and answered "agree" or "totally agree" when asked about this matter. All team members also feel that they have a common language within the team to describe the project. However, the statement regarding the ability to choose what tasks to work on had high spread in team A and the results was ranging from "disagree" to "totally agree" with a mean close to "neither agree nor disagree".

**Self-organizing in team B**

From the survey conducted in team B regarding self-organizing, the first, most noticeable result is that a majority of the members disagrees that the team is able to maintain constant productivity. All members also express a disability to meet their time estimations. This is, hardly surprising, also combined with a feeling of a lack of resources within the team. The members feel that a common language regarding the project is spoken within the team and that they have a sound relationship to cross-functionality within the team.

**Self-organizing in team C**

Team C’s perceived ability to self-organize is high and all members answered, with low spread, that they are very comfortable in the team’s ability to keep a consistent productive level. The members are also very confident that the team has the right resources to achieve the project goals. As seen in figure 5.10, cross-functionality has a bit lower mean compared to the other three areas. This mostly depends on the fact that some members felt that work cannot continue as usual if a team member is unavailable.
5.4. Result from survey

![Result from survey diagram](image)

Figure 5.11: Perceived ability for High-bandwidth communication in team

- **Ability to raise concerns about reaching goals** - This cluster represents team members’ perception on how well concerns about reaching goals can be openly addressed.

- **Effectiveness in regular meetings** - Represents team members’ view on how effective daily meetings are in synchronizing the team’s work.

- **Management communication** - This cluster represents the team members’ view on how well their communication works with the managerial parts of the project organization.

- **Accommodating change of plans** - Represents the team members’ perception on the team’s flexibility when it comes to change of plans.

**High-bandwidth communication A**

As can be seen in figure 5.11, the ability for informal communication is generally perceived as high amongst the members of the Söderenergi team. The carrying out of daily stand up meetings was especially seen as a very effective process by all members (low spread) of the team.
The team is also very comfortable in openly voicing concerns about reaching goals within the project. One or two team members feels less confident in the ability to communicate with management of the team.

**High-bandwidth communication B**

Very few members in team B felt disadvantages in the team’s ability to communicate in a high-bandwidth manner. One can nevertheless see a pattern of conservative answers from some of the members on questions in all areas, were many answers vary between "neither agree nor disagree" and "agree".

**High-bandwidth communication C**

All members in Team C responded that they have a very well working communication within the team. The only parameter that actually stood out was the statement regarding daily meetings and their effectiveness in synchronizing work. This was the only parameter with a slight spread in the answers and one member even disagreed that the activity is effective for synchronization of work.

![Figure 5.12: Perceived ability for Customer collaboration in team](image)

- **Accommodating feedback** - Represents team members’ perception of the team’s ability to accommodate for feedback given by the customer.

- **Clear communication** - This cluster represent the team members’ view on how well their communication works with the customer.
5.4. Result from survey

- **Definition of done directed by customer** - Represents team members’ view on respective customers’ ability to define the state of done for requirements.

**Customer collaboration A**

As seen in figure 5.12, the team is comfortable in their ability to accommodate for changes requested by customers with minimal bureaucracy, an opinion that had very low spread in the results. In contrast to this, the statement regarding if requirements’ definition of done was set by the customer was answered as “neither agree nor disagree” by a majority of the members. When it comes to communication with the customer, some members felt the communication channels were less clear. The majority of the members did express that the customer was easily reachable.

**Customer collaboration B**

The results regarding customer collaboration in team B were generally low. When asked about requirements criteria to be considered done, the survey showed a great uncertainty among the members. However, the majority disagreed, or even strongly disagreed, that the requirements definition of done were directed by the customer. Many of the team members also felt that there was a lack of clear communication line to the customer; a sign of lacking a common interface for communication. A majority of the members also felt that the customer was easily reachable in the project through other means of communication.

**Customer collaboration C**

All team members’ opinions regarding customer collaboration were very positive, in fact all areas besides “definition of done directed by customer” received answers between “agree” and “totally agree” with low spread. Only one of the team members disagreed that requirements definition of done is directed by the customer, whilst the rest agreed that they are.

5.4.2 Decision-making in teams

The results presented under this section were gathered through the survey in the part regarding decision-making. The result will be divided into three topics with subsections.
5.4. Result from survey

Prioritization A

The figure 5.13 presents how the team perceived their ability to prioritize within the project. As could be seen in the figure 5.13 the team feel that they are good at prioritizing, and what has been prioritized usually gets followed through. Both these parameters were answered with low spread as nobody disagreed with the statements linked to these parameters. Also, the team feels that they usually do not have to re-do tasks that have been considered done. Although the average at this parameter is high, the answers had a high spread. While some team members strongly agreed that tasks often had to be redone, a couple of team members actually disagreed with the statement.

Prioritization B

The team at Söderenergi felt that they had a good understanding of which tasks were most important and that they were good at following the plans that the team set up. This led to the
team feeling they rarely had to redo the tasks that had been finalized. However, although the average was quite low on this parameter, there were two team members that felt the opposite, and felt that tasks often had to be redone.

**Prioritization C**

With a low spread, all members of the team agree or strongly agree that the team follow the plans that have been set up. The team also feel like they have a good understanding of what tasks should be prioritized in order to achieve their goals. No one in the team feel that tasks often have to be redone.

**Figure 5.14: Perceived uncertainties and conflicts in all teams**

- **Follow through on plans** - Represents the team members’ thoughts on how well the team can follow through on plans without having to accommodate external parties’ wishes.

- **Completions of tasks** - Represents the team members’ perceived ability to complete tasks they have started.

- **External noise filter** - This cluster represents the team members’ view on how well the team handles external noise.
• **Manageable rotation of tasks** - Represents the team members’ ability to fully commit to tasks before they change.

**Conflicts and uncertainties about directions A**

The team does not feel they are exposed to much external noise coming from the rest of the organization, and feel they can work quite independently without having to change tasks in order to please the organization’s needs. They therefore also feel that they can follow through on the plans set during planning meetings and complete the tasks they have set out to complete. The noise that the team felt they were exposed to seems to be directed only to a selected few within the team, as most of the team strongly disagreed with the statements regarding noise, while a few others strongly agreed.

**Conflicts and uncertainties about directions B**

Although the team unanimously felt they were exposed to a lot of noise from the surrounding organization, they also felt that the tasks were rarely changed. They all, with a low spread in the results, agreed that the tasks that had been started would be completed soon thereafter and that the team was able to follow through on the plans.

**Conflicts and uncertainties about directions C**

As can be seen in the figure above[5.14] the team does not experience a lot of noise from the surrounding organization. Nor do they feel that tasks often change, which can also be seen in the high average of being able to follow through on their plans. Although the team has a high rating on completion of tasks, one person disagreed with the rest of the team and answered disagree to the statement that team members usually complete the tasks they have started.
5.4. Result from survey

Figure 5.15: Perceived ability for shared decision-making in teams

- **Authority evenly distributed** - This cluster represents the team members’ thoughts on how the decision making authority is spread within the team.

- **Manageable complexity of subject in discussions** - Represents team members’ ability to understand and comprehend the team’s discussions in different subjects.

- **Trust in team** - Represents team members’ trust in team consensus.

- **Efficient decision-making** - This cluster represents team members’ view on the teams ability to smoothly reach consensus.

- **Well justified decisions** - This cluster represents how confident the team members feel that the decisions made are justified and how empowered they feel to make decisions.

**Shared decision-making A**

In regards to shared decision-making within the team, most team members felt that the level of the discussion was usually held at a good level, ensuring everyone could participate in the discussions. However, with a low spread, the team agreed that depending on the subject of the discussion, some team members had more authority in the discussions. The team members seem to trust their colleagues, as can be seen in the two parameters "well justified decisions" and "trust in team". Both of these parameters received high results with low spread. Also, making decisions in group was something that the team valued.
5.5 Result from interviews: Perceived agility in teams

Shared decision-making B

Although the team in general felt that the discussions were at a good level, two persons within the team thought that discussions were often too complex, in order for them to give their input. The team also felt that they made well justified decisions since the team member with the deepest knowledge about the subject was usually the one that made the decision.

Shared decision-making C

In general, the team did not think there were too many opinions when a decision had to be made. However, one person agreed with the statement which made the mean a bit higher. The team at CGI also feels they have discussions on such a level that everyone in the team can contribute. They also have a lot of confidence in the team’s ability to make decisions which can be seen in the parameters “well justified decisions” as well as “trust in team” which both scored high.

5.5 Result from interviews: Perceived agility in teams

The following sections will cover interview results regarding the individual team members’ perspective on their team’s relation to agile principles.

5.5.1 General view of agility in team A

In the Söderenergi team, a majority of the members are hired consultants. Some of these have previous experience with agile development. The part of the team that are not hired consultants had none, or very limited, experience from agile development when the project took off. In order to standardize the way of working, the project was initiated with a course in the way of working for the team. One of the team members describes the general state of agility in the team:

“The team relates very well to agile principles. Communication with stakeholders initially worked poorly, which we were forced to adapt to given the nature of the project.” [A3]

The “nature of the project” refers to the fact that end users of the system have problems relating to and realizing potential benefits in changes of the existing system. All interviewees recognized this as a challenge that the team struggles with. It was also mentioned that this is a challenge that has had implications for the team’s ability to work in an agile way since customer feedback is, in many ways, what fuels the agile way of working. One of the interviewees made the following analogy:

“Suppose our customers use hammer and nails today. Even though we give them a demo of a fully functional nail gun, almost everyone would have a hard time understanding the potential benefits and some would even be reluctant to try to understand” [A1]

5.5.2 General view of agility in team B

In this team the usefulness and applicability of agile development is well understood. The project manager explains that:

“The maturity of the team, regarding agile development, is relatively high and developers understand the way of working”[B2]

There is however a big impediment for the agile way of working, pointed out by one of the interviewees:

“A broad picture for the development process has never existed. Goals are either too abstract or ambiguous”[B2]
5.5.3 General view of agility in team C

The overall view of agile development and the main objectives of practicing such a development method were common between all interviewees in the CGI team. A view that was formulated the following way by one of the developers:

"The agile way of working revolves around not planning too far ahead and at the same time sharing a general picture of what has to be done. A picture that is always subject to change." [C3]

One of the interviewees emphasized that this particular project stands out from previous experiences regarding that picture since:

"In previous cases the customer did not understand what agile meant. They did not trust it. That it works." [C2]

Self-Organizing A

In this project, which is carried out in a business that has no previous experience of software development, very comprehensive work had to be done in order to facilitate an appropriate environment for the team. The usefulness and actual execution of practices, such as the daily stand-up meeting and the sprint planning, are very commonly understood by the team members.

There are, however, some parameters causing a negative effect on the possibility for cross-functionality. One of the factors is that all team members have predefined roles. Roles that, in some cases, mean that work is exclusive to that member’s role. As described by one of the interviewees:

"Due to the fact that we have been forced to establish roles, I rarely have the possibility to help with other people’s tasks and vice versa. This often makes us dependent on that things are done in the right order, to avoid team members idling" [A3]

The same interviewee further explains that the project has been in need of people with certain expertise due to the many types of support systems used in the business and argued that roles therefore became a necessity. This expertise is often so precise that developers seldom can be assigned tasks from other areas than their own, which is costly for cross-functionality.

As mentioned earlier, a previous mapping of current processes within the business has been the foundation for the work in the team. One interviewee explained that rigorous work has been done by management in order to leave as much room for creativity and self-governance for the team as possible:

"How things actually should be done, on a detailed level, has never been influenced by management. This process has purely been done by the team. We try this and we try that and finding the path is up to us" [A1]

This is also concurred by one of the other interviewees:

"Weaknesses in the business’s current processes were found previous to this project. How to solve them is entirely up to the team." [A2]
When asked what their biggest obstacle in working in an agile way, all three interviewees unanimously said that the biggest issue was working alongside with external suppliers. It was also mentioned that this had been a problem in the past as consultants were promising more than they could deliver, and therefore delayed other team members’s work. The interviewees also mentioned lack of communication from these consultants as a big problem. Some consultants worked from home and did not answer emails and phone calls, which created irritation in the team, as they could not plan their work because of uncertainties about what the consultants would deliver.

**Self-organizing B**

Due to the lack of clear goals and flow of requirements to the team, autonomy has almost become a necessity in order to produce anything in the team. One of the biggest factors in order to keep the team productive and not just working blindly is the fact that this team is also responsible for running and maintaining the old platform (PPI5) and therefore possesses relevant product knowledge. When asked about the low belief in the team’s ability to keep a consistent level of productivity, one of the interviewees answered:

“In many cases we work but...We are not entirely sure what to do, we have to take chances... Maybe not take chances, but rather work based on hypotheses instead of plans”

[B1]

One interviewee further explained that this also induced complications in the way of working:

”Working this way, with too much autonomy, the team might not get anything done. We do a little bit here and a little bit there.”[B2]

As the team now has entered a phase where requirements are delivered by the requirement groups (which are used to a plan driven way of doing projects), issues have arisen in the team’s ability to control their work:

”There is a fear that has been escalated within the requirement-groups where they are worried that the team is becoming too autonomous. This escalation has resulted in business representatives demanding irregular demonstrations just to make sure the team does not develop too far ‘in the wrong direction’”[B2]

The same interviewee also pointed out that this behavior has affected the team in many ways. The process is time consuming but especially damaging for the ability for controlling work:

”Business representatives can directly call developers, making it hard for developers to say no or propose different solutions to matters”[B2]

This is concurred in another interview:

”We had a problem with adapting blindly to demands from the business, without questioning what they really want” [B3]

The team is, however, working actively with these issues through many channels. Mainly by building awareness and understanding with business representatives in the team’s way of working. This is done in order to facilitate efficiency in developers’ daily work and in regular, formal, demonstrations.
5.5. Result from interviews: Perceived agility in teams

Self-organizing C

As previously mentioned, the detailing of requirements and definition of specific tasks are carried out by the team themselves. This is a job that follows after an initial prioritization within each scheduled package containing user-stories and more overarching requirements. As stated by one of the interviewees, this is something done by the team themselves:

"...all these user-stories are classified in difficulty and size in order for us to decide what to do first. This is done in order for us to understand inter-dependencies, we cannot begin building the top floor before laying a foundation." [C1]

Feedback given from stakeholders on solutions developed by the team are rarely a controlling factor for team members' daily work, and is rather given on a more abstract level (work flows in the system etc.). One developer explains this phenomenon:

"Our developers never feel locked down in either front-end or back-end development. In one way, it feels as if we are the owners of the product, and that we are in charge. The customer rarely comments detailed work." [C2]

As previously mentioned this has not been the case throughout the project, and as stated by the same interviewee, feedback given on more detailed levels can have bigger effects in the team:

"...there were a lot of changes induced by the customer, which meant a lot of overhead work. You could get stuck in their way of thinking when there was potentially a much better way to do things. This obviously impedes creativity" [C2]

When it comes to roles and cross-functionality within the teams, all interviewees agree that team members are both able and comfortable with working outside their spectrum of competence. There are however appointed roles in the team, which are foremost explained as important in order to promote ownership [C1]. This is elaborated further by one of the interviewees:

"Different roles never means you do work in that area alone, it does however mean you are accountable for things done in your area. Of course, there is a danger for people to only care about their area and not do anything else, but it's also a form of leadership, to take more responsibility" [C1]

High-bandwidth communication A

The consultants, which make up the bigger part of the team, is located in an open-space office to promote quick and informal communication. All interviewees pointed out that most of the managerial work carried out has been done in order to facilitate communication and structure within the team.

There are, however, concerns regarding the ability to always maintain informal communication within the team, due to the fact that some of the team members do not work full time in the project:

"...scenarios can emerge when workshops are scheduled, and one of the participants has to urgently resolve another matter. This can, of course, leave my work suffering" [A3]
5.5. Result from interviews: Perceived agility in teams

High-bandwidth communication B
The team has three stand-up meetings every week, meetings with which the interviewees express satisfaction. One interviewee however, mentions that:

“Some of the less vocal team members often have a hard time expressing themselves in the group” [B3]

The same interviewee also expresses another concern that indirectly affects the ability for high-bandwidth communication:

“My daily work often consists of more formal meetings with business representatives. This of course disrupts my ability to work with development with the team” [B3]

High-bandwidth communication C
The CGI team is co-located and shares office space in order to facilitate quick communication, which is pointed out by all interviewees to facilitate high bandwidth communication. After an initial three-week planning of the sprint, most decisions within the team are taken "over the desks". One of the developers further explains:

“Lets say that 4-5 developers commonly found a solution during a meeting and a few days later found this solution to be deficient, no formality in how to find a new solution exists. Determining a new solution could be done “over the desks” or, if this is considered a critical task, could resolve in a more structured meeting” [C3]

Customer collaboration A
When the project initially began, the problem with stakeholder feedback became obvious. Stakeholders were not ready for change and did not see what they could gain from investing time in the project. However, the team was able to start working based on the thorough mapping that previously had been done of the processes:

“In the beginning, when we started development, we were forced to make assumptions. That resulted in a quite big backlash from the business” [A3]

The backlash that the interviewee mentions occurred when the team had the first official demonstration of a working part of the developed system. The demonstration was met with mixed opinions from the business and some of the negative critique caused the development to make a halt in that sub-project. As pointed out by one of the interviewees the backlash brought some positive effects:

“...it triggered a conflict, but at the same time this became a turning point for the stakeholders as they understood that if they wanted results, they had to participate in the [agile] process” [A2]

In the projects current state, a much more structured collaboration between the team and the stakeholders has been established:

“...and since then a much tighter communication with reference groups has been formed. To have a well working interface has definitely facilitated collaboration” [A3]
Customer collaboration B

All interviewees agreed that one of the main issues in communicating with customers is the lack of a structured interface between the team and the stakeholders. As described by one of the developers, external noise can be a big issue:

“...but I believe it’s pretty hard to know how and who to speak to, since we have so many different customers. There are simply too many chefs in the kitchen” [B1]

Another obvious problem in the communication between the team and requirement groups is the clash between the plan driven nature of the requirement teams and the agile way of working in development. One of the interviewees said the following on the matter:

“When the requirement groups first presented requirements, the development team was almost one year into implementation. At that point the developers had had no possibility to demonstrate any concrete solutions, because of lack of requirements. When the requirement groups realized this, they ironically put even higher demands on the development team.” [B2]

A possible solution to these problems would be to have someone with a more rigid product owner role, as pointed out by one of the interviewees. The same interviewee also points out that such a role could have additional functions:

“...a “protector” of the team, who gives them the authority and confidence they need and at the same time ensures that no team member is affected by external opinions or comments” [B2]

Such a protector is also pointed out as a possible solution to a problem in the team where team members sometimes feel individually liable for solutions, lacking a collective ownership of the team’s output.

Customer collaboration C

As previously described, the collaboration between Trafikverket and the CGI team is set up with a rigid, predefined agreement in the way of working. The continuous facilitation of feedback from stakeholders to developers is done through indirect communication:

“...and then there is some informal discussion with people internally responsible for requirements who are responsible to make sure that my suggestions are in line with business demands. I think this communication channel works well, even though my comments to people responsible for requirements are only an extension to communication with the customer. In previous experiences, I have had the responsibility of direct communication with the customer, a process that can become very time-consuming” [C3]

Another interviewee further explains the importance of the smoothly functioning interface between the team and different stakeholders:

“Without the short way to communicate with customers, in form of a product owner, team members are left clueless if they feel that they misinterpreted incoming demands and want to resolve that issue quickly”) [C1]

5.6 Results from interviews: Decision-making in the teams

The results presented under this section were all collected through the interviews held with members of the software development teams. The result will be divided into the same three paragraphs that were used to present the data from the survey.
5.6. Results from interviews: Decision-making in the teams

**Prioritization A**

The team at Söderenergi used overall goals when doing the prioritization within the project [A2]. The team also prioritized by trying to deliver the parts of the project that added most value to the organization first [A1].

"We understood quite early that a new purchasing system was what we should start with. It added the greatest value. We understood that there was a lot of money to be made by making the whole organization do purchases the same way" [A1]

Another strategy used to prioritize their work was to plan what had to be done chronologically, in order to proceed with other tasks marked as "must tasks". When working with such a big re-organization of IT structure, which includes multiple separate systems it has been a challenge for the team to actually prioritize correctly and finalize and launch subsystems.

The product owner of the project has a background from the business side of the organization, and not an IT background. The roles’ purpose within the team is to:

"Make sure the work has its foundation in the business needs. And to make sure that the business part of the organization is part of the process all the way" [A1]

This part of the prioritization process has improved later on in the project. The team now uses reference groups to get the support needed for developing a product that supports the organization. [A3]

**Prioritization B**

Although the structure of team B has improved since the new project joined the team, the team still struggles with prioritization. One interviewee states that:

"In which order to implement things, what is most prioritized, that is the biggest challenge" [B2]

Another interviewee shares this thought, and adds another factor to the prioritization process:

"Sometimes when we know that we need to build something and then get feedback on it, we have to prioritize that, since getting feedback can take quite a while. And while we are waiting for feedback we can work with something else" [B3]

The interviewees agreed that the team had become more goal orientated after the improvement of structure in the project and that they had started to prioritize to finish the sprint goals before starting on other tasks. One interviewee stated that:

"Earlier the tasks were not very specified and we did one task here and one task there. Now it is more "Let's finish this block first”" [B1]

One interviewee also found colliding requirements to be a big impediment for prioritization. It was stated that:

"When the same work group puts down colliding requirements it is very hard for us to make the prioritization, and instead you have to tell them to tell us what we should prioritize” [B3]

It was also stated during the interviews that:

"Without specific requirements it happens that you have to take a chance of what is meant and work out of a hypothesis of what we should deliver” [B1]
Prioritization C

The CGI team has divided the project into nine “packages”. This "sectioning" can be seen as a rough overall prioritization of the project, meaning that requirements included in the first package is prioritized higher than a requirement in package two and so on. Besides that, the CGI team have based a lot of their prioritization on what had to be built in order to move the project closer to delivery. One of the interviewees stated that:

"We have to build the systems in order so that they support each other. We can’t build a four-story building and start with the top floor" [C1]

The team has also classified all their user stories into categories such as easy, hard, big and small. One interviewee further elaborates on the importance of building functionality in the correct order:

"We have to start building up-stream, for example we need to be able to create an exam before we can execute one. We cannot start at that end, because then we cannot deliver something to the business before the whole project is done" [C3]

What the interviewee means is that although progress could be made working in a different order (an order preferred by the customer), some prioritization is made based on what makes most sense in a technical aspect. Although this is top of mind to the team, there have still been occasions when the management team’s prioritizing during the planning phase has failed. This was partly caused by the lack of programming knowledge. However, when such errors have been detected the team does a re-prioritization to cover the issue.

Conflicts and uncertainties about directions A

In the interviews, it was mentioned that the team knows where the project is aimed to finish. One interviewee mentioned that this has not always been the case. It was stated that:

"We worked autonomously, and we got no feedback on what we did well and what we had done bad. It has become better later on; the business is now more involved and have an insight in what we are doing" [A3]

Another challenge faced by the team has been uncertainties about what the business would like as a final product. In order to solve this issue, the team invited business experts to workshops with the goal to clarify any uncertainties that were still remaining. Although some business experts showed up, far from everyone did:

"Some persons felt they couldn’t put 4-5 hours into talking at a workshop. Now, I think, they have finally realized that they have to do so, because otherwise they will not be able to get a good system" [A3]

Conflicts and uncertainties about directions B

During the time PPI5 has been used there seems to have been some confusion about what the platform should be used for. This led to a platform that had been configured into a complex system hard to handle, which in turn made the team scared of further adjustments because of the risk of old modifications crashing after the new change.

When internal conflicts arose the team usually solved them with discussions. One interviewee stated that:

"When an internal conflict arose, there would be a lively discussion. I guess we have had a problem with that since team members that are not as vocal as others have a tendency to not get heard" [B3]
It was also mentioned in the same interview that:

"There is no one in the team that is only satisfied with their own solution. Everyone is responsive when it comes to discussion, and there is no prestige involved when decisions are being made" [B3]

One interviewee further elaborated on a possible cause of why sometimes there were external conflicts in the team’s close environment:

"I think that it was a problem that there was no common picture for the project. Nothing was structured and it was very unclear where we would end up" [B1]

All three interviews stated that it was not unusual that team members got pulled of the project during the sprints. They saw this as a major impediment in regard to time estimations made by the team. When asked why the team thought they were bad at time estimations one interviewee responded:

"It is because we are used to getting resources pulled mid sprint, I think that’s the biggest reason" [B3]

Conflicts and uncertainties about directions C

It is not unusual that the developers and the requirement engineers have different opinions on the essence of a user story. This is usually solved with a discussion between the two parties. Although, sometimes it could take a little time to solve the issue, one interviewee mentions the importance of having such discussions:

"That’s where it sometimes clashes. But I wouldn’t say it’s negative. Usually we get some really important stuff out of it” [C3]

Although the interviewee values the discussions he understands the benefit of having a requirements team:

"That’s the requirement group’s job, to make sure we produce what the business wants, and what it should want. It’s hard if everyone should have that responsibility individually" [C3]

It was also mentioned during the interviews that the decisions made are not final:

"It happens that we make a decision, then two days later when you’re working on it you understand this is not working. When that happens you usually just make an informal decision to change it, and if it’s a bigger issue you have another meeting to discuss the issue”[C3]

Shared decision-making A

Most of the decision-making at a low level within the team at Söderenergi is role based. When the team members get tasks to actually work with, they are mostly goal and purpose oriented and do not describe in detail how these goals are supposed to be reached. One interviewee states that:

"In my area I have most of the authority on how to do things. It is my knowledge, and no one else has the knowledge needed to make the decision, so of course my opinion carries weight” [A3]

It was also stated in the interviews that during the sprint planning meetings the team voted on the decisions that had to be made. However, it was also stated that:
5.6. Results from interviews: Decision-making in the teams

“Everyone is making the decision, but if I say it is going to take $x$ hours, and it is close to my area of expertise, nobody is going to object to that” [A3]

One interview subject also stated that:

“I think shared decisions would have a bigger impact if we were less role based. In case we had two of each role instead of one” [A3]

The same interviewee followed up by stating that he appreciated having someone to discuss ideas with, and was missing that in this particular project.

**Shared decision-making B**

All interviewees agree that shared decision-making is something that the team values. This has been one of the things the team has worked hard with since the new project manager joined in the beginning of the year. One interviewee stated that:

“Now it is always group decisions before individual decisions in the team” [B2]

Most of the decisions made by the team is are made during the sprint-planning meetings. During the sprint-planning meetings all developers get a chance to reach out for help about problems that have arisen during the latest sprint. This is done in order to make sure that all decisions are made by the group rather than the individual. According to the two developers interviewed, the decisions-making within the team are shared. It was also mentioned by the same interviewees that the most experienced developer within the problem area under discussion has the highest authority. This was stated as one of the reasons for the group being time optimists:

“They [the specialists] can be very optimistic about how long it will take” [B1]

**Shared decision-making C**

When it comes to shared decision-making the team has a procedure. Although it does not involve all team members, all parties that are affected get called to a meeting. In that meeting the parties with the most knowledge around the area have the highest influence on the decision:

“The relevant persons for this particular decision go into a meeting and discuss the issue. My opinion weighs less in certain circumstances, and more in others.”[C2]

When the parties can’t agree on a decision, the scrum master makes the decision:

“In the end, if we can’t agree the scrum master has the last word, but he always takes in our opinions into account”[C2]

This approach of shared decision-making is valued by the interviewees and one interview even states that:

“I don’t think it’s super important that everyone has an opinion about everything. It’s better to have an opinion only when the issue is regarding your area of expertise” [C3]

This kind of meetings are used when it comes to what the developer specifies as more important decisions. When a decision about a “lower level” issue has to be made it is usually done by one or two persons. One interviewee describes it as:

“I usually just walk over to the relevant person within the team and have a short discussion there and then”[C3]
This chapter contains analysis on the main themes of the report. The analysis will have a
cross-case nature and findings will be correlated with the theoretical reference.

### 6.1 Three different projects

Before we go ahead discussing and analyzing the results of this case-study, a very important
question needs to be answered; **are these projects comparable?**

All three projects studied in this thesis have a lot of commonalities and, of course, a lot
of differences. The first most notable similarity between the teams is the fact that all teams
are developing remastered software with roots in old solutions. All projects are also, on a
corporate level, managed in a more traditional plan-driven way with more or less adaption
to strengthen compatibility with agile methods. All team members expressed problems that
have occurred, either presently or historically, that were direct effects of a clash between an
agile nature of development and a stricter plan-driven type of management.

One of the biggest differences is the complexity of the software. In both team A and team
B, significantly more stakeholders are involved in the project compared to team C. These
stakeholders are, in many cases, very independent and often only interested in “their” part of
the projects results.

The lesser number of stakeholders could, of course, be one of the more obvious reasons
that team C appears to have, in many areas, the most successful adoption of the agile way
of working. As described in Flow, limiting the number of stakeholders could be one of the
most important facilitators to increasing productivity in an agile team [17]. However, from
this case study, the image of a lesser number of stakeholders in team C might be a projection
of a successful interface between the agile team and the business.

Another similarity between the teams is the external expectations. All teams are expected
to work according to agile principles and the teams therefore share, almost, the exact same
way of working. Within the teams, experience and the definition of roles differs slightly, but
members of all teams ultimately share the image of the way of working. We believe that
6.2 Self-organizing

In all teams, the agreed way of working is meant to facilitate the ability for self-organizing with sprint-planning carried out in a democratic manner. However, team B showed many of the negative symptoms of "water-scrum-fall"\cite{43}, such as the inability to receive overarching requirements in the project and business representatives trying to directly control the daily work of the team members. As stated in the interviews, direct interference by business in the team’s work makes it hard for members to propose alternative solutions and voice opinions in different matters. This is an obvious impediment for the ability to self-organize and is also, according to the interviews, one of the factors that lowers the team’s ability to maintain constant productivity. The mindset, and the actual planned way of working for the team-members implies lesser granularity in requirements given to the team. The requirements set for team A and team C on the other hand was initially on a far less detailed level.

In contrast, neither the members of team A nor team C showed signs of this symptom and interviewees in both teams point out that their ability to organize their own work is strengthened by the commitment in the project. The members of team C did mention that they were initially affected by a controlling factor from the stakeholders and that this caused an impediment for creativity. Interviewees from the same team also pointed out that this change came from a growing respect to the agile way of working. In team A, this respect was established from the start of the project but got challenged later on. The team were then met by another symptom of agile development in large organizations where they suffered from the effects of "isolation" \cite{9}. When team A made their first demonstration of their work, they were met with very mixed opinions from the different stakeholders, which decreased the level of respect for the team.

As the ability to govern the planning and execution of work has been compromised during the project in team A due to the problematic situation with external dependencies. All interviewees from team A said that it also has had an impact on the team’s morale. This further supports the importance of team governance but also points out the importance of a team having the right competences in order to ensure the ability for self-organizing.

6.3 High-bandwidth communication

When it comes to high-bandwidth communication all teams answered very similarly in the survey, indicating that all teams felt similar comfort in their way of communicating. This was no surprise, considering the fact that all teams share an open-office environment and generally were in agreements on how to communicate within the team. This further supports Flows theory that the agile team is extremely dependent on a surrounding environment that supports the way of working.

In the interviews with team B, it was pointed out that members of the team could feel that their daily work was often disturbed by overhead work, such as meetings with business representatives and therefore contradicted the essence of high-bandwidth communication. In this case, a more plan-driven and formal way of working in other parts of the project organization seems to be the cause of the problem.
In team A, one of the interviewees explained how part time members of the team could have a negative effect on the ability for high-bandwidth communication since they often would be forced to resolve other, more urgent matters. The reactive nature in the way of working for the part time members is in this case a big factor. As seen in team A and team B, one could argue that the team’s ability to communicate in a high-bandwidth manner is not only affected by a common way of working within the team, but also by the way of working in the whole project organization.

6.4 Customer collaboration

That the members of team C were a lot more satisfied with customer collaboration was not a big surprise, because of the organizational structure of the team. As described by Serrador and Pinto [33], managing these kinds of projects in a more agile manner facilitates a more common language and a common vision between management and the team. This is described by Serrador and Pinto [33] and Bannink [2] as two of the most important tools to mitigate issues surfacing when mixing plan-driven management and agile development. One of the interviewees in team C did, however, mean that management’s adoption to agile principles did not affect the work in the team considerably. We do not agree with this assumption as we believe that management’s way of working has a big impact on how the development team can carry out their work.

Even though Team C only perceive the benefits of management’s way of working as getting fast feedback and that they are trusted, we think that it has a greater impact. First of all we would argue that unless management accepted and trusted the agile methodology, the development team would not be allowed to work as independently as they currently do. Neither do we believe that the interface between the two parties would work as seamlessly if management was working in a more plan-driven manner. Following these two arguments we also believe that it can be hard for a development team to fully understand the impact management’s way of working has on the team. This might be because it is often easier to reflect upon problems rather then on parts that already work well. Our view in the matter is of course also based on the contrast with team A and team B, where agile acceptance and understanding is missing in parts of the project organization.

6.5 Prioritization

It was mentioned in the literature that multiple customers could impede a team’s ability to prioritize. Although all three teams had similar answers during the survey, there was one major difference that came up in the interviews. Both Team C and Team A had a product owner working closely to the team. At Team B when asked if there was a product owner in place one of the interviewees answered:

"Eeh, yes, I think so. I am not really sure about her name though"

This clearly shows that the role of a product owner was neither present nor defined in the team. As a possible consequence, the development team found themselves with colliding requirements from the organization, not knowing which requirement to prioritize. Another consequence that emerged because of the lack of a product owner could be seen at the multiple communication lines in and out of the team. We were told that people from the organization contacted the developers directly, trying to ensure the requirements they had created were to be prioritized. It became obvious that the team were lacking a standardized communication channel to and from the organization. We think that a product owner could,
6.6 Conflicts and uncertainties about directions

if not eliminate, at least decrease the effect of these obstacles at Trafikverket.

Another challenge in prioritization that became obvious is how specialization can impede prioritization. Team A, at Söderenergi is role-based, and with good reason. The program at Söderenergi is so wide, with so many separate systems, having a team that is fully cross-functional is hard to achieve. However, this does create a challenge when it comes to prioritization. One of the interviewees stated that:

“If my highest priority task is dependent on someone else’s fifth highest priority task, I will have to wait until I can complete mine”

If the team were more cross-functional, the first member could complete as much as was needed of the second member’s tasks, in order to complete his own task.

It was also made clear in the empirical data the importance of having a common picture of what the project should achieve. In the literature, it was described as: how can the developers make sure they are working on the right things? The team at Trafikverket seemed to lack this common picture which seemed to create confusion about which tasks to prioritize. This was made obvious during one of the interviews where it was stated that:

“Earlier we did one task over here, and one task over there. Now we have more focus on “lets complete this function first””

A common picture of what the project should achieve seems to be of absolute importance and without it, it seems easy to lose track of the scope of the project. This problem, although it was expressed in a different way in the literature, is one of the major obstacles regarding prioritization when working with agile software development.

6.6 Conflicts and uncertainties about directions

In big organizations, such as the organizations studied in this thesis, vision usually differs depending on the employee asked within the organization. Walker[5] stated that:

“The most significant success factor for project teams is that they have a common and shared idea of what difference they are trying to make as a result of the project”

Lack of such an idea was obvious during the collection of data made at Trafikverket. Instead of aiming to reach the overall goal, the team got stuck in trying to accommodate every wish the employees at Trafikverket had. This led to a project not moving closer to launch, since much of the basic functions were not yet completed. We believe that an overall vision for this project, along with clearly defined goals would have been of great help for the team in their journey between the beginning and finalization of the project. A product owner that is fully involved in the project could also be of great help, since a part of that roles mission is to protect the development team from noise coming from the organization.

Team B was also the team that experienced the weakest filter against external noise, which can be seen in the figure [6.1]
6.7. Shared decision-making

In previously written literature researchers have pointed out the value of keeping external noise to a minimum. In the interviews in two out of the three teams (team A & team B) in this study it was stated that external noise did have an impact on the daily work. While the team at Söderenergi did not think of it as a major implication and more as a natural consequence of agile development, the team at Trafikverket seemed to struggled a lot more with the issue. This leads us to believe that it might not necessarily be the amount of noise that reaches the team but rather the impact that particular noise has as it reaches the team. We also believe that having a shared vision within the team, as Flow suggests, would diminish the impact of external noise as it gives the developers themselves a better overall picture of what is tried to be achieved with the project.

In team B, it had also been quite common that team members got pulled from sprints, either temporarily or permanently. Both Drury et al [12] and Moe et al [27] also described this as an issue, mostly in regards to being able to do time estimations which was mentioned as an issue in the interviews. Also, we believe that it affects how committed employees feel towards their work. When the rest of the organization does not accept the agile way of working, which arguably could be the case when pulling team members out of the sprints, the commitment to the project seems to decrease. The reason for this could be that when management prioritize other projects above the agile project, this affects the team’s view on how important the project is.

6.7 Shared decision-making

All teams used shared decision-making. However, in what fashion, and the reasoning for shared decision-making varied. Moe et al [27] specified specialization as one of the obstacles to shared decision-making. Although the team at Söderenergi felt they used shared decision-making, all interviewees said it was role-based. In order to make a decision the team voted, which one could argue would make the decision shared, but on the other hand it
was against the norm not to vote for the solution that was presented by the team member in charge of the subject. One could argue that this norm might hold back the team’s creativity as team members with less knowledge about the area would not argue for their solution. As a consequence, the team might keep to old patterns, and not see improvement areas.

In team C, the shared decisions were knowledge-based. This led to more parties being able to take part in discussions. One could argue that this is the ideal way of making decisions in agile software development. However, with more opinions, reaching a decision that satisfies all team members could become harder. Although the team at CGI had not yet experienced this, it could become an issue in the future. When asked how such an issue would be solved, one of the interviewees said:

"I guess the Scrum master would have the last word in such events."

In team B shared decision-making was used for another reason. The team highly valued shared decision-making since it was seen as a tool to increase the willingness to try new solutions. It was mentioned in the interviews that it was important that no one should be singled out when a solution did not work as intended, and rather see it as it was the whole team’s fault. These thoughts seem very healthy since improvement often comes as a consequence of failures.
Considering the amount of possible influencing factors, that comes with a study of this nature, it is close to impossible to draw any straight conclusions. Bearing this in mind, we will discuss what the data collected in this study suggests and also discuss the method chosen for this thesis, the validity of the results and possible ethical concerns connected with this kind of study.

7.1 Results

Self-organizing

The biggest impediment in successful self-organizing for agile development teams in large organizations was, surprisingly, found to be external noise rather than a lack of initially given control by project management. In some of the occurrences, this external noise was found to be caused by a lack of acceptance to the agile way of working. The most obvious example of this behavior was found in team B, where parts of the organization were very afraid that the team were to develop “too far in the wrong direction”.

As stated by Kallman and Kallman, an adaption for agile development from management’s side could be as important as an adaption to management’s way of working from the agile team side. In the team B case, a heavy clash between plan-driven project management and agile development were found and were obviously a big impediment for the ability to self-organize. Even though we cannot fully analyze it, due to the cases’ unit of study were limited to the team rather than both team and management, we believe our results point towards a much-needed adaption both from the team and from the workgroups in the PPI6 project at Trafikverket.

As found in all teams and especially in team A and B, the ability to self-organize was challenged when developers felt a lack of trust for the process. This therefore also suggest that, from the developers point of view, the ability to successfully self-organize is not only dependent on freedom given to the teams but is very dependent on the commitment felt by team members. Commitment that is based on the ability to handle the external noise and to maintain a common vision among team members. Both of these factors are mentioned as key
parameters of Flow[17].

What is not mentioned in Flow, that we found could heavily affect the team’s commitment and therefore ability to self-organize, is the matter of external dependencies. In team A, a large blow was made to the team’s morale because of the fact that governing power were lost due to external dependencies. What is interesting is that this part of the results suggests that, even though an agile team is given control by management, an ability to successfully self-organize can always be disrupted by factors outside the project organization. Even though Flow thoroughly discusses these symptoms, they point to multiple stakeholders in the project as the root cause. Our study, on the other hand, suggests that being dependent on e.g. external consultants is another threat to the ability to self-organize.

**High-Bandwidth communication**

One of the least differentiating areas of the case study was the team members’ perception of their ability to efficiently communicate within the teams. From our point of view this is very interesting and we believe this result wraps up some of the core concepts presented in Flow. As the findings of this study show, all teams are comfortable in their way of working and, in all cases, tend to place different impediments for communication outside the boundaries of the team.

The results of our thesis therefore suggest that a big part of the facilitation for high-bandwidth communication within the team is constructed outside the boundaries of the team. As presented in Flow, agile principles need to be understood by higher layers of the project organization in order to mitigate these impediments[17]. We claim that our results supports this conclusion.

**Customer collaboration**

The results of this study definitely highlight the importance of a well working customer collaboration for agile development. We found that all teams were extremely dependent on the customer’s devotion to the projects and the possibility to receive fast feedback.

The most important factor for drawing any conclusions regarding customer collaboration is the fact that one of the teams, namely team C, differs a lot regarding their communication interface to the customer. This team, compared to the other teams, is the only one where an effort to vertically scale the agile way of working has been made. The results of this study suggest that this scaling has greatly affected the team’s perception of customer collaboration.

As found in the study, all interviewees of team C meant that management’s way of working did not directly impact how the team was functioning. The results, however, suggest that even though developers might not directly feel affected by a vertical scaling of agile development, the adaptability of the customer to a fast pace and frequent communication is the key to a successful customer collaboration. We further mean that the results suggest that a customer’s adaption to an agile way of working is the shortest path to get there.

This conclusion partially reflects the corresponding concepts presented in Flow. Kallman and Kallman[17] mean that being agile in higher layers of an organization, i.e. vertical scaling, does not imply management or customer practicing "team-level" agile (scrum etc.), but merely embracing core principles of agile development, and that trying to do vertical scaling of "team-agile" could in fact be harmful in the long run. In the case of this study, the results imply that adopting "team-level" agile was very important. From the developers
view however, the most important outcome seems to be the embracement of the core values, rather than management practicing the actual "team-level" agile way of working.

Considering the scope of this study, we can of course not confirm neither deny that scaling "team-agile" to higher layers of the organization would be a problem.

Prioritization

The results of this study suggest that, irrespective of how the team and the project organization works, prioritization will always be a challenge. This could be seen as all teams, independent of the organization, struggled with prioritization. The reason for the struggle varied, but were very distinct in all teams studied. Because of the complexity of this challenge, this study can of course not present a generic solution to that challenge. However, the study has made it clear that some key factors can help mitigate the challenges regarding prioritization. It has also made clear that some attributes of a team can make prioritization more difficult than it has to be.

The study points out the importance of a product owner. Without a project owner, agile software development teams within large organizations seem to struggle. This could be seen clearly in Team B, who had problems directly relating to big quantities of external noise, which we believe would be mitigated by a product owner. We also believe that the need of a product owner increases with the number of stakeholders the project has. We suggest that the role as "product owner" is not necessary but we do believe that a role that has its main focus on the interface between team and organization is necessary, regardless of what the title of that role might be. One of the biggest challenges, but also one of the most important ones, for a product owner is most certainly going to be to get several stakeholders to agree upon what the project should achieve. This might seem as an easy task, but our study shows that it usually is not. This was seen in Team B where stakeholders could call developers to add or edit requirements to serve their needs. With a common picture of what the project should achieve, these interactions would be unnecessary.

We also found that a certain amount of specialization could be preferable when prioritizing tasks within a project. The results of the study indicate that specialization seems to decrease the amount of time needed to reach a final decision, as specialization was used in all teams, in various degrees, in the decision-making process. However, specialization seems to also have a negative impact on an agile software development project. It was apparent in team A that the specialization affected the team's ability to reach the final goal, negatively. The reason for this was the sequential nature of a specialized team. In order to continue with a task, you as a team member could see yourself completely dependent on some other team member's work.

The issue regarding too many stakeholders is considered to be one of the main challenges that Flow mitigates. One of the main purposes of the define and distill phases of the framework is to mitigate this exact problem. The second issue discussed above, on the other hand, is not mentioned as a problem by the authors of Flow. The authors mention the importance of Right people but do not specify if those persons are specialized or not. This leads us to believe that they might think that specialization is preferable in some cases. We, however, believe that is not the case, and that heavy specialization will impede the prioritization of an agile software development team. This was seen in Team A, who often struggled with prioritization problems.
7.2 Method

Conflict and uncertainties about directions

Two out of the three teams studied during this thesis experience external noise. However, during the interviews it was made clear that the two teams reacted to the external noise in two different ways. Team A did not seem to be affected by the noise, while Team B experienced it as a big challenge. We believe that the main reason for the two different effects of the same issue, is the vision that the team members in Team A shared. The results of this study indicates that a shared vision does change the impact that external noise has on an agile software development team. The results also indicate that a common vision, whether it is seen as a useful tool or not, is beneficial for an agile software development team’s ability to deliver results. This conclusion is further supported by the established vision in team C. This vision was not seen, by the developers, as a key moderator for success. On the other hand, project management felt that it had a big impact on the team’s ability to deliver results.

The study also pointed out the importance of keeping the team consistent throughout the whole project. One of the teams in the study regularly experienced team members getting pulled out of their ordinary work in order to work on tasks within the organization that was prioritized higher by management. The data collected during this study shows that this issue decreases the commitment that team members have towards the project. One of the other teams, team A, experienced this, but to a far lesser extent. Even though it did not happen regularly, the team members in Team A felt that it prevented them from carrying out their work.

Flow proposes that a common vision is of uttermost importance when building a successful agile software development team[17]. The results of this study support that theory, as it can be seen that a common vision has a positive impact on agile software development teams. The other issue pointed out above is not considered in the Flow framework. We, however, believe that it has major impact on the team’s ability to deliver results. The requirement of having a stable team (same team members) throughout the sprints might sound obvious, but our study suggests that it is an issue that teams struggle with.

Shared decision-making

All three teams studied had different ways and reasons for shared decision-making. From the data collected for this report it is not possible to draw any conclusions in regards to why and how shared decision-making should be used. What we could see is that the three different teams used it differently, and they all had their individual reasons for why they used it. Although the whys and hows differed, all three teams spoke about shared decision-making in a positive manner making us believe that the whys and hows are not important in regards to the issue. For instance, we mentioned earlier that agreeing upon all decision might be time consuming, but the team that tried to reach an agreement never mentioned it as a problem. Instead, we rather believe that it is important that it is used. The reason we believe this is that although the decision-making differed, no interviewee expressed anything negative about it. We also believe that shared decision-making makes team members feel that they belong and are a valuable part of the team.

7.2 Method

We chose to conduct a case-study to collect the data used in this thesis. Although we felt this was the method best suited for this study there are arguably flaws to it. One of the flaws that is hard to get around is that the subjects of the study are human beings. From time to time it felt like some of the interview subjects tried to say what the researchers wanted to hear, instead of describing the real situation. We tried to mitigate this issue by making clear
before the interviews that we just wanted them to explain how they themselves perceived their work place and team.

As discussed by Runesson et al.\textsuperscript{32} the teams under study in case-studies are usually chosen upon availability, and not because of their characteristics making them suitable for the objective of the study. One could argue that the three teams are not similar enough to make a comparison between the three of them. One example of this could be that the projects, although carried out in similar organizations, differ quite a lot in the three separate cases.

It could also be argued that the co-sourced team should not have been included in the study since they are working on a project that they could be dismissed from if the result of the study described them as a dysfunctional team, or if negative comments were made about the employer. We do not think this was the case, as all answers appeared truthful but it could however be the case.

Because of time- and resource limits only three team members of each project were interviewed. This could have two possible consequences. Firstly, the interviewees did not feel completely anonymous which could lead to answers that do not reflect the reality entirely. Secondly, it could give us a picture of the team that does not reflect the whole team’s opinions.

The survey that was sent out to the teams was written in English. We decided that English was the preferable choice in order to minimize dubious translations of domain specific words. Because English is not the mother tongue of most of the respondents this could have caused issues. For example, we understood during the interviews that the word “management” had separate meanings depending on who was being interviewed. We thought that the meaning of the word was quite obvious, but apparently, this was not the case. Also, some answers were inverted in order to create graphs that were easier for the reader to comprehend (i.e. bigger figure, better result). If the questions instead were asked in a way that did not need inversion, the answers might have been different.

If these questions instead was, from the beginning, asked in way that did not need inversion, answers might have been different.

All interviews were held in Swedish because this was believed to enable the interviewees to express themselves to a larger extent, not having to worry about a possibly insufficient vocabulary. This was not a problem, since Swedish is our mother tongue as well which meant that both the interviewees and the interviewers could express themselves freely. However, although this is not an issue for the conclusions of this report, we still wanted to address that this could have been an issue if the conclusions were to be based on the English quotes instead of the Swedish ones.

Although the response rates in the different teams were quite high, reaching from at least 70% to 100%, the response rates could still be an issue. Because of the small sample sizes, one or two persons with opposing opinions would change the overall average of the team substantially. One could also argue that a person that appreciates their current team and work place would be more inclined to answer the survey then a person that thinks the opposite, giving the team a score that does not reflect the reality.

7.3 Validity and reliability

7.3.1 Internal Validity

Both multiple units of study and data sources were used in this study. Several measures were taken in order to approach all data sources from the same angle such as interview
7.3. Validity and reliability

guides, introduction letters and data analysis tools. Of course, with a study of this nature, all validity ultimately rests on the authors ability to interpret, present and analyze the data. All these parts have been considered throughout the report and as this report had two authors, objectivity was strengthened and reassured through regular discussions on the topic.

A possible threat to the internal validity of this thesis is the likelihood for interview/survey subjects having hidden agendas. Since the members of each team has different work-agreements (external consultants, co-sourcing etc.), there might be a possibility for them to see various benefits from participating in the study. One may see it as a possibility to mention negative aspects in the way of working in order to induce a change. Other subjects may see it as an opportunity to shine, hoping it would generate additional assignments in the future. All interview-subjects in the study did however give the authors of this report an honest and trustworthy impression. With all interviewees mentioning several negative and positive aspects in their way of working, this further suggests sincerity from all subjects.

Quotes had to be translated from Swedish to English in this report and with that comes a risk that the interviewees are not being quoted correctly. This was tried to be mitigated by sending all the interviewees quotes to each person for proof reading. Another issue that arose was when quotes contained idiomatic expressions, which are perfectly understandable for Swedish speakers, but do not make sense for anyone else. These expressions were therefore re-formulated with other words, trying to keep the essence of the message in order to make sense in English.

7.3.2 External Validity

The external validity, which refers to the extent which results from a study is generalizable, could naturally be hard to assure in a case study as many external factors can affect the outcome. When the planning phase of this thesis was carried out, any generalizability of eventual results was seen as close to impossible. However, during the study, much more similarities than expected emerged in the surrounding environment of the studied units.

As stated by Runeson and Höst on the topic external validity:

"...for case studies, the intention is to enable analytical generalization where the results are extended to cases which have common characteristics..."

Given the multiple-case study approach of this thesis, and the amount of similarities found between the teams' surroundings, generalization to environments with similar characteristics should be considered possible.

7.3.3 Reliability

The results of this study are all presented as the outcome of a thoroughly described step-by-step approach. Repeating that approach would be highly possible. However, as Yin formulated it, a case study is used to:

"...investigate a contemporary phenomenon in depth within its real-life context"

In this study two of the three studied units were undergoing major changes in the way of working. The data used can therefore be seen as highly temporary. However, if one would overlook this factor and try to carry out the study in a near to replicated environment, the method should yield the same results.

Interview guides were used during the interviews and the nature of semi-structured interviews could be a problem in terms of reliability. Although most of the questions asked
were planned and asked directly from the interview guide, some questions were asked in order to follow-up on subjects stated in earlier answers from the interviewee. This could certainly have an impact on the reliability of the data gathered as some of the answers to these follow-up questions contained crucial data for the study.

It is also important to mention here that all conclusions drawn from the data gathered are based on the authors’ interpretations. Interpretations that could, of course, differ if the study were to be carried out by someone with another background or knowledge-base.

### 7.4 Ethical and societal aspects

Both Yin [46] and Runeson & Höst [31] stress the extensive need for ethical concern when conducting case studies involving human subjects. Yin [46] further claims that making subjects aware of actions taken from the authors’ side to ensure ethical concerns is as important as the actions themselves. All interviewees were of course given the option to deny participation. They were also, prior to the interview, briefed on how the authors intended to assure confidentiality reporting the results. Recordings of interviews were optional and interviewees were allowed adjustments to answers mid-interview.

Although the participation in the study was voluntary, team managers consent to carry out the study in the teams was given in advance. This could have as a consequence that team members felt forced to participate in order to not look bad in front of their manager. The results of this study were also presented to the higher management within the company which could have made participants feel uncomfortable with the situation. However, in order to establish trust with the handling of information gathered during interviews, all material used from each interviewee was sent back, to the interviewees, for feedback as suggested by Runeson and Höst [31]. The results were also sent back with an example of how the quotes were to be presented in the report. One of the interviewees edited some of the quotes and sent them back. This was done because the interviewee felt that the translations had changed the sense of some quotes. The interviewees edited quotes were later used in this report.

Because of time limitations, only three persons from each team were interviewed. This could enable certain people, probably involved in the project, to figure out which of the interviewees has said what in the interviews. If the number of interviewees increased, this would become a tougher task, even for someone with insights within the project.

It could be the case that, depending on the outcome of this report, the organizations would be more likely to, or not to, employ consultants working for another company. This could have as a consequence that employees or consultants’ roles get redundant. The result could also be a factor affecting the teams, and the team members’ status within their current organization, maybe making them less likely to be picked for future agile software development projects.

This report is one-sided, meaning the scope of the thesis is limited to present the perspectives of members in agile software development teams. This could of course have as an effect that people’s work gets belittled purely because there is a lack of insights on the researcher’s side. Reflecting the whole situation, from every perspective would be at risk for poor execution and has therefore been considered to be out of scope.
The main aim of this study has been to investigate agile software development teams and what made them function better or worse in large-scale environments and specifically how their internal way of working was affected by external factors. We have also studied how the concepts of a modern-day framework, used to mitigate problems in large-scale agile development, corresponds to our results. Both while studying the three teams participating in this study and when studying the Flow framework, our results showed that impediments for productivity in teams often seemed to be caused by factors outside the boundaries of the team. In the studied teams, the team members’ perspective on how well the teams were functioning, was seldom based on the teams’ internal way of working.

The research done in this thesis has shown that software development done according to agile methodologies is highly dependent, maybe more than ever, on a well-functioning interface between developers and more managerial layers of an organization. The Flow framework, from our point of view, is a very high level and, in certain aspects, ambiguous tool. However, the framework highlights aspects that differ from a common view of the adaption to agile methodologies for software development teams. Aspects that, in many cases, were supported by our studies. The insights gained by this research has led us to the conclusion that agile software development, which has arguably become industrial mainstream, is often being adopted as a set of practices rather than a set of principles.

Even though we consider the concepts of Flow lacking some granularity in order to be fully applicable when analyzing software development on a team level, the authors of the framework definitely pinpoint some of the more crucial areas also found being impediments for the teams in this study when it comes to decision-making and human-centric agility.

We now return to the formulated research questions from chapter 1 of this thesis, and comment the provided answers:

1. Is Flow a valid framework for analyzing human-centric agility of software development teams in large-scale environments?
All teams participating in this study have been given, at least on paper, the right tools to conduct their daily work in an agile way. However, the results of this thesis have shown that adopting agile practices in a team does not necessarily ensure a more agile way of working. The human-centric agility was found to be very dependent on both the amount of external noise, but foremost the members’ commitment to the project.

We also found that developers felt no direct effect of the presence of agile practices in management levels of a project organization, e.g. that management were using Scrum practices such as backlog. They did however feel very affected by management embracing agile principles, such as the continuous, flexible feedback loop. This is in line with the concepts presented by Kallman and Kallman. Even though an embracement of agile principles seems to be of high importance, rather than an actual adoption of agile practices, this study further shows how management’s adoption of agile practices seems to be the quickest path to embrace agile principles.

With minor deviations, Flow describes the same factors needed to facilitate a high level of human-centric agility as we found in the study. We therefore mean that Flow is a valid framework when analyzing agility in software development teams.

2. Is Flow a valid framework for analyzing individual developers’ ability to make justified and/or tactical decisions in large-scale software development teams?

We cannot confirm that vision has as big impact on decision-making as Flow suggests. However, we can confirm that a common vision has an impact on how teams handle external noise. Also, we can confirm that, as Kallman and Kallman suggest in Flow, external noise, especially many involved stakeholders, impede the team’s ability to make tactical and strategic decisions.

We also found that specialization, in contrary to what Flow means, can have a negative impact on teams’ ability to achieve their goals. The study shows that reaching a decision goes faster with specialized team-members. But the study also shows that specialization brings other issues like being dependent on certain team members delivering their part before being able to continue with tasks.

As mentioned earlier in this report, Kallman and Kallman suggest that group decision-making within teams enhances innovation and improves work satisfaction in software development teams. We cannot confirm or deny this statement but we can conclude that team members values shared decision-making highly.

Because of the many similarities between the factors that we found during this study and the issues and mitigations that the Flow framework highlights, our results indicate that Flow, although not perfectly, is a valid framework for analyzing team members’ ability to make well justified decisions within agile software development teams.

8.1 Future work

Our goal with this thesis was never to measure the potential improvement that an implementation of Flow is argued to have. In future work this would be interesting to measure. In order to do so we propose that future researchers perform a controlled longitudinal case study, where higher layers of an organization are introduced to the Flow framework and potential effects are measured on a team-level.
During the study, we found that the adoption of agile practices on a team-level does not necessarily facilitate teams’ agility unless management change their way of working as well. Existing research, such as the studies done by Serrador and Pinto \cite{33}, suggest that management’s adoption of “team-level” agile practices correlates with higher project success rate. However, Kallman and Kallman mean that scaling team-level agile practices to higher hierarchies of an organization could be harmful. Due to the scope of this study, we could neither confirm nor deny this claim. However, it would be interesting for future research in the area to further investigate potential problems for higher layers of an organization when vertically scaling agile practices.

As seen in this study, a challenge that most agile software development teams, working in a more traditional environment seem to struggle with is the unwillingness from higher layers of an organization to change their way of working. Therefore, we see a general need for further research in frameworks that bridge and join both worlds on principles rather than practices.

Finally, the results of this study were limited to three cases. Given the growing popularity of agile methodologies, larger studies, with more quantifiable results, will be needed to fully grasp and generalize parameters affecting decision-making and agility in software development teams.
Bibliography


[18] Kate Kelley, Belinda Clark, Vivienne Brown, and John Sitzia. “Good practice in the conduct and reporting of survey research”. In: International Journal for Quality in Health Care 15.3 (2003), pp. 261–266. ISQHC.


A Survey questions

All survey questions were answered according to a 9-point Likert-scale (1-9) and were structured in different themes. The answers ranged from "totally disagree"(1) to "strongly agree"(9). In this appendix the questions are presented under each of the themes.

Decisions

1. I feel that there are, occasionally, too many opinions when a decision has to be made
2. I feel that I’m empowered to make a strategic decision about the project I’m currently working on
3. I feel that decisions made during planning meetings are usually followed through
4. I feel that sometimes the subject of a decision during a meeting is too complex, making it hard for me to add to the conversation
5. I trust the consensus of my team, even though the decision is not aligned with my thoughts for the best solution
6. My team members’ authority in decision making is based on the knowledge they have regarding the topic of the decision
7. My team usually makes well justified decisions
8. My team members would listen to my perspective even though I have lacking knowledge around the area of the current discussion
9. My team members’ authority in discussions vary in regards to the topic

Tasks

1. Tasks change so rapidly, it’s hard for me to commit to them
2. Tasks that have been completed in earlier sprints often have to be redone
3. I would inform my team manager about issues regarding the project my team is working on
4. I can reach out to my team members regarding problems I encounter
5. My team has a good understanding of what tasks are most important
6. My team members complete the tasks they have started
7. My team members sometimes get pulled out of their assigned work to solve another issue outside the project

Organizing work
1. Team members can choose themselves which tasks to work on
2. Time- & work-estimations are done by the people who will carry out the work
3. My team members maintain a constant productive level, without being overworked
4. My daily work is planned, and carried through without, or with minimal, interference from my supervisor
5. My team can continue work as usual even though a team member is unavailable
6. My team has a common language to describe the work in the project
7. Management shares that language
8. My team is cross-functional and people are willing to work outside their specialities
9. My team's time estimations are usually about right

Communication
1. The teams daily meeting is effective in synchronizing work
2. My team has a clear and transparent communication line to management
3. My team voices all concerns about reaching goals openly
4. My team embraces change of plans

Customer collaboration
1. My team has a clear and transparent communication line to the customer
2. My team tries to accommodate changes requested by the customer with minimal bureaucracy
3. My team has no problems to reach our customer (through direct or indirect communication)
4. Requirements definition of done is directed by the customer

Goals and vision
1. My team is aware of the overall goal/vision for the outcome of our work.
2. My team is aware of the company's overall vision
3. My team's progress in the project is presented highly visible
4. My team has a good understanding what tasks should be prioritized
5. My team has the right resources to achieve our goals
Interview Guide: Developer

Generell formalia

- Presentation av examensarbetets syfte och mål.
- Presentation av upplägget för intervjun (semi-strukturerad).
- Förklara hur intervjuerna skall analyseras och återkopplas.

Bakgrund

- Beskriv projektet du arbetar i.
- Beskriv projektorganisationen.
- Beskriv den utvecklingsmodell teamet har.
- Beskriv din roll i teamet.
- Hur ser ditt vardagliga arbete ut?
- Kan du beskriva teamet och hur ni arbetar?
- Kan du beskriva visionen för projektet?
- Kan du beskriva processen då dina arbetsuppgifter formuleras?

Beslutsfattande

- Kan du beskriva hur ni tar gemensamma beslut i teamet?
  - Hur är då auktoriteten fördelad?
  - År dessa beslut skrivna i sten?
  - Hur förändringstolerant skulle du säga att teamet är?
  - Om det uppstår intern konflikt, hur löses en sådan?
- Om det skulle uppstå en extern konflikt, hur löses då en sådan?
• Kan du beskriva dina befogenheter för att ta egna beslut i projektet?
  – Känner du dig någonsin osäker på om dessa beslut går i linje med vision och mål för projektet?
• Kan du beskriva hur uppgifter prioriteras i teamet idag?

Det agila arbetssättet
• Kan du beskriva din bild av agil utveckling i stort?
  – Varför används det?
  – I vilka miljöer tror du agil utveckling passar bra/mindre bra?
• Hur tycker du teamet förhåller sig till agila kärnvärden? (flexibilitet, snabb återkoppling osv)
  – Är det några faktorer som du känner hindrar den agila utvecklingsprocessen i teamet? Hur skulle dessa kunna lättas?
• Hur känner du att teamet ges möjlighet att kontrollera sitt eget arbete?
• Vem är kunden/slutanvändaren till projektets resultat?
  – Hur känner du att feedback-loopen fungerar med kunden/slutanvändaren?

Övriga frågor
• Händer det att medlemmar blir otillgängliga under sprintar?
  – Hur påverkas teamet då?
• Hur kompatibel anser du att överliggande verksamheten är med den agila utvecklingsmetoden?
  – När det gäller kommunikation?
  – När det gäller formulering av krav/mål/användarfall?
• Hur tror du att en väl formulerad vision kan påverka arbetet i ett agilt team?
• Kan du beskriva teamets största hinder från planering till resultat?
• Finns det något annat du känner skulle kunna förändras i kringliggande miljö för att höja produktiviteten i teamet?
Generell formalia

- Presentation av examensarbetets syfte och mål.
- Presentation av upplägget för intervjun (semi-strukturerad).
- Förlägga hur intervjuerna skall analyseras och återkopplas.

Bakgrund

- Beskriv projektet.
- Beskriv projektorganisationen.
- Beskriv den utvecklingsmodell teamet har.
- Hur ser ditt vardagliga arbete ut?
- Kan du beskriva teamet och hur ni arbetar?
- Kan du beskriva visionen för projektet?

Överliggande verksamhet

- Hur kompatibel anser du att överliggande verksamhet är med den agila utvecklingsmetoden?
  - När det gäller kommunikation?
  - När det gäller formulering av krav/mål/användarfall?
  - Hur skulle verksamheten kunna agera annorlunda för att bli mer kompatibla?
- Hur ser teamets kommunikation med mottagare/kund ut?
  - Finns det några delar av den kommunikationen som du känner stjälper det agila arbetssättet?
Förmedling av vision och mål för arbetet

- Hur tror du en väl förnämnad vision kan påverka arbetet i ett agilt projekt?
  - Hur stor inverkan tror du det har i det här projektet?
- Använder du vision som ett verktyg i detta projekt?

Övriga frågor

- Vad känner du är det största hindret från planering till resultat i detta projekt?
- Agil utveckling innebär allt som oftast mer befogenheter för medlemmar. Vilka faktorer tror du är viktigast för att teammedlemmar ska använda dessa befogenheter på rätt sätt?
- Känner du att projektet har resursmässigt rätt folk för att uppnå önskat resultat?
- Finns det några problem du tror skulle kunna uppstå om teamet blev mer autonomt än vad det är idag?
### Grouping of questions: Human-centric agility

<table>
<thead>
<tr>
<th>Graph</th>
<th>Parameter</th>
<th>Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-organizing</td>
<td>Maintaining productivity</td>
<td>Organizing work.3</td>
</tr>
<tr>
<td></td>
<td>Team governance</td>
<td>Organizing work.4 &amp; Organizing work.8 &amp; Vision and goals.5</td>
</tr>
<tr>
<td></td>
<td>Individual governance</td>
<td>Organizing work.1 &amp; Organizing work.6</td>
</tr>
<tr>
<td></td>
<td>Cross-functionality</td>
<td>Organizing work.5 &amp; Organizing work.8</td>
</tr>
<tr>
<td>High-bandwidth communication</td>
<td>Ability to raise concerns about reaching goals</td>
<td>Communication.3</td>
</tr>
<tr>
<td></td>
<td>Effectiveness in regular meetings</td>
<td>Communication.1</td>
</tr>
<tr>
<td></td>
<td>Management communication</td>
<td>Organizing work.7 &amp; Communication.3</td>
</tr>
<tr>
<td></td>
<td>Accommodating change of plans</td>
<td>Communication.4</td>
</tr>
<tr>
<td>Customer collaboration</td>
<td>Accommodating feedback</td>
<td>Customer Collaboration.1 &amp; Customer Collaboration.3</td>
</tr>
<tr>
<td></td>
<td>Clear communication</td>
<td>Customer Collaboration.4</td>
</tr>
<tr>
<td></td>
<td>Definition of done directed by customer</td>
<td>Customer Collaboration.4</td>
</tr>
</tbody>
</table>

### Grouping of questions: Decision-making

<table>
<thead>
<tr>
<th>Graph</th>
<th>Parameter</th>
<th>Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prioritization</td>
<td>Tasks never have to be redone</td>
<td>Tasks.2</td>
</tr>
<tr>
<td></td>
<td>Follow the plans</td>
<td>Decisions.3 &amp; Tasks.8</td>
</tr>
<tr>
<td></td>
<td>Good understanding of prioritization</td>
<td>Tasks.5 &amp; Vision and Goals.4</td>
</tr>
<tr>
<td>Conflicts and uncertainties about directions</td>
<td>Follow through on plans</td>
<td>Decision.3 &amp; Tasks.2</td>
</tr>
<tr>
<td></td>
<td>Completion of tasks</td>
<td>Tasks.8</td>
</tr>
<tr>
<td></td>
<td>External noise filter (inverted)</td>
<td>Tasks.7</td>
</tr>
<tr>
<td>Share decision-making</td>
<td>Authority evenly distributed</td>
<td>Decision.6 &amp; Decision.8 &amp; Decisions.9</td>
</tr>
<tr>
<td></td>
<td>Trust in team</td>
<td>Decision.3</td>
</tr>
<tr>
<td></td>
<td>Efficient decision-making (inverted)</td>
<td>Decisions.3</td>
</tr>
<tr>
<td></td>
<td>Well justified decisions</td>
<td>Decision.2 &amp; Decision.7</td>
</tr>
</tbody>
</table>