# AGILE SCALING @ Topicus

Scaling Scrum with help of Agile Scaling frameworks at Topicus Finance

Author:
M.M. van Leeuwen

topicus

UNIVERSITEIT TWENTE.

# Agile Scaling @ TOPICUS

*Scaling Scrum with help of Agile Scaling frameworks at Topicus Finance*

Master Thesis
Enschede, 25th of August 2015

## Author

***Monique van Leeuwen BSc***

| | |
|---|---|
| Study Programme | Master of Science in Business Information Technology |
| Track | IT & Management |
| Faculty | Electrical Engineering, Mathematics and Computer Science |
| Student Number | 1006754 |
| E-mail | mmvanleeuwen@outlook.com |

## Graduation Committee

***Dr. Klaas Sikkel***

| | |
|---|---|
| Faculty | Electrical Engineering, Mathematics and Computer Science |
| Department | Services, Cybersecurity and Safety |
| E-mail | k.sikkel@utwente.nl |

***Dr. Ir. Ton Spil***

| | |
|---|---|
| Faculty | Behavioural, Management and Social Sciences |
| Department | Industrial Engineering and Business Information Systems |
| E-mail | a.a.m.spil@utwente.nl |

***Yoni Meijberg MSc***

| | |
|---|---|
| Department | Topicus Finance |
| Function | Change Manager |
| E-mail | yoni.meijberg@topicus.nl |

## Preface

Dear reader,

There it is! My master thesis which symbolises the end of an era as a student. By submitting this thesis my student career comes to an end. Luckily, this does not mean that the learning will stop.

This thesis is concerning Agile Scaling at Topicus Finance. Agility is nice to be able to cope with the changing environment. Scrum is an example of how to organise Agile development. Theoretically Scrum is designed for one team and one product, but in real life this is usually not the case. At Topicus there are multiple Scrum teams working for different customers, working on one product. The abundance of Agile Scaling frameworks show that there is a need to be able to expand the Agility to a wider level in the organisation.

Although, personally I did not use to be so Agile, I preferred sticking to predefined plans. This internship has taught me to sometimes *"Let it Go"*; quoting the famous, or should I say notorious, Frozen Disney song. I have experienced that changing processes and people on paper is easy but in practice I experienced some resistance. People wanted change within the organisation, however when they were asked to change their way of working, it was not always easy. This perfectly fits with the following quote from Tolstoy:

> *"everybody thinks of changing humanity, and nobody thinks of changing himself" (Tolstoy, 1900)*

I enjoyed my internship at Topicus a lot. Everybody was willing to share information and help me to get the best results out of this research. I really appreciate that decisions could be made fast. This is also what I value from the university, help is always nearby and quick. Additionally I was happy that I always had the support of the Management Team to execute my research and experiments. Also I really liked that the people of Topicus act out-of-the-box and come up with daring solutions. Always trying to be different from the crowd.

I would like thank all the PRODUCT D teams for hosting me in their office and for their collaboration. Especially Linda and Frits for doing the Travellers experiment. Also, a thank you to the Scrum Masters and Product Owners for their interview- and meeting time for decision making and experiment planning. Thanks to the Management Team for supporting me in my research and experiments. Last but not least I want to thank my supervisors Yoni, Klaas and Ton, for providing me with constructive feedback, and pushing me for the extra mile.

Unlike Elsa from the film Frozen, the cold does bother me. So I will be travelling across the globe looking for the sun before I will start my next challenge at…..?

Hope you enjoy reading this report. If there are any questions or remarks please feel free to contact me.

Kind regards,

Monique van Leeuwen

## Management Summary

**Context -** At department E of Topicus Finance where the product PRODUCT D is developed, there are several challenges regarding working with multiple Scrum teams, such as knowledge sharing and task planning. Next to the fact that there are multiple teams working on one product, there are also three different customers, Customer X, Customer Y and Topicus itself (Pakket). Topicus posed the question how to deal with this challenge and still remain Agile.

**Research goal -** Propose, test and partly implement an Agile Scaling framework to improve the functioning of the Scrum teams working on PRODUCT D version 3

**Conclusions -** Of the three researched frameworks, there is no single Agile Scaling framework which offers a complete solution for all the challenges faced by Topicus. As part of this research we did a case study at Company B Department C, where we observed that not one entire framework had been implemented, but several different practices are used to fit their specific context. Based on the two examples of Topicus and Company B it is likely that there is not one single answer to the Agile Scaling challenges which companies face. Therefore in the future companies should clearly investigate their Agile Scaling challenges, and select the practices from the frameworks which best fit their organisation.

Furthermore, the adoption of Agile Scaling frameworks requires changes in the way of working of the employees as well as in the structure of the organisation. Although the employees acknowledged that change is needed and they were involved in the decision making, the actual change was challenging. So knowledge from the change management field is very useful for the adoption of Agile Scaling frameworks. While, there were some difficulties in implementing the Travellers[1]. However at the end of the experiment the team members and especially the Product Owner were enthusiastic about the Travellers method and its results.

Additionally, the chosen Large Scale Scrum (LeSS) practice, Travellers, could not be implemented in the exact way as described by the literature. The idea of exchanging teams was kept, but the shape of the Travellers concept had been altered to fit the context of Topicus.

Next to the implied effect of knowledge sharing, the Travellers experiment also lead to mutual respect and understanding for the differences between the internal and external customer teams. Also, this experiment lowered barriers between the two teams, and team members indicated that it is more likely to approach each other in case help is needed. Lastly, the team member of Pakket who had joined the Customer Y team, noticed personal growth by stepping out of her comfort zone.

**Recommendations -** The recommendations are split in recommendations for now, short-term, long-term and recommendation for further research.

**Now -** *Chickens*[2] - Topicus should repeat the Chicken experiment but with more structure and guidance to see whether this will result in enhanced knowledge sharing. This can be achieved by the creation of an Environment Map and a planning of which stand-ups to visit in advance.

*Travellers* - The Travellers should be continued in the format used with the experiment. So briefly it can be stated that the Travellers experiment needs to be executed in the following way.

[1] Travellers: LeSS practice where team members are exchanged between teams for one or multiple Sprints.

[2] Chickens: LeSS practice where team members join stand-up meetings of other teams.

- Exchange team members who voluntarily choose to join
- Plan one to two Sprints in advance
- Choose a topic which is easy to get started on (if possible)
- Let somebody from team A join team B for one Sprint, afterwards change
    - The people who switch places preferably have the same function (i.e. developer)
- After the exchange plan an evaluation with both teams to share experiences on wider level

**Short-term -** *Shared Space (LeSS)* - Start with the increase of shared space, since this is also mentioned by the team members as a necessity. This means that the Customer Y teams have to move at least to the same side of the building as where the other PRODUCT D teams are located.

*Joint Sprint Review Bazaar (LeSS)* - This can be done in addition to individual Sprint Reviews. During this bazaar each team requires their own space where they can explain and demonstrate what they have been doing the previous Sprint.

*Open Space (LeSS)* - This is a meeting where team members have the opportunity to share their concerns with people of other teams. At the start of the meeting people share what they want to discuss, and these topics are scheduled in parallel meeting sessions of 20 minutes.

*Same start/stop day* - In order to make joint meetings possible, all the teams need to have their start/stop day on the same day.

**Long-term -** *Uniformity/standardisation of teams* - More uniformity and standardisation across teams is needed in order to be more flexible with the planning of work. This means that there should be one single Product Backlog, one Definition of Done and one Definition of Ready.

*Joint Sprint Planning* - When there is one Product Backlog and teams have the same start/stop day, it is possible to start using the Sprint Planning as done by Company B.

*SAFe®* - As indicated by the Product Owners, Scrum Masters and team members there is a demand for a translation of the strategy and vision towards a more operational level, so decisions regarding the development of the product can be made so it supports the strategy and vision of the company. LeSS focuses on practical coordination tools on team level. SAFe® on the other hand takes the entire organisation in to account. SAFe® describes Agile Scaling on team, portfolio and program level. Therefore LeSS and SAFe® complement each other. SAFe® uses the strategy, vision and roadmap of the company as input for the Product Backlog.

**Future research -** *Scientific Proof of Frameworks* - During this research I noticed scientific proof of the success of the different Agile Scaling frameworks is lacking. For the majority of these frameworks cases are available which can be used for further research to get scientific proof of these frameworks.

*Characteristics Frameworks* - Furthermore, additional research should be done on what framework is best in which kind of situations. So, when organisations have defined their Agile Scaling challenges, it is easier to select the most suitable framework.

## Table of Contents

# 1 Introduction

*This chapter gives an introduction to this research. In Section 1.1 the rationale behind this research is given. In Section 1.2 some background information is given regarding the context of this research. The research goal and the sub-research questions are discussed in Section 1.3. The next section (1.4) of this chapter motivates how the sub-research questions are answered and how the research goal will be achieved. The last section (1.5) provides an overview of the structure of this research report.*

## 1.1 Relevance

In the early 1990s the strategic management and manufacturing literature introduced the concept of Agility (Salmela, Tapanainen, Baiyere, Hallanoro, & Galliers, 2015). Goldman et al. (1995) stated that organisations need to be flexible in order to respond to new opportunities to stay ahead of the competitors. Agile methods are widely used in software development. One of these Agile software development methods is Scrum. Although in the early 1990s Agility was introduced in the scientific literature, the Harvard Business Review already published an article of Takeuchi and Nonaka about the predecessor of Scrum in 1986.

When reading about Scrum a common question is whether Scrum can be applied outside the field of Software Engineering (Solingen & Rustenburg, 2012) . Moreover it is mentioned that Scrum is developed for software development (Sutherland, 2001) (Kester, 2013). However when going back to scientific roots of Scrum (AgiliX, 2013), the paper of Takeuchi and Nonaka (1986) describe how Honda and Canon used a production/development method with overlapping phases instead of the traditional sequential approach. This overlapping method was named Rugby, because "the ball gets passed within the teams as it moves as a unit up the field" (Takeuchi & Nonaka, 1986). This is exactly what happens within Scrum teams.

So Scrum finds its origin in the production industry, later it was adopted within the development of IT. Scrum is described to be used by one team working on one product. As this is not always the case and companies adopt multiple Scrum teams working on one product, problems may arise. The abundance of Agile Scaling frameworks and consultancy firms show that Topicus is not the only company facing this scaling challenge. The main challenge for Topicus is while scaling, to still remain Agile.

In case a company expands this can bring several advantages such as being able to serve more customers as well as financial benefits because of economies of scale. Possible disadvantages of scaling are that the communication lines are longer and loss of flexibility.

However some companies want to scale but remain flexible. This is where Agile Scaling pops in. Agile Scaling can happen at different levels. On a more operational level where the number of teams are increased. But also higher in the organisation where the entire organisation is Agile. The independent research office TOTTA Research together with Xebia conducted a research study about Agile at organisations in the Netherlands. The results are given in Figure 1. This research shows that 47% of the 704 respondents adopted Agile on team level. 9% mentions that the entire organisation is designed around Agile. (Xebia, 2014) This shows there are several levels of adopting Agile in an organisation, and that there are also possibilities for Agile Scaling. Furthermore, the research of Kettunen and Laanti (2008) concluded that new product development companies need an holistic system wide view to be able to take benefit from Agile software development.

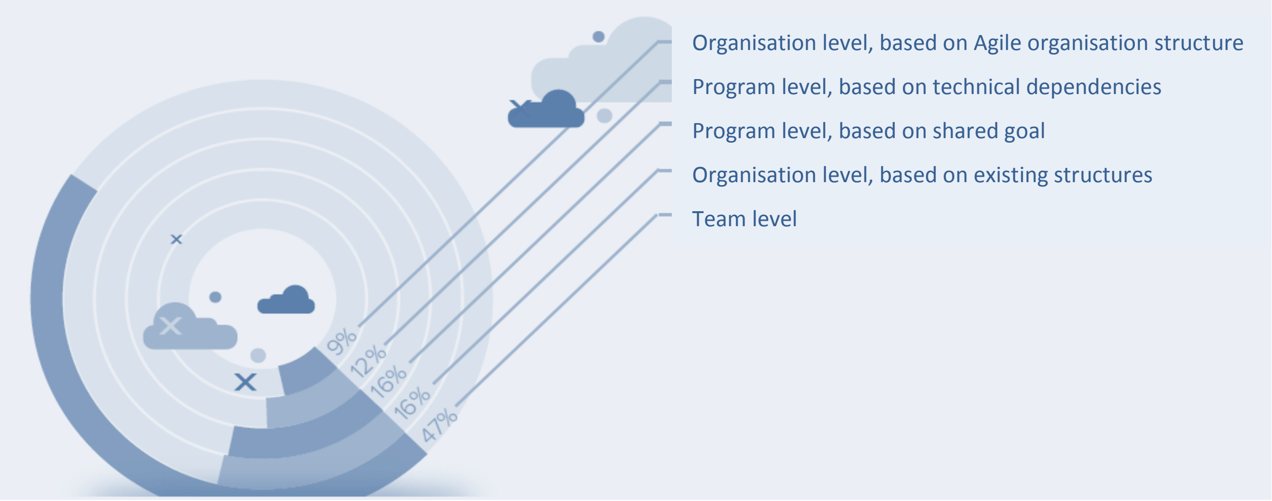


**Figure 1: Implementation level of Agile (Xebia, 2014)**

This research will try to find a solution on how to solve the scaling challenges in an Agile manner on team level. As every company is unique with their own culture and habits, this thesis is written with some aspects of an action research (McNiff & Whitehead, 2009) and the problem solving cycle (Berends, van Aken, & van der Bij, 2012) to be able to get the best results possible for Topicus. This research will help Topicus getting insight in the different Agile Scaling frameworks and for what purposes these frameworks might be beneficial.

The scientific contribution of this research is concerning the comparison of Agile Scaling frameworks. During my preliminary desk research, a limited amount of papers regarding Agile Scaling was found and no papers were found on the comparison of Agile Scaling frameworks. This research will contribute to the scientific literature by providing a comparison of different Agile Scaling frameworks.

## 1.2 Context

In this section background information is given to create a basic understanding of the environment in which this research takes place. First of all a short description is given of the company Topicus, their divisions and products related to this research. In Section 1.2.2 the working philosophy Agile is described. Section 1.2.3 gives the theoretical background of the methodology used at Topicus, Scrum. The last part 1.2.4 discusses the basics about Agile Scaling for the final solution of this research.

### 1.2.1 Topicus

Topicus is an ICT organisation founded in 2001. It develops service concepts in the finance, education, government and health care sector. Topicus focuses on chain integration and Software as a Service (SaaS) solutions. Topicus has a strong focus on the wellbeing of the employees, the talents working at Topicus are triggered to continuously improve themselves. For example by events such as TopiConf, to share the knowledge among the different departments. (Topicus, 2015)

The mission of Topicus Finance, its biggest division, is to improve the financial chain with its software solutions. Topicus Finance believes that the most important chain partners should cooperate in order to achieve the best financial solution for the customer. This is of interest for all stakeholders: customer, intermediary, accountant and lender.

Within Finance the products are sub-divided under different domains; mortgages, business lending, wealth accumulation, e-commerce and health insurances. (Topicus, 2015)

This assignment is concerning one of these domains. Within this domain five products are offered. One of these products is called PRODUCT D. This product is a mid and back office system used in the finance industry. The product offers a high degree of Straight Through Processing within all finance processes. (Topicus, 2015)

At the moment Topicus Finance is using the Scrum methodology for the development of their product PRODUCT D. This product exists of three versions. For this research my focus is on the (current) seven Scrum teams working on the third version of PRODUCT D. Since product version three is built by seven different teams it is important for a team to know what the other teams are doing. For these seven teams there are three Product Owners available who have product meetings to discuss the progress. Because Topicus Finance is moving from a project oriented approach to a product oriented approach and the number of teams working on one product will grow in the future, good portfolio management becomes more important. With better portfolio management the reliability will be improved and the product will sooner valuable.

### 1.2.2 Agile

According to the Oxford dictionary the word Agile is defined as "able to move quickly and easily" (Oxford Dictionary, 2015). Within the software development Agile Methods are described as: iterative, incremental, self-organising and emergent (Lindvall, et al., 2002). Within software development Agile methods have become very popular. Being Agile helps the software developer to meet the changing demand of the customer. Which is very important in an online environment which is likely to evolve quickly. The principles of Agile software development are documented in the Agile Software Development Manifesto. This manifesto was written in 2001 by seventeen well known software process methodologists (Chow & Cao, 2008). The philosophy of Agile software development in the manifesto is defined as the following:

*"We are uncovering better ways of developing software by doing it and helping others do it.*
*Through this work we have come to value:*

| | | |
|---|---|---|
| *Individuals and interaction* | *over* | *processes and tools* |
| *Working software* | *over* | *comprehensive documentation* |
| *Customer collaboration* | *over* | *contract negotiation* |
| *Responding to change* | *over* | *following plan* |

*That is, while there is value in the items on the right we value items on the left more." (Beck, et al., 2001)*

Next to the definition just given, there are twelve principles which have to be followed in order to be a perfect Agile organisation. There are several software development methodologies which work according to the Agile manifesto. According to Chow & Cao (2008) the list exists of Extreme Programming, Scrum, Feature-Driven Development, Dynamic System Development Method, Adaptive Software Development, Crystal and Lean Software Development.

### 1.2.3 Scrum

Scrum is the Agile approach used at Topicus Finance. Scrum is a framework used for organising and managing work (Rubin, 2013) it is also used for the development of innovative products and services (Rubin, 2013). Scrum should be seen as a framework to guide a team through the development of software. Scrum provides a foundation within a team to create a common set of values, principles and practices (Rubin, 2013). Scrum exits of a set of roles, activities, artefacts and rules. How these elements interact within the Scrum process is shown in Figure 2.

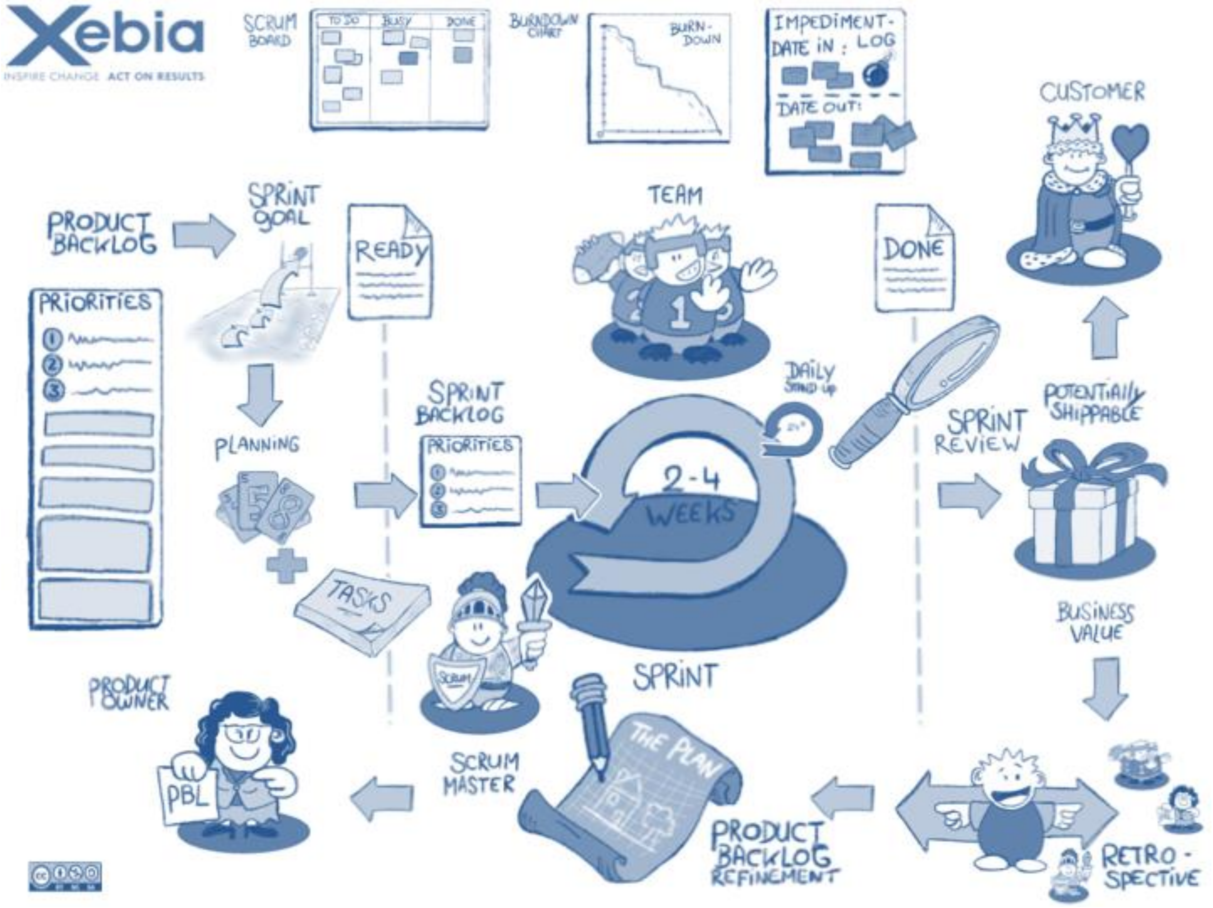


**Figure 2: Scrum Process (Burm, 2013)**

The Product Owner is responsible for the product and communicates the desires of the customer with the team. The desires of the customer are listed in features in the Product Backlog by priority. During the Sprint planning meeting, planning poker is played. With planning poker every individual of the team gives an estimate of the time needed to finish a certain feature by showing a card with a number. In some cases not all team members have similar ideas about the time needed, so the team members explain to each other why they think that the feature requires a certain amount of time. With the new information, people again show their new estimates. This is repeated until the team reaches a consensus. All features are discussed according to the planning poker methodology.

Next a Sprint backlog is created. The Sprint backlog exists of features listed at the top of the Product Backlog which can be finished within one Sprint. These features should meet the Definition of Ready, which means there is enough information available to be able to finish the feature. The features are then split-up into tasks which are attached to the Scrum board. During the Sprint each team member picks a task to start working on. During a Sprint, every day is started with a daily stand-up meeting. Where the team discusses in 15 minutes what they have been doing the previous day, what they will be doing the rest of the day and what is stopping them from being able to work, the so called impediments. The Scrum Master is responsible for solving these impediments. Once the features meet the Definition of Done at the end of the Sprint there should be a potentially shippable product. The Sprint is finished with a Sprint review in which a demo is given of the product. The Sprint review is usually attended by the Scrum team, Product Owner, Scrum Master, management, customers and developers from other projects (Mountain Goat Software, 2015). During the retrospective the team discusses how they can improve their own work processes. This entire cycle is repeated until the product is completely finished. (Franken, 2013) (Solingen & Rustenburg, 2012)

The philosophy of Scrum is that every Sprint, a completely finished part of the total package is finished. Because the customer gets to see a working part of the product more often, more feedback is generated. Due to Scrum the developing team is more flexible in coping with changes from the customer because the Product Backlog is not something which is fixed in advance. The original Scrum methodology is meant for one team working on one product (Franken, 2013).

### 1.2.4 Agile Scaling

Scrum focuses on the optimisation of work within one team and on one product. However it is not always possible to work on one product with one team. In order to keep the Agile and Scrum spirit, a new framework or an extension of Scrum is needed to be able to manage multiple Scrum teams working on one product. The just described situation is also the case at Topicus Finance, this is confirmed by the Product Owners, Managers and Scrum Masters. Agile Scaling is an overall term used for the set of frameworks and methodologies available which enable the scaling of the Agile methodologies mentioned in Section 1.2.2.

## 1.3 Research goal

As the Scrum methodology is used within the teams the plan is to find an Agile Scaling methodology to manage the development of the product PRODUCT D. The objective is to test and (partly) implement one methodology within the teams. This can either be an existing methodology, a combination of more methodologies or an adjusted version of a current methodology. Depending on what fits best for the teams. In the end an advice is given on the usage of the methodology.

The research goal is formulated as follows:

*Propose, test and partly implement an Agile Scaling framework to improve the functioning of the Scrum teams working on PRODUCT D version 3*

In order to achieve the above mentioned goal five research questions have been formulated.

1. Which Agile Scaling problems are experienced within Topicus Finance?
2. What are the requirements for Topicus Finance for an Agile Scaling framework, based on the problems experienced?
3. Which different Agile Scaling methods exist and how are the different characteristics of the models interrelated.
4. Which (part) of these models or combination of these models can be applied to the current situation in such a way that it will improve the current process?
5. Is the proposed model valid for Topicus Finance?

## 1.4 Approach

For this research a Design Science Methodology is most appropriate to solve the problem of Topicus. There are several Design Science Methodologies such as the ones written by Hevner (2004), Wieringa (2014) and van Aken et al. (2012). The one from van Aken et al. was chosen because it focuses on problem solving in a business context whereas the other two frameworks primarily address engineering. In order to answer the research questions and structure this report the problem solving cycle is used (Berends, van Aken, & van der Bij, 2012). This cycle is shown in Figure 3. This cycle is one of the two process structures from the book Problem Solving in Organizations (Berends, van Aken, & van der Bij, 2012). The Problem Solving Cycle helps by giving guidance solving problems in organisational context. By structuring the problem mess, a problem definition is created. This problem definition may change during the project based on new insights. Also a generic literature review should be done at this stage. The next stage is to analyse the problem and diagnose what causes the problem. When possible causes are known a solution is designed. In this phase is also determined how the solution is going to be executed in the intervention phase. During the intervention step the solution is implemented. At the learning and evaluation step the results of the intervention are discussed which may result in a new problem definition.

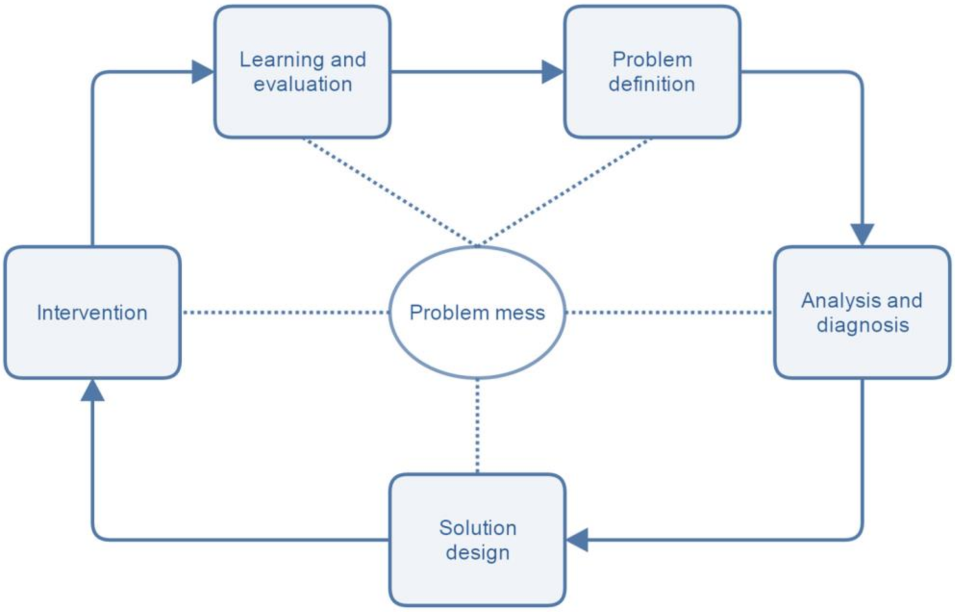


**Figure 3: Problem Solving Cycle (Berends, van Aken, & van der Bij, 2012)**

The research goal is met by answering the five research questions. The answer for the first research question is obtained by interviewing the Scrum Masters, Product Owners and the Management Team which work on PRODUCT D v3. These interviews are semi-structured to derive as much information as possible regarding Agile Scaling and product management. For the interview questions see Appendix A. For efficiency purposes, not all of the Scum Masters and Product Owners are interviewed. The people are selected based on the availability of the Scrum Masters and Product Owners. In case the interpretation of how PRODUCT D v3 is currently managed varies between the interviewees, I can decide to interview the other Scrum Masters and Product Owners to get a complete story. With the answers to these interview questions I can see how the different people interpret the current way of working. In case other problems arise which are not within the scope of Agile Scaling these are reported to my supervisor at Topicus, so he can try to solve these issues. Additionally Scrum meetings and product meetings will be attended to observe what is discussed and what issues arise.

The second question is answered with help of the previous research question. Based on the answers a prioritised list of desires is created. During a meeting with the Scrum Masters, Product Owners and Management Team, this list is discussed to get approval on the correctness.

Using a literature study the third question is answered. For this part both scientific and non-scientific literature is studied, since part of the information concerning Agile Scaling is found at (consultancy) organisations, which due to best practices developed an Agile Scaling framework.

The list of frameworks will be minimised to three frameworks based on a set of requirements which will be defined later on in the research. These three frameworks will then be thoroughly analysed and compared.

Based on the comparative analysis made in the third research question an analysis will be done to decide which Agile Scaling framework fits best for Topicus. In case multiple models have different aspects which match the requirements of Topicus, an integrated model is designed which combines best of both worlds. When none of the existing frameworks match the requirements of Topicus a new framework is designed. During the design feedback is given by the Scrum Masters, Product Owners and Management Team in order to prevent a misfit. The final decision for a specific framework is made together with the Product Owners, Scrum Masters and Management team to create support for testing.

The validity of the proposed framework is tested with an experiment. How the validity is measured, depends on the framework chosen. The framework is tested within multiple Scrum teams during two or three Sprints depending on the time available for this research and the teams. The number of teams collaborating with testing the framework depends on several factors such as work load, willingness to cooperate, trust in the proposed framework. To keep everybody informed about the progress of this research several meetings are joined to give an update.

## 1.5 Chapter Overview

This section provides an overview of how the problem solving cycle and research questions are linked to the different chapters. This overview is given schematically in Figure 4.

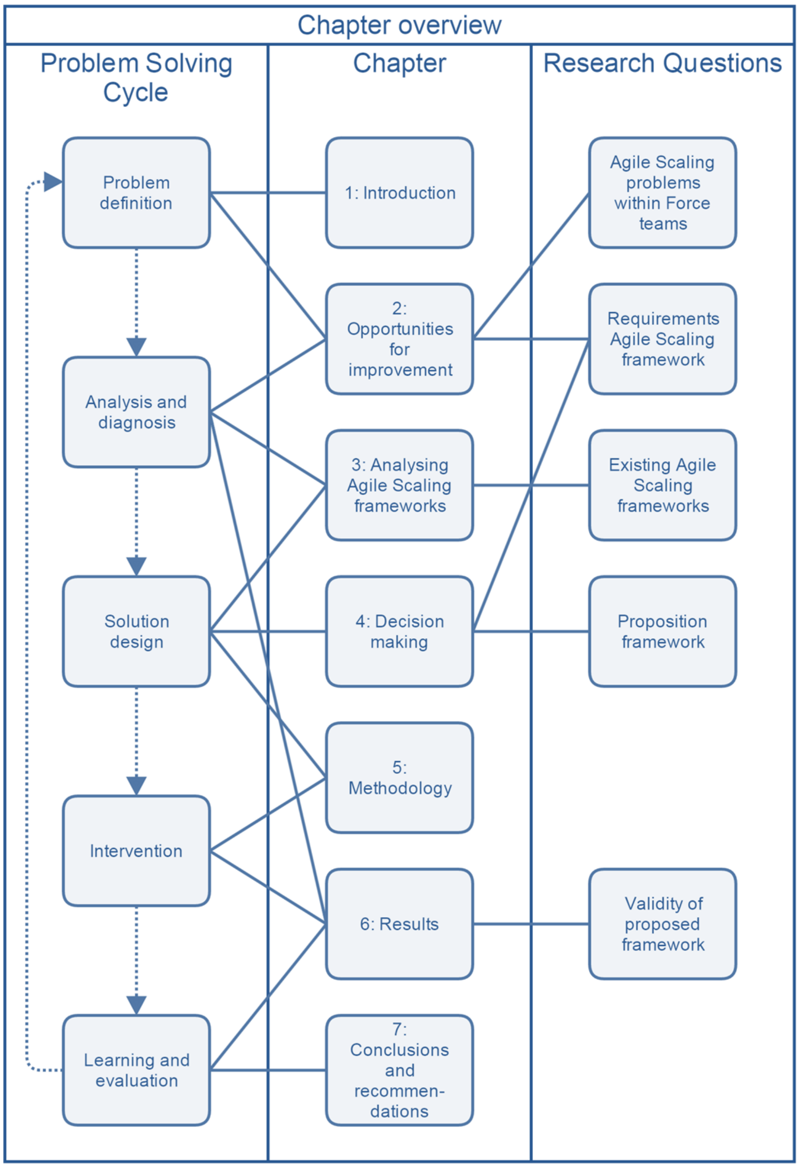

Chapter overview
Problem Solving Cycle
Chapter
Research Questions
Problem definition
Analysis and diagnosis
Solution design
Intervention
Learning and evaluation
1: Introduction
2: Opportunities for improvement
3: Analysing Agile Scaling frameworks
4: Decision making
5: Methodology
6: Results
7: Conclusions and recommen-dations
Agile Scaling problems within Force teams
Requirements Agile Scaling framework
Existing Agile Scaling frameworks
Proposition framework
Validity of proposed framework

**Figure 4: Chapter overview**

This first chapter is to provide the reader with an introduction to the context of the research matter. In this chapter a basic understanding of the problem is given together with the research goal and research questions.

In the second chapter a more thorough analysis of the current situation is given to be able to set more specific requirements on what the Agile Scaling framework should solve.

The third chapter provides an analysis of the different Agile Scaling frameworks based on the requirements set in the first chapter. After the downsizing of the large list of frameworks, three framework are analysed in more detail.

Chapter 4 combines the analysis from the previous chapters to propose a framework which will contribute to solving the scaling challenges faced by Topicus.

The methodology on how to do the intervention is described in Chapter 5. In this chapter is given how the experiment is set up and how the results will be collected.

In Chapter 6 the results of the experiment are given. The results provided will be split up in the pre-experiment and post-experiment. Additionally the results are evaluated. Moreover the validity of this research is discussed.

The seventh chapter gives a conclusion of this research. In this conclusion the answers to all research questions are briefly discussed. Additionally the main conclusions of this research will be given. At last the recommendations for Topicus and science are discussed.

# 2 Opportunities for Improvement

*Based on the interviews with the Scrum Masters, Product Owners and Product Owner Architects a list with opportunities for improvement is created regarding the scaling of Scrum. These opportunities are prioritised and discussed in this chapter. In Section 2.1 the different opportunities and the rationale for these opportunities is given based on the view of the Scrum Masters and Product Owners. In the second section backup for the rationale is given from the scientific literature.*

## 2.1 Opportunities

Because a lot of information is needed to be able to spot the opportunities, four out of nine Scrum Masters, four out of five Product Owners, two out of four architects and the Chief Product Owner were interviewed. The interviews resulted in a list of desires for coping with the scaling issues. The interviews revealed that the scaling challenges were not specifically related to Scrum. When the waterfall method was used in the past there already were problems regarding scaling, however at that time it was of a different scope because of the size of PRODUCT D and the smaller number of customers. As Topicus is currently using Scrum and wants to stick to the Agile philosophy for the future, it is decided to research the different Agile Scaling frameworks in the next chapter. The list of desires which is given underneath and their relationships are shown in Figure 5.

- The product documentation should be better.
- The product documentation should be easy to find
- One Product Backlog (no differentiation between customer and package teams)
- Functional/component teams with own PO
- PO for each customer
- Start stop day of all teams in the same week.
- No further meetings
- Vision and strategy of Product D should be clear
- Know where the specific contribution of the teams fit within the complete Product D picture
- Easy to find what is done in the past by whom
- Know what should be done for coming 2 months to be able to prepare better stories
- Usage of measures and warning signals
- Teams should be located closer to each other

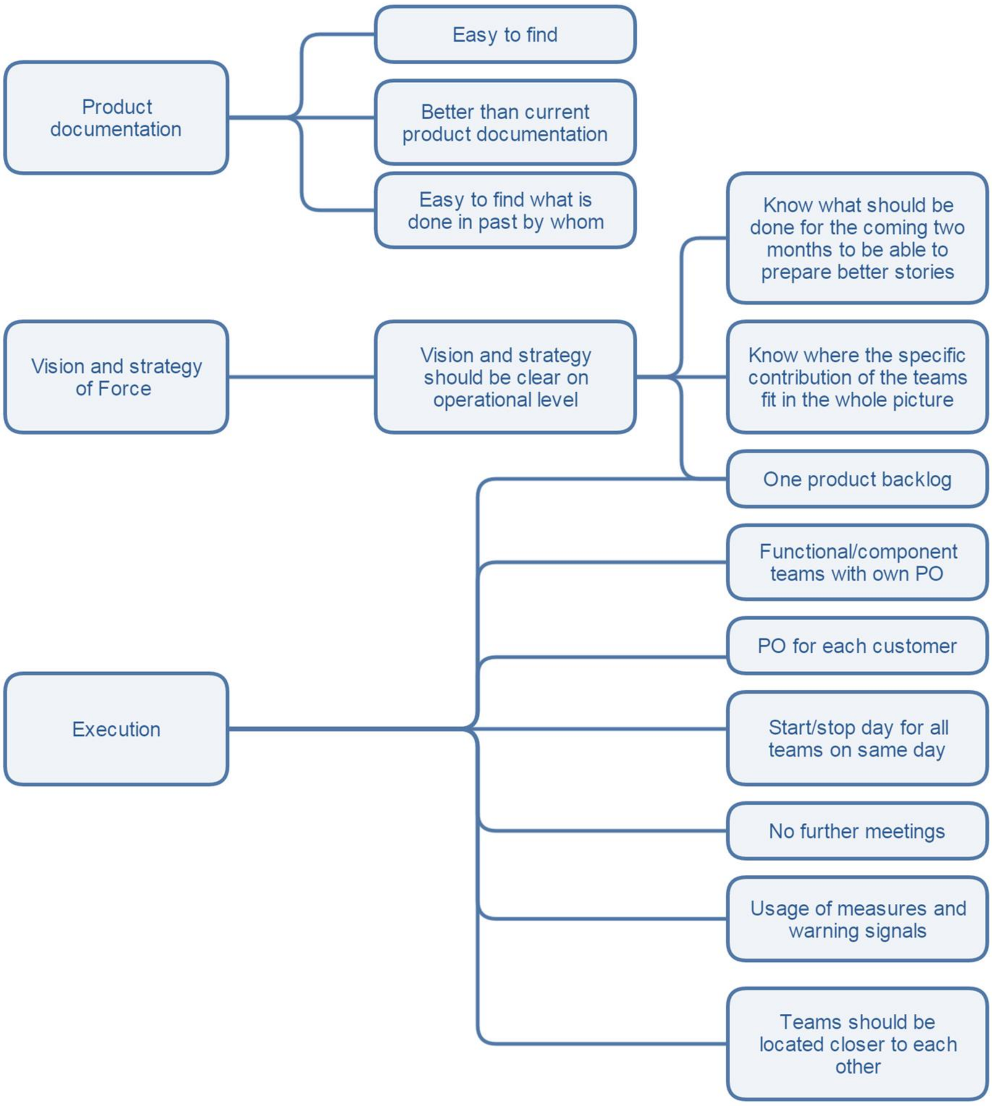

Product documentation
Easy to find
Better than current product documentation
Easy to find what is done in past by whom
Vision and strategy of Force
Vision and strategy should be clear on operational level
Know what should be done for the coming two months to be able to prepare better stories
Know where the specific contribution of the teams fit in the whole picture
One product backlog
Execution
Functional/component teams with own PO
PO for each customer
Start/stop day for all teams on same day
No further meetings
Usage of measures and warning signals
Teams should be located closer to each other

**Figure 5: Cluster desires**

The list from above, was discussed during the Scrum Master and Product Owner meetings to find the rationale behind the desires. During these meetings additional desires were added. Then the desires were clustered again, but this time by the attendees of the meeting, so the ranking could be done per domain. The clustering was initiated by the Chief Product Owner. The ranking is done by choosing a common top three of the domains created. The top three of the Product Owners (PO) and Scrum Masters (SM) is shown in Table 1. Not everything is related to scaling. The desire for improved knowledge sharing is a consequence of scaling, since working with multiple teams makes is harder to keep the other teams up to date. The division of tasks not directly linked to scaling, however when tasks are easier and better planned upfront this can also contribute to more effective knowledge sharing. The translation of Topicus' strategy to operational level does not have a match with scaling. The problem of an unclear operational strategy is not related to a growing number of teams. Prioritisation before integration is a scaling problem, because with only one team is does not require a planning on who is allowed to integrate at what moment in time.

| PO | SM | |
|---|---|---|
| 1 | 2 | Improved knowledge sharing |
| | | Prevent single point of failure, technical and functional |
| | | Information should be documented, not stored in people's heads |
| | | Documentation should be stored in a central location |
| | | Traceability of who worked on what (who is experienced in…) |
| | | A method to share important information with everybody |
| | | Knowledge sharing should be with minimum effort |
| 2 | 2 | Better division of work up front |
| | | Prioritising backlog should be easier |
| - | 1 | Overall strategy/vision should be translated to operational level (what does it mean for Architecture, Teams…) |
| 3 | - | Priority rules before integration |
| - | - | Close contact with customer should be maintained |
| - | - | Increased flexibility of the teams regarding availability |
| - | - | Rationale for certain decisions should be documented |

**Table 1: Opportunities for Improvement**

The rationale behind the knowledge domain, is that without good knowledge management productivity decreases. Knowledge sharing should also prevent functional mismatches. Additionally due to knowledge sharing re-inventing the wheel should not be the case anymore. When the work division is improved and known beforehand, teams already know their interdependencies, this way the teams can seek for collaboration when necessary and gather information regarding that certain topic in advance. Having a clear vision and strategy which is translated to operational level, provides support to the teams regarding decision making, because the teams know what the vision is for the product. The priority rules before integration are necessary to know what functionality has priority over another functionality when it comes to integration since there is one codebase available and not all functionalities can be integrated at once.

## 2.2 Scientific rationale

Besides the rationale provided by the Scrum Masters and Product Owners, within the scientific literature additional validation was found. When looking in Scopus and Web of Science the term "knowledge management sharing Agile organizations" was used. I realise this search term is specific and therefore possibly excludes relevant articles. Searching with this term provided sufficient information to support the rationale and relevance of knowledge management. Knowledge Management is crucial for the productivity in Agile software development as well as for keeping up with the competitors (Richardson, Casey, O'Riordan, Meehan, & Mistrík, 2009) (Razzak & Ahmed, 2014). According to Rus and Lindvall (2002) on the business side knowledge management is important to decrease cost and increase quality, as well as to make better decisions. The knowledge itself also has needs regarding, keeping up-to-date with new technologies, the access to domain knowledge, knowledge sharing concerning local policies and practices, keeping track of who knows what, at last collaboration and sharing of knowledge (Rus & Lindvall, 2002). Knowledge sharing between teams is useful to prevent conflicts between the different dependent teams (Babinet & Ramanathan, 2008).

Confirmation in the literature for the relevance of prioritisation of the backlog was found by searching in Scopus for "task planning software development". Several other search terms have been used as well (task division; labour division, job planning) but these terms provided papers concerning work division across men and women at home. The search term "Task planning software development" resulted in 1120 articles. By limiting the results to the years 2010-2015 and the subject area of business, management and accounting, 21 articles were left. Useful articles were then selected by title and abstract. Shao et al. (2014) mentioned in their article that the organisation and planning of knowledge workers for iterative software development is critical for the success of the software project. At Ericsson AB the technical dependencies between the Agile teams cause problems with the planning and prioritisation. The recommendation is done to improve knowledge sharing and change the mind-set in order to overcome dependency issues like planning and prioritisation. (Sekitoleko, et al., 2014) Backward citation of the paper Technical Dependency Challenges in Large-Scale Agile Software Development (Sekitoleko, et al., 2014) resulted in more references.  The paper of Babinet and Ramanathan (2008) also acknowledge that the change in priorities in the team backlog can lead to conflicts between teams in case there are dependencies among the teams. So prioritising the backlog in advance and working with only one backlog is necessary in order to prevent such conflicts.

In the literature, importance of the translation of the strategy and vision to action is confirmed. By searching for "translat* strategy vision to operational" and "translat* strategy to action" in Scopus, the search was limited to articles, conference papers as well as computer science, business management and accounting, and economics, econometrics and finance but also limited to publication between the years 2010 and 2015. This resulted in 70 articles. Based on the search results and backward citation of the articles, Kaplan and Norton on the Balanced Scorecard were cited abundantly. Based on this large amount of papers on this topic, I assumed there is a demand for operationalising the corporate strategy. Additionally searching for "why strategy to operations important" resulted in 173 articles, afters narrowing down the search on business management and accounting 34 articles were left. Boyer and McDermott (1999) show the importance of strategic consensus in their paper. They mention that the consensus creates effectiveness since it guides employees with daily decision making on operational level. Forward citation lead to an article on aligning strategic priorities by Joshi et al. (2003). Their research shows that aligning the business strategy to an operation strategy indirectly positively influences the performance of the company. Moreover, Weelwright (1984), found through backward citation of Joshi et al (2003) and Boyer and Mc Dermott (1999), emphasises that a shared philosophy and culture are important for the execution of the strategy, since these are basic elements which guide decision making in a company also at operational level.

*To summarise, this chapter defined the opportunities for improvement given by the Product Owners and Scrum Masters. The top three is composed of knowledge sharing, task planning and strategy on operational level. For the top three additional motivation for their importance was given based on the scientific literature. These opportunities for improvement are required to be solved by (a part of) an Agile Scaling framework. The next chapter will give an outline of the different frameworks. In this chapter the first research question regarding the Agile Scaling problems within the PRODUCT D teams is answered. Moreover, the second research question is partly answered, considering the requirements for the Agile Scaling frameworks. Chapter 4 will give some more information on the requirements of the Agile Scaling framework.*

# 3 Analysing Agile Scaling Frameworks

*This chapter will elaborate on the different Agile Scaling frameworks. The first section of this chapter shows the search and the deduction of the extensive list of frameworks. Thereafter in Sections 3.2 (LeSS), 3.2(SAFe®) and 3.3(DAD), the top three frameworks are described. A comparative analysis is made to see what similarities the different frameworks have, this is shown in the fourth section. The last section 3.5 shows an example of how Agile Scaling is done at Company B.*

## 3.1 Creating short list

When using Google Scholar and the library database of the University of Twente the search terms "Agile Scaling" and "Scaling Scrum" provided me with a limited amount of material. Searching for the same terms on the Google search engine, resulted in a bigger list of different Agile Scaling frameworks. (Dolman & Spearman, 2014)

- Agile Software Solution Framework (ASSF) (Qumer & Henderson-Sellers, 2007)
- Descaling as alternative to Scaling Agile (Lewitz, 2014)
- Disciplined Agile Delivery (DAD) (Lines & Ambler, Disciplined Agile Delivery, 2015) (Ambler & Lines, 2010) (Ambler, 2009)
- Dynamic Systems Development Method (DSDM) (DSDM Consortium, 2015)
- Enterprise Scrum (Beedle, 2015)
- Large Scale Scrum (LeSS) (Larman & Vodde, Large-Scale Scrum, 2014)
- MAtriX Of Services (MAXOS) (Singleton, 2014)
- Recipes for Agile Governance (RAGE) (Trapani, 2014)
- Scaled Agile Framework[3]® (SAFe®) (Leffingwell, Scaled Agile Framework 3.0, 2015)
- ScALeD Agile Lean Development (Beck, Gärtner, Mathis, Roock, & Schliep, 2015)
- Scrum at Scale (Sutherland, 2014)
- Scrumban (Toebes, 2014) (Ladas, 2008)
- Scrum of Scrums (Bird & Davies, 2007) (Sutherland, 2001)
- Spotify Model (Kniberg & Ivarson, 2012)
- The Mega Framework (Maranzato, Neubert, & Herculano, 2012)

The different Agile Scaling frameworks are tested on several criteria. The five criteria are as follows:

- Framework should be applicable on Scrum
- Enough (scientific) information on framework should be available for research
- The framework should have sufficient success cases
- Courses about the framework should be available
- The framework should comply with the Agile Manifesto

The frameworks were compared with each other and are scored on a five point scale (--, -, o, +, ++). A "?" means that no information was found. Topicus Finance indicated that they are content with the usage of Scrum and that they would like to stick to this. Therefore the framework should score a + or ++ in order to be taken into consideration. A score of ++ was given in case the information mentioned that the framework is an extension of or applicable to Scrum. Enough (scientific) information should be available in order to be able to do proper research, also once a framework is chosen, sufficient information should be on hand to implement the framework. A framework with most (scientific) information available scored a ++, the framework with barely any information available got a --. Cases can function as an example, to see what should and should not be done when implementing a framework. The framework with the highest amount of detailed explanations of successful cases scored a ++. Frameworks without any cases or negative results scored a --.  Available courses can help the employees to master the framework and to be a good ambassador of the framework within the company. When courses were offered at multiple organisation the framework scores a ++. In

[3] SAFe and Scaled Agile Framework are registered trademarks of Scaled Agile Inc

case no courses could be found the framework scores a --. According to Topicus the Agile philosophy is very important to them. The framework which best complied with the Agile Manifesto received a score of ++. The framework which least matched with the Agile Manifesto scored a --. The results of this comparative analysis are shown in Table 2.

| | Applicable on Scrum | (Scientific) Information Available | Relevant Success cases | Courses available | Agile | Total<br>-- = -2<br>- = -1<br>O = 0<br>+ = 1<br>++ = 2 |
|---|---|---|---|---|---|---|
| Agile Software Solution Framework[4] | ? | - | -- | -- | ? | N/A |
| Descaling as alternative to Scaling Agile[5] | ++ | -- | -- | o | + | -1 |
| Disciplined Agile Delivery[6] | ++ | + | + | + | ++ | 7 |
| Dynamic System Development Method[7] | + | o | - | + | ++ | 3 |
| EMAtriX of Services[8] | o | - | -- | -- | ++ | N/A |
| Enterprise Scrum[9] | ++ | + | o | + | ++ | 6 |
| Large Scale Scrum[10] | ++ | ++ | + | ++ | ++ | 9 |
| Recipes for Agile Governance[11] | ++ | - | -- | + | + | 1 |
| Scaled Agile Framework®[12] | ++ | + | + | ++ | ++ | 8 |
| ScALeD Agile Lean Development[13] | ++ | - | -- | -- | ++ | -1 |
| Scrum at Scale[14] | ++ | + | o | + | + | 5 |
| Scrumban[15] | + | o | - | + | + | 2 |
| Scrum of Scrums[16] | ++ | + | o | - | + | 3 |
| Spotify Model[17] | + | - | o | -- | ++ | 0 |
| The Mega Framework[18] | ++ | - | -- | -- | o | -3 |

**Table 2: Comparative analysis all frameworks**

[4] (Qumer & Henderson-Sellers, 2007)
[5] (Lewitz, De-Scaling Your Organization and Using Temenos, 2014) (Lewitz, 2015)
[6] (Ambler & Lines, 2010)
[7] (DSDM Consortium, 2015)
[8] (Singleton, 2014) (Continious Agile, 2015)
[9] (Beedle, 2015) (Greening, 2010)
[10] (Larman & Vodde, Large-Scale Scrum, 2014) (Paasivaara & Lassenius, 2014) (Heikkilä, Paasivaara, Lassenius, & Engblom, 2013)
[11] (Thompsom, 2013)
[12] (Leffingwell, Scaled Agile Framework 3.0, 2015) (Leffingwell, 2007)
[13] (Beck, Gärtner, Mathis, Roock, & Schliep, 2015)
[14] (Sutherland, 2014) (Scrum Inc, 2014)
[15] (Ladas, 2008) (Toebes, 2014) (Net Objectives, 2013) (Accelebrate, 2013) (Peltier, 2013)
[16] (Bird & Davies, 2007) (Sutherland, 2001) (Barton, 2007) (Cohn, 2007)
[17] (Kniberg & Ivarson, 2012) (Harasymczuk & Kniberg, Spotify Engineering Culture part 1 (Agile Enterprise Transition with Scrum and Kanban, 2014) (Harasymczuk & Kniberg, 2014)
[18] (Maranzato, Neubert, & Herculano, 2012)

Based on this analysis the top three was chosen for further analysis and comparison. The top three is decided based on the total score. Only the top three was chosen because not all frameworks could be compared in the timeframe of this research since the chosen framework also needs to be tested within some of the teams. Since the final scores of the different frameworks were quite close, this could mean that possible useful aspects of other frameworks, such as Enterprise Scrum and Scrum at Scale, were not be taken into consideration. The top three frameworks exist of:

1. Large Scale Scrum (LeSS)
2. Scaled Agile Framework® (SAFe®)
3. Disciplined Agile Delivery (DAD)

## 3.2 Large Scale Scrum

Large Scale Scrum (LeSS) is meant to expand the one-team Scrum. A division can be made between LeSS and LeSS Huge. The standard LeSS is meant for around 8-10 teams working on one product. LeSS Huge can be used for a larger number of Scrum teams up to a few thousand people who work on one product. LeSS works with one Product Backlog, one Definition of Done, one Potentially Shippable Product Increment and one Product Owner. All teams work in the same Sprint. LeSS shows a clear preference for the usage of feature teams. In case there are too many people to manage in one meeting, each team sends one or two representatives. Within a Sprint the Sprint Planning, Sprint Review and Sprint Retrospective of the different teams should be at the same time.

As can be seen from Figure 6 the process works the same as Scrum, only with multiple teams working in parallel. Meanwhile there is one overall Product Owner responsible for the Product Backlog. The Product Owner is the bridge between the customer and the teams. The Product Owner prioritises the Backlog himself, however clarification of the items on the Backlog is done in collaboration with the teams. With LeSS, team members are encouraged to have direct contact with the customer so that the Product Owner can function as a bridge rather than an intermediary. The Product Owner only attends the overall meetings, such as, Sprint Planning 1, Product Backlog Refinement, Sprint Review and overall Retrospective. The individual team meetings are left out.

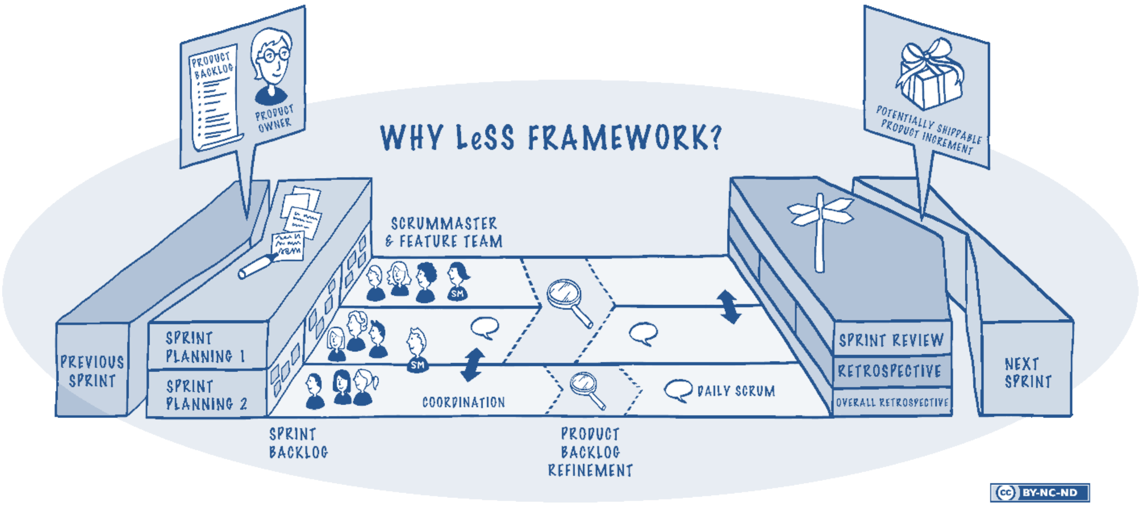


**Figure 6: LeSS Framework (Larman & Vodde, Large-Scale Scrum, 2014)**

With LeSS the Sprint starts with a Sprint Planning 1 meeting. This meeting is attended by the Product Owner and the team representatives. The Product Backlog is the input for this meeting. Each team picks up several items of the Product Backlog for their Sprint Backlog. During this meeting is also decided whether a Multi-team Sprint Planning 2 session is necessary. Next is a Sprint Planning 2 session where the individual teams plan their Sprint. In case a user story influences more than one feature team a Multi-team Sprint Planning 2 is necessary. During this session the teams with dependencies have their own individual Sprint Planning but are located in the same room so questions can easily be asked and cooperation can be found quick (Larman & Vodde, Large-Scale Scrum, 2014).

All teams work on their individual Sprint session just as the regular Scrum. However coordination and integration is required between the different teams. The coordination can either be centralised or decentralised, although decentralised coordination is preferred since teams are self-organised. Babinet and Ramanathan (2008) also mention this in their paper on dependency management in a large Agile environment. There are several possibilities to structure the integration and coordination.

Centralised coordination techniques are Scrum of Scrums, this meeting is joined by team representatives and the same questions are answered as with daily stand-up, except that in the answer should be mentioned to what extend the work influences the work of other teams. Another option is Open Space, in which all team members come together, and each team gives an overview of their issues which might involve other teams. Each team will host parallel meetings of 20 minutes explaining their issues, so the teams can join other teams to discuss issues and learn from each other (Babinet & Ramanathan, 2008). The Town Hall meeting is a public meeting which is joined by anyone who is involved with product development. Participation is on voluntary basis. During the meeting the employees are offered the chance to present their concerns. The Joint Scrum planning meeting is done with the Product Owner and the different teams or their representatives. The Definition of Done should be clear in advance. The epics or themes are written down on cards and the teams are able to select the cards they like to work on. The Product Owner has the final say in the distribution of the cards. Moreover the Product Owner must avoid that all high priority items end up at one team. The last centralised coordination option is the Joint Sprint Review bazaar. With this format each team has their own space in a room where the teams present their work done during the Sprint. The team members walk around to look at each other their work and engage in a discussion.

In addition to the centralised methods there are also several decentralised methods described. One of them is to send so called Chickens to the daily stand-up. This means that a team sends a representative to the daily stand-up of another team to get an idea of what they are doing. A prerequisite is that the team knows where they depend on other teams (this can be done with environment mapping), so the stand-up can be planned at a different time than the dependent team (Babinet & Ramanathan, 2008). Environment mapping as mentioned earlier, can be used in addition to sending chickens. With environment mapping the team puts itself in the middle of the paper draws lines to the other teams where they have dependencies. These dependencies are also named. A simple example of such a diagram is given in Figure 7.

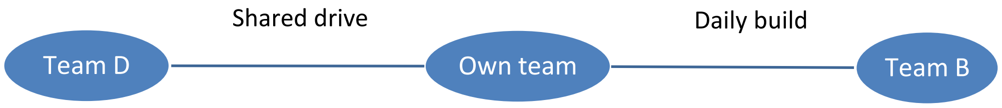


**Figure 7: Example of Environment map**

Furthermore, there is a concept called Travellers, Travellers are experts who join teams who need their expertise during one Sprint. Because these experts join different teams each Sprint they know what is going on in the different teams and can share this knowledge with the teams. Additionally, Communities of Practice (CoP) is another way to share information. A CoP exists of volunteers who share a common interest who like to practice this with fellow employees. The increase of shared space also stimulates knowledge sharing by decreasing distance and increasing meeting opportunities. Communicating in code between the developers can help since code is unambiguous. The final option for decentralised coordination is coordination working agreements. In these agreements teams can write down who, when, where and how they are going to plan their coordination with other teams. Since this is an agreement made at the beginning of the Sprint this needs to be reflected in the Retrospective. (Larman & Vodde, 2010)

Halfway the Sprint the teams perform Product Backlog Refinement to prepare stories for the next Sprint. An overall Product Backlog Refinement session is done with the Product Owner and the different team representatives. Afterwards the individual teams do a Product Backlog Refinement for their own backlog, and in case there is overlap between teams a Multi-team Backlog Refinement is done.

At the end of the Sprint all teams integrate their work into a Potentially Shippable Product Increment (PSPI). This PSPI has to meet the Definition of Done which is product based and equal for all teams. The Sprint Review is done by all the teams and stakeholders together. Afterwards the teams have their individual Retrospective. At last there is the overall Retrospective which should at least be attended by Scrum Masters, team representatives and the Product Owner. The purpose of this meeting is to look back on the previous Sprints on product level and how the different stakeholders worked together to deliver a PSPI.

In case of LeSS Huge, Area Product Owners are introduced, these people are responsible for an area of the product and the different teams working on that part. The Area Product Owners reports back to the Product Owner. This structure can be found in Figure 8.

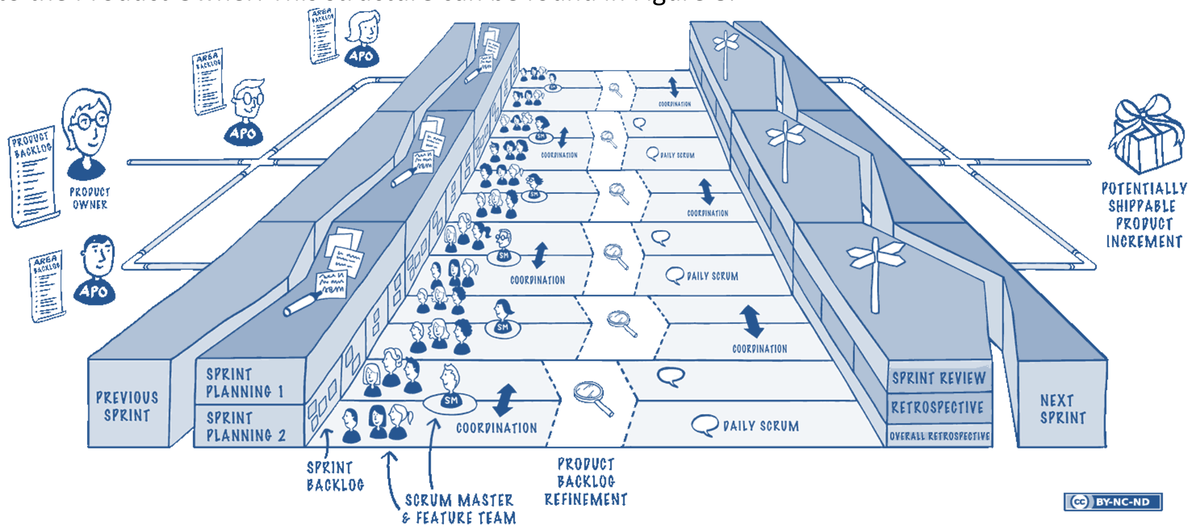


**Figure 8: LeSS Huge (Larman & Vodde, 2014)**

## 3.2 Scaled Agile Framework®

In this section the Scaled Agile Framework® or SAFe® for short is discussed. SAFe® consists of three levels; Team, Program and Portfolio. Each level describes the usage of Agile at that specific level. The overview of the framework can be found in Figure 9.

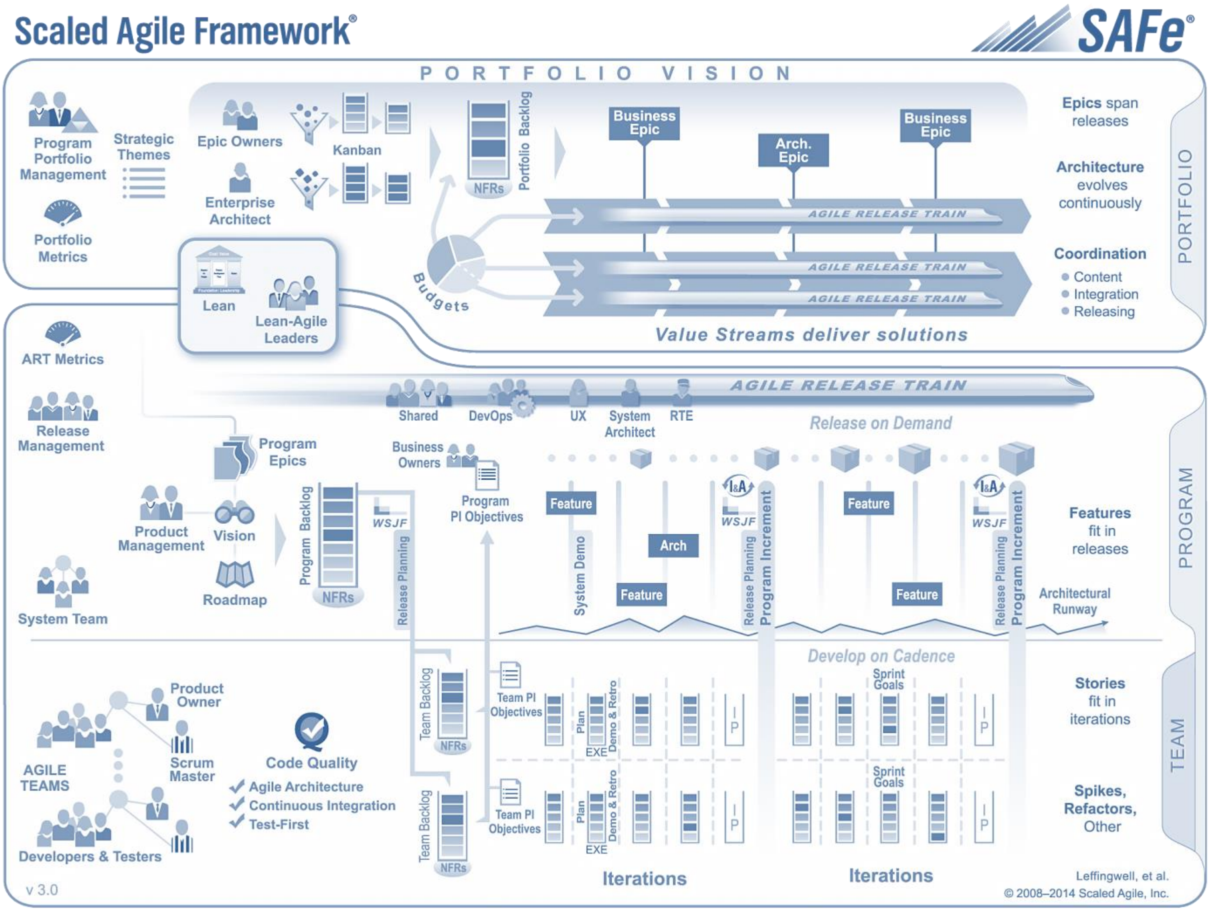


**Figure 9: Scaled Agile Framework®** Reproduced with permission from © 2011-2015 Scaled Agile, Inc. All rights reserved

### 3.2.1 Team

The team level exists of Agile teams similar to Scrum teams. The teams have Product Owners, Scrum Masters, Developers and Testers. Each team has their own Team Backlog with stories. The teams typically have interfaces with architecture, product/program management, and quality. SAFe® suggests that it could be useful to add an architect to the teams in case architecture is determined at systems level together with the business analysts. Teams work in iterations (or Sprints in Scrum terms) which exist of define, build and test. So far this is all according to the Scrum methodology. (Leffingwell, 2011) When scaling the team the next level is reached; program.

### 3.2.2 Program

When organising the teams at scale a feature or component division can be chosen. Component teams require the shifting of user stories among the different features and therefore a dependency on other teams. However this results in robust components. When working in feature teams there is less overhead and the backlog is simpler compared to the component teams. At Program level a system team is needed in order to complement the component or feature teams to make sure the integration of the features/components work well. The Release Management Team is there to support the teams with release governance authority. The Product Manager is responsible for the result of the product overall. In order to manage the product in a good way a vision is needed so the different teams know why they are working on a product, the vision is time independent. The vision is the basic input for the Program Backlog, the features on the Program Backlog have to meet the functional and non-functional requirements. Based on the vision a Roadmap is created to show which features are planned in what timeframe. The development of the product is done within an Agile Release Train (ART). The ART is a "standard cadence of time boxed iterations and milestones that are date- and quality-fixed but scope variable" (Leffingwell, 2011) At the end of the ART a potentially shippable increment is released. During the Release Planning the vision is translated into features and stories. The frequency of the Release Planning is depending on the responsiveness required from the client. (Leffingwell, 2011)

### 3.2.3 Portfolio

The portfolio is managed by the Program Portfolio Management, they are responsible for the strategy and investment funding, governance and the program management. At different parts of the portfolio level metrics are included, these are the lean portfolio metrics, portfolio Kanban board, measuring epics, portfolio management self-assessment and the enterprise balanced scorecard. The strategic themes have the function of creating context for decision making. It helps with allocating budgets to the different value streams and release trains. Additionally the goal of the strategic themes is to help as input for the portfolio and program backlog. The epic owners are responsible for an epic, they keep in contact with the stakeholders of the epic across the different ARTs and business units to have a successful implementation in the end. It is the enterprise architect his job to define the architecture strategy. This exists of the five elements; choice of technology and usage, system architecture strategy, infrastructure strategy, inter-program collaboration, implementation strategy. The epic owners and enterprise architects fill their Kanban which is the input for the portfolio backlog.

## 3.3 Disciplined Agile Delivery

Disciplined Agile Delivery (DAD) is defined by IBM rational as "an evolutionary approach that regularly produces high quality solutions in a cost-effective and timely manner via a risk and value driven lifecycle. It is performed in a highly collaborative, disciplined, and self-organising manner within an appropriate governance framework, with active stakeholder participation to ensure that the team understands and addresses the changing needs of its stakeholders. DAD teams provide repeatable results by adopting just the right amount of ceremony for the situation which they face. (Ambler, 2009)"

The Disciplined Agile Delivery framework works with three phases, which are executed in Sprints. An overview of the framework is given in Figure 10.

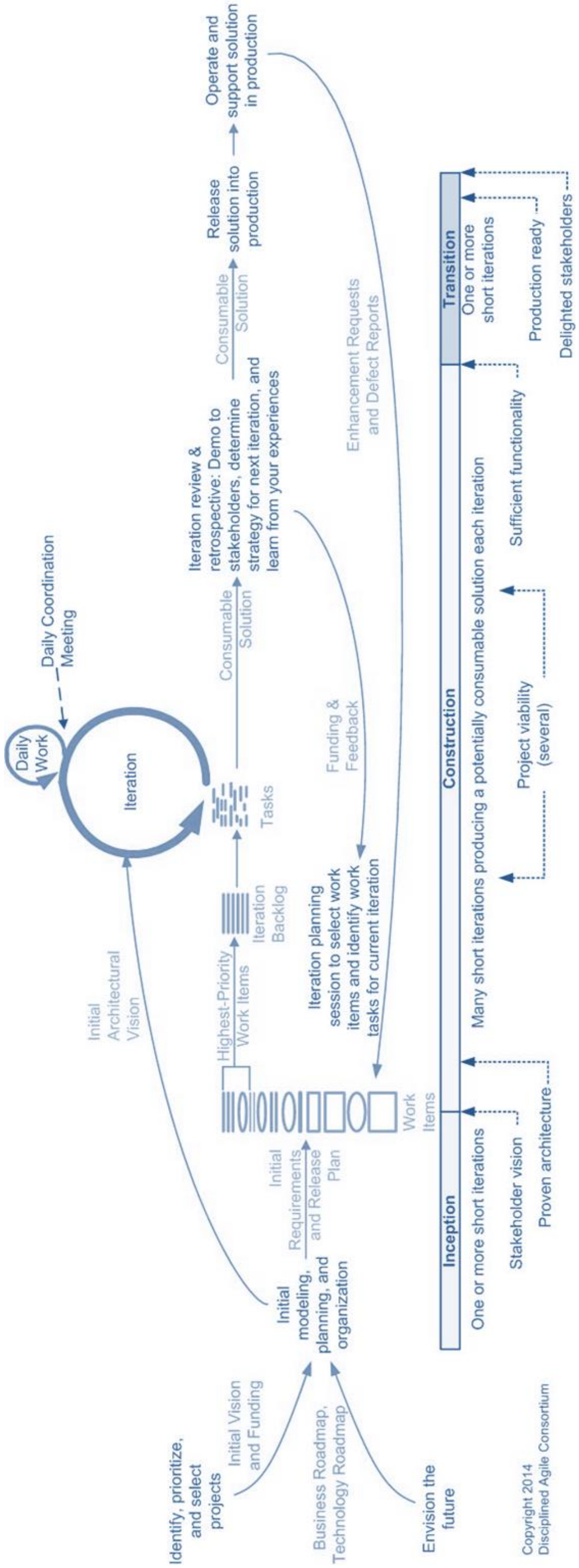

Identify, prioritize, and select projects
Initial Vision and Funding
Business Roadmap, Technology Roadmap
Envision the future
Copyright 2014 Disciplined Agile Consortium
Initial modeling, planning, and organization
Initial Requirements and Release Plan
Initial Architectural Vision
Work Items
Highest-Priority Work Items
Iteration Backlog
Iteration planning session to select work items and identify work tasks for current iteration
Tasks
Iteration
Daily Work
Daily Coordination Meeting
Funding & Feedback
Consumable Solution
Iteration review & retrospective: Demo to stakeholders, determine strategy for next iteration, and learn from your experiences
Enhancement Requests and Defect Reports
Consumable Solution
Release solution into production
Operate and support solution in production
Inception
One or more short iterations
Stakeholder vision
Proven architecture
Construction
Many short iterations producing a potentially consumable solution each iteration
Project viability (several)
Sufficient functionality
Transition
One or more short iterations
Production ready
Delighted stakeholders

**Figure 10: Disciplined Agile Delivery Framework (Lines & Ambler, 2015)**

The first phase is named the Inception Phase, this happens at the start of a project. During the coordination part of this phase teams are initiated and meetings are held with stakeholders for envisioning the project, this will last up to a few hours. Regarding collaboration, the eventual team is composed, ideas for requirements and architecture are discussed. Also the project is aligned with the enterprise strategy as well as a shared vision is created. Additionally an initial release planning is made and the environment for the project is set up. The collaboration tasks will take a couple of weeks. The end of the inception phase is concluded with a light-weight milestone review and the communication of the vision towards the stakeholders. At the end of this phase there should be a stakeholder consensus.

The second phase is called the Construction Phase. In this phase the project is built. The first few hours of the iteration are spend on coordination, so the planning and modelling of the iteration. Collaboration is divided into standard and advanced practices. Standard practices exist of; work visualisation, daily coordination meeting (daily stand-up), refactoring, developer regression testing, model storming, continuous integration, sustainable pace, requirement prioritisation, architecture spike, collective ownership, burn down chart and automated metrics. The advanced practices are; test-driven development, acceptance test driven development, continuous deployment, look-ahead modelling, parallel independent testing, continuous documentation, non-solo development and look-ahead planning. At the end of the iteration a demo of the potentially consumable solution is given, a retrospective is held, an update is given on the release planning and a “go forward” strategy is determined.

The last phase is the transition phase. At the coordination stage phase planning is done. Collaboration exists of transition planning, end-of-lifecycle testing and fixing, data and user migration, pilot/beta the solution, finalise the documentation, communicate deployment, prepare support environment, training of stakeholders. In the conclusion stage a production readiness review is held and the solution is deployed. After this finale phase production is ready and the product is ready for actual usage. (Ambler & Lines, 2010)

DAD describes different ways to scale. One of them is to create a product delivery team, this team usually exists of the team leads of the different delivery teams. The delivery team organises itself, and is autonomous in what way and frequency information is shared. The delivery team is chaired by the portfolio manager. Scaling can also be done by creating a product management team, where the product owners of the different delivery teams come together with the chief product owner. The product management team is responsible for the vision of the portfolio. Additionally the product management teams prioritise the work items for the overall project and divided the work items among the different delivery teams. Domain experts can be asked to join the product management team in case of complex domain issues. Another option is to create an Architecture team, composed of the architecture owners of the different delivery teams. This method is also mentioned in the paper by Babinet & Ramanathan (2008) It is their responsibility to form a strategy for the architecture of the program or even at enterprise level. Moreover the team helps out with technical issues at the architectural level. The team is led by the chief architecture owner. Furthermore an independent test/integration team can be put into place to serve the great domain complexity. The independent team can run a parallel end-to-end test to support the delivery teams. In Figure 11 is shown how the teams which support scaling are organised. (Lines & Ambler, 2015)

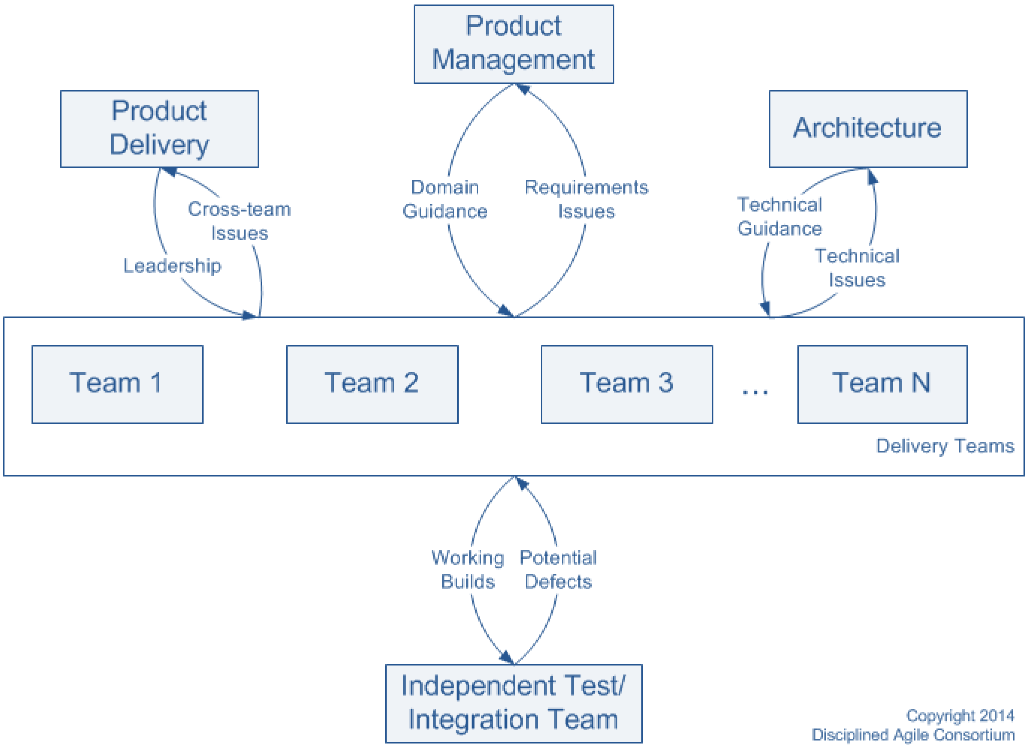


**Figure 11: Scaling DAD (Lines & Ambler, 2015)**

The delivery teams can be organised in different ways, component teams, feature teams, functional teams and internal open source. The DAD framework however does not show a preference for one option how to organise your team. When working together with several teams, a couple of strategies can help the teams to perform better. DAD describes six of these strategies. The first is to do more requirements exploration upfront to create a better understanding of the stakeholders' desires. Secondly is to do more up-front architectural modelling to create a better understanding of the strategy for developing a solution. The third is to spend more time on initial planning so dependencies between the teams become visible. Fourth is to adopt more sophisticated coordination strategies such as the ones described above. The adoption of more sophisticated testing strategies is the fifth strategy. Finally, regular integration. This is especially important when working together with a large number of teams, when integration becomes harder. (Lines & Ambler, 2015)

## 3.4 Comparison

After having discussed the different frameworks in the previous section, this section gives a comparison of those frameworks. First the similarities are discussed and second the differences are given.

### 3.4.1 Similarities

At team level all frameworks have adopted the Scrum methodology, however DAD has given their own names to the roles and activities. All frameworks emphasise that each company is different and that their framework does not necessarily fits every organisation. Organisations need to be selective on what to implement and what not. Quality assurance is addressed in all frameworks, however they all have a different way of incorporating it. LeSS uses one Definition of Done and by incorporating knowledge sharing good quality is promoted. SAFe® has a code quality check incorporated at the team level. DAD makes sure that quality assurance professionals are incorporated in the Scrum teams as well as testing is done independently.

### 3.4.2 Differences

The frameworks differ in how much the entire organisation is engaged in scaling Agile. LeSS shows limited involvement of the rest of the company in order to scale. SAFe® describes how to scale up at more levels of the organisation namely; team, product and portfolio. DAD does not scale up to enterprise level. SAFe® puts focus on a clear vision and strategy from the top management regarding the product, DAD does this on project level. LeSS and DAD show different ways to organise scaling by adding specialty teams. LeSS provides tools to increase knowledge sharing between the teams, although without a clear vision and strategy for the product it is hard to decide how to use the tools. Regarding the team set up, LeSS and SAFe® show a preference for Feature Teams, DAD mentions four different ways to sort the teams but does not mention an overall success composition. LeSS provides a lot of different ideas of how knowledge can be shared and how integration and coordination can be deployed. LeSS is the only framework which makes a distinction between the number of teams which need to scale. Regarding architecture SAFe® and DAD pay attention to this topic.

## 3.5 Agile Scaling in Practice

To get an idea how Agile Scaling works at other companies, Xebia and AgiliX were contacted. AgiliX introduced me to Company B. Unfortunately Xebia did not reply. Additionally an Agile knowledge sharing session is joined from Quint Wellington. The goal was to find out whether other companies are using certain tools and techniques which are useful to solve the scaling issues at Topicus.

### 3.5.1 Company B

With help from AgiliX, I was introduced to one of the cluster managers at Company B Hengelo to see how Agile Scaling is applied at the Department C.  Company B is working in component teams in Sprints of four weeks. The teams are all part of the department Sprint together with the integration and test teams. Each component team is responsible for a specific part of the process taking place within the radar.  All the different components which are produced are integrated by a dedicated integration team to complete the feature which has added value for the customer, this is done during the department Sprint. There are separate desks and large monitors aligned specially for integration so when walking by the integration can be seen on the monitors. After integration, the integration is separately tested by a test team, also during the department Sprint. Architecture is also done by a specific team. Currently the System Architect, Processing Architect, Processing Tester and System Tester are involved in the department Sprint, but their work is not part of the department Sprint. Company B is planning on involving the processing roles into the Sprints in the coming year. Tasks, stories, features and epics all need to pass their own test in order to be done. For progress tracking one wall within the department is used for the roadmap. Another wall is used as Sprint board with post-it notes, this version also exists digitally in Jira.

Company B decided not to implement feature teams because of their recent change from project teams to component teams. Therefore the Scrum teams were scaled to department level, where all teams work from the same Product Backlog and on the same Sprint goal. Table 3 shows which Scrum related meetings are done at department level, and which meetings are done by the individual Scrum teams.

|  | Department Level | Individual Team Level |
|---|---|---|
| Sprint planning | X | X |
| Sprint goal | X |  |
| Product Backlog | X | X |
| Sprint backlog | X | X |
| Daily stand-up | X | X |
| Backlog refinement | X | X |
| Sprint retrospective | X | X |
| Sprint review | X | X |

**Table 3: Meetings Department vs Individual level**

The different meetings on department level are attended by people with different roles and functions. An overview of who is attending which meeting on scaled level is shown in Table 4. In the next few sections the different meetings are explained.

|  | Product Owner | Scrum Master | Team representatives[19] | External Stakeholders | Project Managers | System testers | Project architects | Line management | Integrators | Domain architects | Domain experts |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Defining Sprint goal | X |  |  |  |  |  |  |  |  |  |  |
| Order Product Backlog | X |  |  |  | X |  | X | X |  |  |  |
| Sprint Planning | X |  | X |  |  |  | X | X |  |  |  |
| Daily stand - up |  | X | X |  |  |  |  |  |  |  |  |
| Backlog refinement |  |  | X |  |  |  | X |  | X | X | X |
| Sprint retrospective | X | X | X |  |  |  |  |  |  |  |  |
| Sprint review | X |  |  | X | X | X | X | X |  |  |  |

**Table 4: Attendance department meetings**

#### 3.5.1.1 Sprint planning

In order to manage the Sprint planning efficiently, all features which are likely to be added to the department Sprint must meet the Definition of Ready. Then the Product Backlog is prioritised, firstly by story mapping for one project and secondly by prioritising features across the projects.

Story mapping for one project is done by sorting epics/features from high to low priority. Per epic/feature the stories are divided among the teams with the right component specialisation. Before an epic/feature can be accepted and planned for a department Sprint, it needs to meet the

[19] Team representatives are not necessarily the same person. Depending on the topic and time available it can be decided whether it is useful for a specific person to join a meeting.

Definition of Ready. Since all projects are run simultaneously per department a department Sprint backlog is needed.

Since every project has done their own story mapping. The epics/features of the different projects are prioritised among each other by the product owner together with the line management, project management, project architects and the team representatives. In Figure 12 this process is depicted. Sometimes an entire epic/feature does not fit to close the backlog. Even though stories are supposed to be built specific teams sometimes a story needs to move to another team to be able to fill the entire Sprint with epics/features. In very exceptional cases the engineers, which are not working in Scrum teams, can help out.

The Sprint planning is made on Tuesday which is the start/stop day, the last Friday of the previous Sprint a double check is done whether there are any impediments regarding the development of an epic/feature. On the last Monday of the previous Sprint a check with the project managers is done to see if there are any last minute priority changes within the project which requires alteration of the Sprint planning.

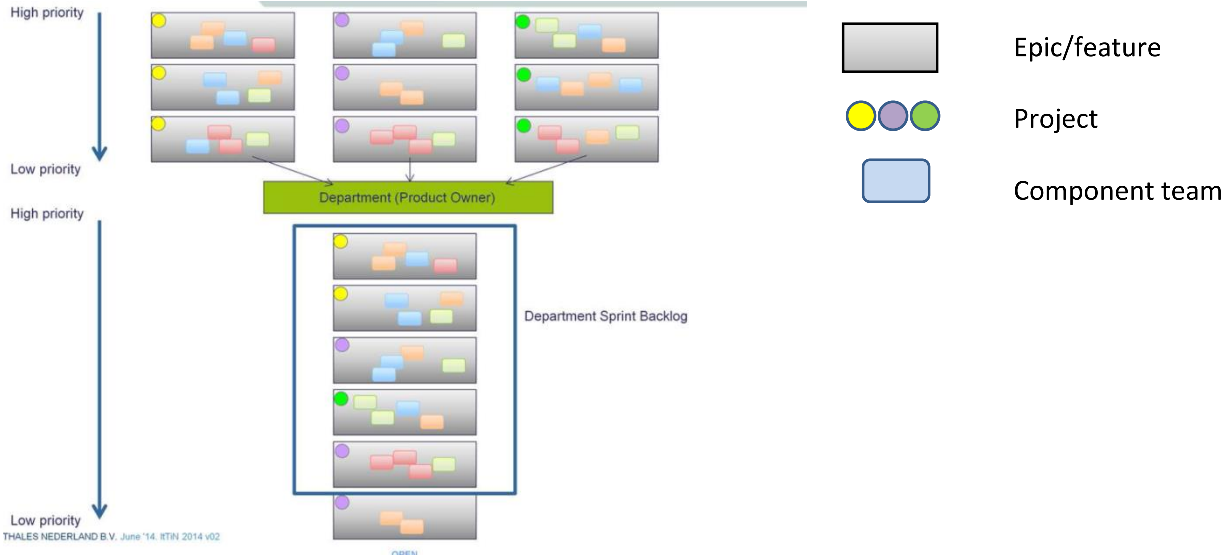

High priority
Low priority
Department (Product Owner)
High priority
Department Sprint Backlog
Low priority
THALES NEDERLAND B.V. June '14. ItTiN 2014 v02
OPEN
Epic/feature
Project
Component team

**Figure 12: Department Sprint at Company B**

### 3.5.1.2 Daily stand-up

The (department) Scrum Masters and team representatives are always present. The other roles can join the stand-up voluntarily. Jira is used to track the progress of the Sprint. Additionally the wall of the coffee cubicle is used as Sprint board. This Sprint board is used during the daily department stand-up meeting where team representatives update each other and the Chickens on the progress. Moreover, impediments and dependencies are discussed.

### 3.5.1.3 Backlog refinement

The backlog refinement sessions take place twice per Sprint. During these sessions the features at the top of the backlog are refined into stories, suitable for the individual Scrum teams. The stories are then allocated to the right Scrum team, so the teams can do the refinement of the stories themselves during their individual refinement session.

### 3.5.1.4 Sprint Retrospective

The purpose of the department retrospective is to find out whether and how the performance of the department can be improved. Furthermore the impediments which are applicable across multiple teams, or impediments which are not able to be solved by the individual teams, are solved during this session.

#### 3.5.1.5 Sprint Review

The Sprint review is to show what work should have been done, and what is done eventually. The next topics are optional. The current Product Backlog is presented. Additionally the department velocity is given, so this can be related to the project milestones. The department Product Owner presents the Sprint review information to the external stakeholders. Also a demo is given of one or a couple of features.

#### 3.5.1.6 Comparison

When comparing Topicus with Company B, Topicus has their integration integrated within the Scrum teams and has therefore implemented the Scrum methodology more deeply into the organisation and process. Except that Company B does not do their integration within the Sprint cycle of the teams, the concept of having a separate open area to do integration is a positive contribution to knowledge sharing. I believe that this openness and visibility creates mutual understanding as people can see progress when walking by.

Topicus has their teams divided per customer, within the different customer teams there are no specialisations. This approach is different than at Company B where each component of the radar is represented by one team. As mentioned with the description of the different frameworks, feature teams are advised. Both Company B and Topicus do not meet this standard. Although Company B has one backlog, something which is requested within in Topicus. However, for Topicus to have a shared backlog across the teams requires to change team dedication. This means that the teams should not be customer specific. Also all teams should have the same start/stop day as well as a common Definition of Done and Definition of Ready.

Additionally Company B has a clear hierarchy how epics, features, stories and tasks relate to each other, at Topicus this is not the case. It would avoid confusion in case Topicus has a commonly accepted hierarchy of epics, features, stories and tasks. Jira[20] is also used at Topicus, although Topicus uses more tools of Atlassian like HipChat[21] and Confluence[22]. Topicus is not using paper versions of a Sprint board nor a roadmap. Paper versions are not a necessity since this information can be found on Jira, especially since at Topicus all teams have a large monitor in their room to display Jira.

Moreover at Topicus not all teams are located in the same area. At Company B one of the line managers mentioned that because they work in an open and shared space it is very easy to connect and share information.

Sprint planning at Topicus is done separately by the customer teams with their own Product Backlog. Joint Sprint planning will only be possible at Topicus when teams work from one Product Backlog. The way Company B organises and prepares Sprint Planning, makes sure this meeting can be done effectively with limited discussion. This method can also be applied at Topicus where the different projects can for instance represent the different customers. Shared daily stand-ups are done within the Customer X teams.

[20] Project management support tool
[21] Internal chat box
[22] Platform for team collaboration

Backlog refinement is done every once in a while based on external influences. The business analysts keep an eye on the backlog at Pakket. While at Customer Y and Customer X this is done from the customer side based on the projects. This is rather ad hoc and contradicting the way Company B does this.

The Sprint retrospectives and Sprint Review are done by the individual teams as well as on scaled level.

Overall the structure at Company B on higher level is better organised due to a uniform way of working, shared meetings, shared backlog and shared Definition of Ready and Definition of Done.

*To sum up, in this chapter the extensive list of Agile Scaling frameworks was reduced to three frameworks; SAFe®, LeSS and DAD. These frameworks were analysed and compared. Additionally an overview has been given on how Agile Scaling is done at Company B. The way of working of Company B was also compared with Topicus' way of working. With help of this information the next chapter will discuss which framework to choose. In this chapter the third research question was answered which is about the different Agile Scaling frameworks, and how these are related.*

# 4 Decision Making

*In Chapter 3 the different frameworks LeSS, SAFe® and DAD were discussed. Also a description of scaled Scrum at Company B was given. Additionally the employees at Topicus shared their concerns and desires which can be found in Chapter 2. In this chapter the decision is made which (part of the) framework is going to be tested based on the information of the frameworks, Company B, Scrum Masters, Product Owners and Management Team.*

The decision for a framework can be based on several criteria. First, the priority of the problems which need to be solved can be criterion. Second, the framework which will solve most issues can be chosen. Another criterion could be time. Regarding time there are several possibilities such as, implementing an entire framework in 4-6 weeks' time. Another option is to implement certain aspects of a framework. The expectation is that with the attempt of the implementation of the entire framework there will be less results visible at the end of the testing period since this requires more time. On the other hand, when only implementing a specific part of a framework, this possibly results in better measurable outcomes.

Together with my supervisor and one of the management team members was decided that the solution which is expected to show quick results during an experiment is preferred. So which options show quickest result in shortest amount of time. This means that the testing and implementation of one entire framework is excluded.

SAFe® would provide the most complete solution for shaping the organisation in an Agile way, since it addresses all levels in the organisation to adopt Agile. It does not only focus on the team level but also program and portfolio. Additionally it gives guidance on how to deal with quality of code and architecture. However it does not support the company with practical tools and techniques.

The way Company B has structured their meeting and planning sessions similar to the sessions proposed by the LeSS framework. The LeSS framework however, does not incorporate how to prioritise the backlog in case of different customers involved. At Company B a clear structure is given how to prioritise within and among the different projects. The way Sprint planning at scale is organised at Company B matches with the demand for more structure, a centralised and better planned backlog of Topicus. A shared Product Backlog and joint Sprint planning can also help to solve issues regarding knowledge sharing since the teams know what the other teams are doing. However the Topicus teams are mixed with customer teams which makes it hard to change the current teams to component or feature teams and additionally implement the scaled meetings.

As stated in Chapter 2, knowledge management and knowledge sharing have the biggest priority to be improved. The LeSS framework offers a lot of manageable and easy executable options to improve knowledge sharing and interaction between the teams. Therefore the LeSS framework is chosen to experiment with, at Topicus. The different Centralised and Decentralised Coordination methods of the LeSS framework are described in Section 3.2. The different methods are listed below.

- Centralised
  - Scrum of Scrums
  - Open Space
  - Town Hall
  - Joint Sprint Planning
  - Joint Sprint Review Bazaar

- Decentralised
    - Chickens
    - Travellers
    - Communities of Practice
    - Increase shared space
    - Communication in code

In two separate meetings the different Centralised and Decentralised Coordination methods were discussed. The reason for having two meetings was to create a bigger opportunity for the people to attend the meeting. During these two sessions the same presentation was given. In this presentation a short introduction to LeSS was given. Thereafter the different Centralised and Decentralised Coordination mechanisms were described. After which there was time for discussion about which options to choose, where and how to test. The first meeting was attended by two Customer Y Scrum Masters, Customer Y Product Owner and Scrum Master of the component team. The other meeting was attended by two Product Owners, Scrum Master package, Scrum Master Customer X and the Innovation manager.

A summary of the outcome of these meetings can be found in Table 5. At the end of both meetings it can be concluded that Scrum of Scrum is already in place. My supervisor had worked with the Open Space technique before and in his opinion it is a powerful tool. However the others assumed the organisation of such meetings is difficult and there would not be sufficient topics to discuss. The Town Hall meetings are rejected because it is too voluntarily, the people who attended the meeting feared that this option will not achieve the best results. The Joint Sprint Planning ended in a second place with three votes. The Joint Sprint Review Bazaar, is not adopted because the feedback you get during such a demo is too late according to the attendees, because the part demonstrated is already finished. Although according to the Scrum methodology the Sprint Review is there to get feedback so the feedback can be implemented in the beginning of the next Sprint. Additionally previous demo sessions were not considered useful, and therefore it is supposed that the Joint Sprint Review Bazaar will not turn out to be helpful.

Sending Chickens in combination with environment mapping received two votes. The Travellers concept received most votes (4). Communities of Practice are already in place, however they are not giving the required results yet. It needs more formalisation as well as dedication. The increase of shared space was not possible as a result of the lay out of the building. The communication in code was not considered an option because the knowledge sharing should be accessible for everybody and as not everybody is able to read and write code this is not possible.

|  | Technique | Selected | Rationale | Additional |
|---|---|---|---|---|
| Centralised | **Scrum of Scrums** | No | - Already used |  |
|  | **Open Space** | No | - difficult to organise<br>- not sufficient topics to discuss. |  |
|  | **Town Hall** | No | - too voluntarily |  |
|  | **Joint Sprint Planning** | Yes | + know what features other teams work on | Turned out not to be possible |
|  | **Joint Sprint Review Bazaar** | No | - feedback too late<br>- previous demo sessions not useful<br>- start/stop not on same day |  |
| Decentralised | **Chickens** | Yes | + easy to implement<br>+ takes little time and effort | In combination with Environment mapping |
|  | **Travellers** | Yes | + learn what it is like to work for customer |  |
|  | **Communities of Practice** | No | - Already used in own form |  |
|  | **Increase shared space** | No | - not possible due to building |  |
|  | **Communicate in code** | No | - not everybody can read code<br>- information not accessible for everybody |  |
|  | **Environment mapping** | Yes | + easy to implement<br>+ takes little time and effort | In combination with Chickens |

**Table 5: Summary decisions on LeSS techniques**

Because of the current location and team changes of the Customer Y teams, they did not see any chance that any of the options will work for them. Additionally the Customer X teams are currently in a turbulent situation so their Scrum Master could not give a confirmation whether the Customer X teams could actively join the experiment.

The coordination methods with the most votes were chosen. Since for the execution of this experiment support is needed from the Product Owners and Scrum Masters it is important that these people feel comfortable with using the chosen coordination methods. The methods selected for an experiment are:

- Joint Sprint Planning
- Chickens (in combination with environment mapping)
- Travellers

When refining the ideas to put them into action, we experienced some hurdles for the joint Sprint planning and Travellers. The Joint Sprint planning eventually was not an option considering the amount of change required beforehand. Joint Sprint planning is useful in case of a shared Product Backlog, this is for example the case within the Customer Y, Pakket and Customer X teams. Within Customer X teams there is a Joint Sprint planning. At Pakket this planning is made by the Product Owner. The Customer Y teams have individual Sprint planning per team together with their teammates at Customer Y. Also the Sprints do not start/stop on the same day so a shared planning is not possible.

Also the original concept of the Travellers required some alteration, since there was no demand for experts joining teams for an entire Sprint. But, there was a demand for getting somebody in the teams with a different view. Therefore, the original concept was changed to exchanging team members. Moreover, we encountered some resistance regarding the trial of the Travellers. The current situation of the Customer X and Customer Y teams was mentioned as a reason not to participate in the Travelling experiment even though the management team had already given their approval. This argument did not erase the concerns of the Product Owner and the Scrum Master, so a quick meeting with a management team member and chief Product Owner was planned together with the people who had concerns so these concerns could be shared. During this meeting several alternatives of the Travelling concept were offered. These alternatives are sending and receiving Travellers between other business lines and exchange employees between Customer Z and Customer Y teams. These options are out of scope, since the focus is on improving the collaboration between the Customer X, Customer Y and Pakket teams which work on the same product on the same codebase and therefore they share dependencies. Eventually the decision was made with the Management Team to exchange members from one of the Customer Y teams with one of the Pakket teams because these teams are most stable. The starting date had to be postponed once, because of a demo which had to be made by the Pakket team. This caused a delay of one month.

When discussing how to shape the experiment we came to the conclusion that the exchange of team members is not possible. This is because the Customer Y team has only one developer and the developer of the Pakket team did not feel confident enough of taking over an entire role on her own. So several alternatives were discussed. One of them was that the developer of Pakket would join the Customer Y developer for one week so she could get familiar with the work at the Customer Y team. After this week the developers could really exchange places. Another option was that the developer of Pakket joins the Customer Y for an entire Sprint. Since support and dedication was needed of the developers and their Scrum Masters, they were the ones to decide which option to choose. They decided to go for the second option where the developer of Pakket joins the Customer Y developer for one Sprint.

An overview of all decisions made so far in this research is given in Figure 13.

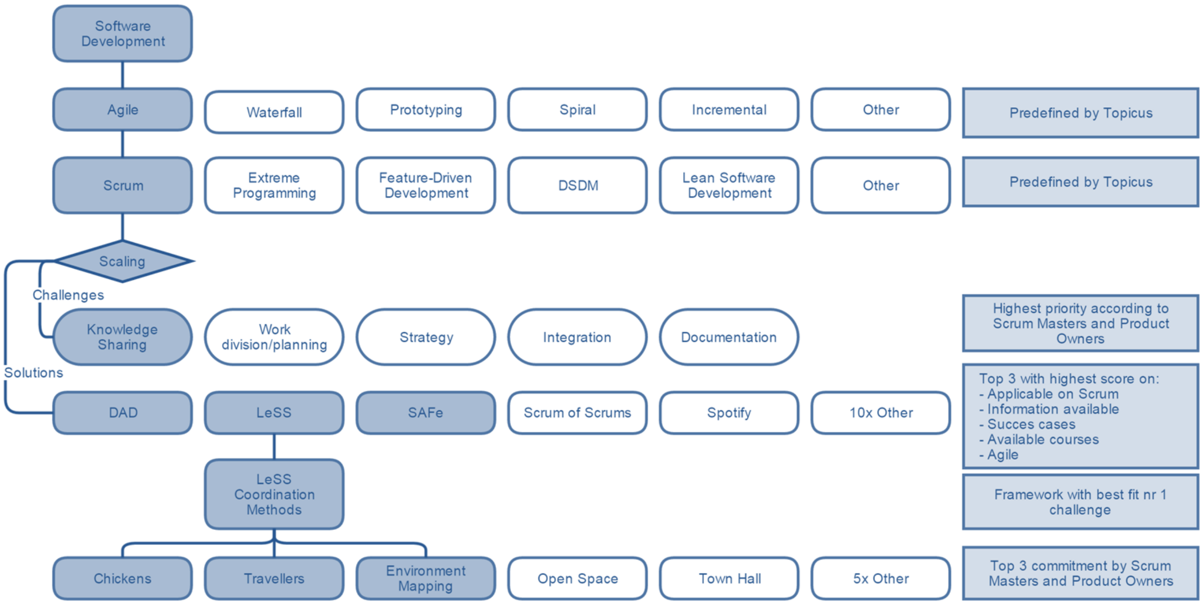


**Figure 13: Overview decisions**

*To recap, this chapter discussed to what extend the frameworks are possible to implement at Topicus. The choice was made to do an experiment with LeSS. Several coordination techniques which focus on knowledge sharing are shared with the Product Owners and Scrum Masters. An experiment is done using the Chickens and Travellers coordination methods. How this experiment is executed and how the results are measured is described in the next chapter. In this chapter the fourth research question is answered about which Agile Scaling framework is best suitable for the Product D teams of Topicus Finance.*

# 5 Methodology

*The reasons to test Chickens and Travellers was written in the previous chapter (4). This chapter elaborates on how the interaction techniques will be tested, measured and evaluated.*

According to van Aken et al. (2012) a post-test only evaluation is the least valuable since no comparison can be made and possible improvements cannot be shown. They advise to do a pre- and post-test comparison. In order to undermine possible results which are not due to the intervention, a comparative post-test is advised (quasi-experimental design). The end of the experiment should be evaluated with an formative evaluation. Contradicting to the summative evaluation which only looks at what is measured, the formative evaluation looks at the why and how of the results. (Berends, van Aken, & van der Bij, 2012)

## 5.1 Pre-experiment

The research methodology used to conduct this research contains elements of an action research with a pre- and post-test. This way of doing research best supports the dynamic research environment. The preliminary research already showed where the problem is (Chapter 2) and showed which decisions are made regarding testing (Chapter 4). In order to make the effect of the knowledge sharing techniques more explicit a questionnaire (see Appendix B) was added. This questionnaire is distributed among all team members with the role of tester, developer and information analyst. The number of people invited for the questionnaire was 55. The results are supposed to give a better overview of the current practices in knowledge sharing among the testers, developers and information analysts. The preliminary research was based on the stories of the Product Owners, Scrum Masters, Architects as well as observation during the Sprints by joining the teams at their office space.

## 5.2 Experiment

For this research two techniques are tested. The approach of how to test each technique is written down in the sections below.

The experiment is kicked-off by visiting the daily stand-up of the teams and by explaining them the expectations. Also the questionnaire is filled out during the stand-up.

To measure the effect of the techniques, the meetings are attended where the techniques are applied. Additionally I talk to the people to get an idea of how the changes are received. Moreover observation of the teams is done as well.

### 5.2.1 Chickens

With the Chickens technique all the Customer Y, Customer X and Pakket teams were able to participate. The plan was that two times a week a person from a Customer Y, Customer X and Pakket team joins the stand-up from another team which belongs to another customer. The teams could choose themselves which team they wanted to visit based on the dependencies they have. To make it easier to plan the visits, an overview was made of the stand-up times so the teams can select when they attend which meeting.

### 5.2.2 Traveller

The Traveller concept was only tested within the Customer Y and Pakket teams due to the current circumstances of the Customer X teams. During a period of one Sprint, one team member of a Pakket team joined one of the Customer Y teams. The alternative opted for the “free” experts according to the literature was to exchange team members between teams. This was not executed because the Customer Y team has only one developer. Therefore a developer from Pakket joined a Customer Y team for one Sprint. Since among the Customer X, Customer Y and Pakket teams there is close contact, it was chosen not to exchange team members between Pakket, Customer Y or Customer X teams. In a meeting with the Scrum Masters and the to be exchanged team members, an overview was made on what should be arranged in order for the team member of Pakket to be able to work, such as building access, transportation and log in details. Also in this meeting was discussed what day is the most convenient to start and which days are at the customers' site.

### 5.2.3 Resistance

Not all the teams were able to contribute to the experiment, therefore it was chosen to have all the teams who were able to participate, be a part of the experiment. Another option which was considered, was to have each technique tested by two teams. Although this option was easier to manage, due to different sub-cultures within the teams, this option was not preferred because it is hard to conclude whether the technique works because of the technique or because of the two specific teams. Preferably the Traveller experiment lasts longer than one Sprint. However due to the resistance described in the end of Chapter 4. The Traveller experiment had to start later than the Chickens. Due to the time constraint of this research an experiment which lasts longer is not possible.

## 5.3 Post-experiment

At the end of the experiment a similar questionnaire (see Appendix E) as in the beginning was done to see if the figures show improvement. The post-experiment questionnaire was distributed among the same audience as the pre-experiment. Moreover, after the experiment informal conversations with the teams are held to gain more insight in how they experienced the Travellers and Chickens. So the information gathered in this phase is both quantitative and qualitative.

During the evaluation the results from the pre- and post-experiment were compared. For the evaluation of this experiment two methods were used. The formative- and summative evaluation. With the summative evaluation the effect of the experiment was reviewed. The formative evaluation helped to get insight in how and why certain outcomes happened. The teams which did not actively take part in the experiment, which means the teams did not send or receive a Traveller and did not send a Chicken, are used as a control group. This way it is easier to draw conclusions on whether changes in the level of knowledge between teams are due to the experiment or other circumstances.

*To summarise, this chapter described how the experiment is going to be executed. Additionally a description is given on how the results are measured.*

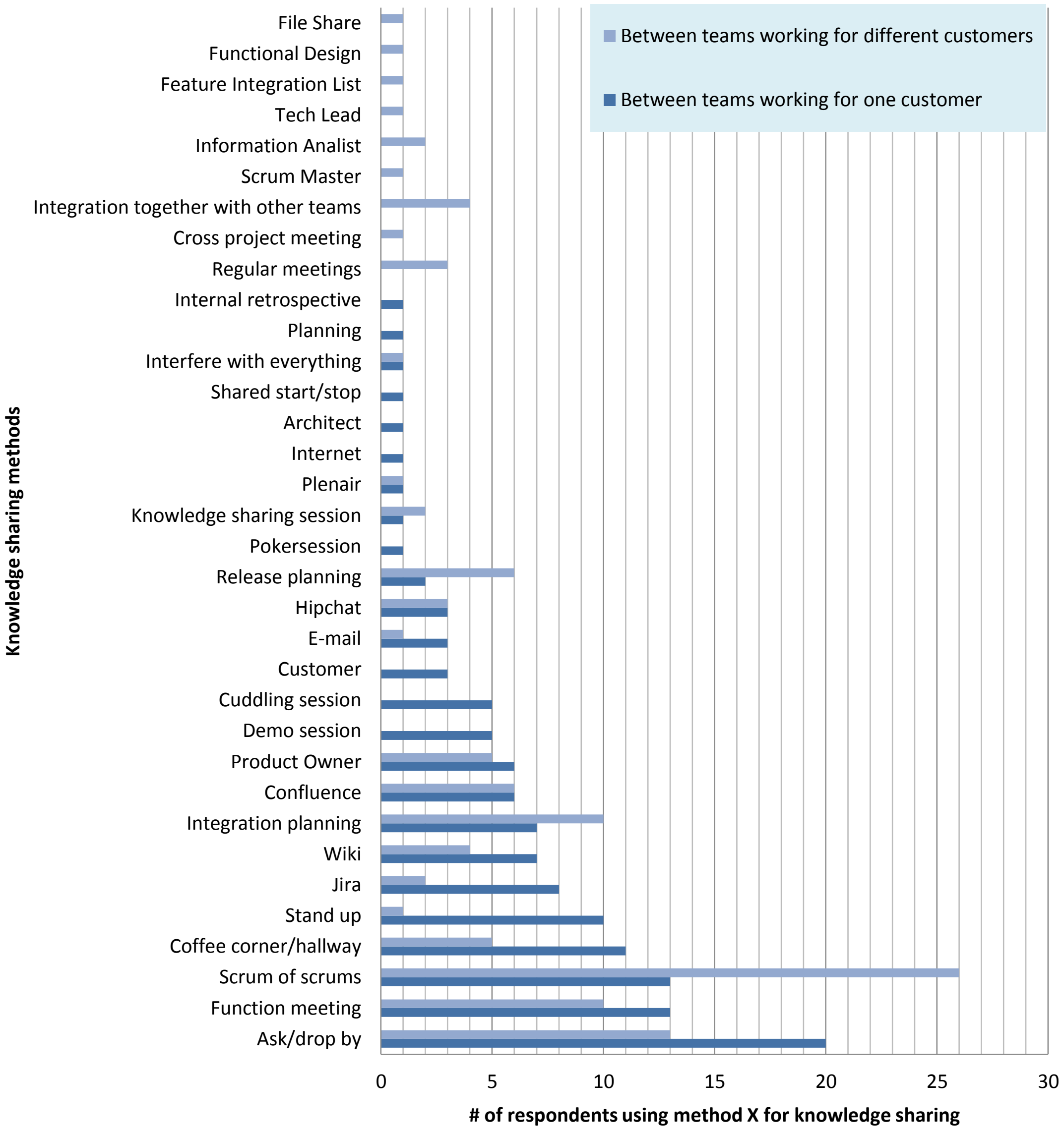
# 6 Results

*After the testing of the frameworks, in this chapter the results are shown and discussed. In the first section the results of the preliminary research are given. In the second part the results are given from after the experiment. Based on the outcomes of the experiment and the observations during this research option for improvement are given. In the last section the validity of this research is discussed.*

## 6.1 Pre-Experiment

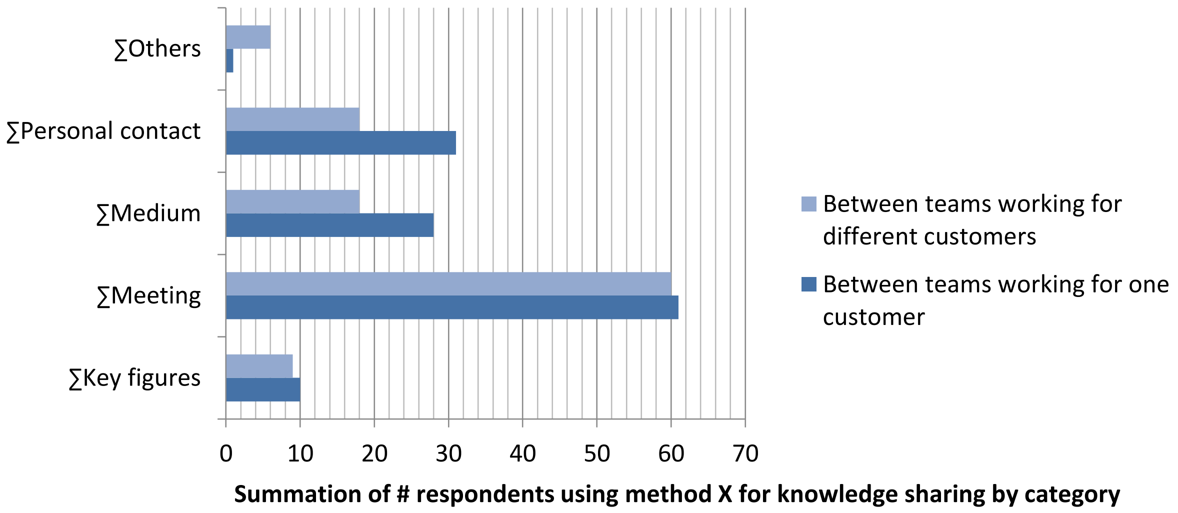
The pre-test questionnaire had a response rate of 80% (44 responses) and all teams are evenly represented. The Product D teams are currently organised in the following way. The teams exist of a Scrum Master, Business Analysts, Developers and Testers. The Customer Y and Customer X teams have their full Topicus teams located in Deventer, but are also part of combined customer/Topicus teams. The Customer X and Pakket teams are located on the same floor in the same wing and share the same pantry. The two Customer X teams are mixed with Customer X employees, every week part of the teams visit the customers' site and vice versa.  Additionally the Customer X teams in Deventer have, next to their individual stand-up, a shared stand-up meeting.  Also one of the Customer X teams has video conference stand-up with their team members of Customer X located at their own office. Furthermore, Customer X has a so called cuddling session where the teams share what they will work on in the next Sprint. At Pakket team Rapido has a daily afternoon stand-up to see whether they are on the right track to meet the goals which was set in the morning. The Customer Y teams are located at the other side of the building. The people working for the Customer Y product are in combined teams with employees from Customer Y. For this reason the people of Topicus are located at the customers' site on the Tuesday, Wednesday and Thursday. On the Mondays and Fridays video stand-ups are used.

Some of the information sharing methods were already mentioned above. An entire overview of the results of the questionnaire on which methods team members use to share information are shown in Graph 1.  The results show that some ways of knowledge sharing are more popular among the teams working for the same customer, and others are more popular among the teams working for different customers. A big difference is in the Scrum of Scrums (26 vs 13). This meeting is more valued as a method of knowledge sharing with teams working  for other customers than the teams working for the same customer. The people working for the same customer directly ask their colleagues in case they need information more often (13 vs 20). The stand-ups are (1 vs 10) mentioned more frequently as a source of gaining knowledge of the other teams working on the same customer. As these results are the result of an open question, it could be the case when this question was asked in a multiple choice style that it resulted in different answers. For example certain knowledge sharing techniques could have been chosen more frequently.



**Graph 1: Knowledge sharing methods**

The different knowledge sharing methods depicted in Graph 1 are clustered by roles, meetings, media, personal contact and others. The result of the clustering is given in Graph 2. In this graph the frequencies of the different knowledge sharing methods which can be grouped under one theme are added up (see Appendix C – Clustering Knowledge Sharing Methods). It shows that meetings are most frequently used to share information. Personal contact came in at a second place. Personal contact was more often used (18 vs 31) within the teams working for one customer than between the different customer teams. Third was medium (e.g. Hipchat, Confluence) with almost a similar difference between the different classifications of between teams working for different customers and within teams working for the same customer (18 vs 28). Key figures play an equal role when it comes to knowledge sharing.



**Graph 2: Knowledge sharing methods - clustered**

Some of the knowledge sharing methods represented in Graph 1 could be unfamiliar for some of the readers. Therefore an explanation is given in de sections below.

The meetings exist of Scrum Master meetings which are held every other week. Additionally every week there are Product Owner meetings. The different functions have their own meetings as well, such as the business analyst meetings and tester meetings every other week. Testers meet more frequently than every other week, but in those meetings not all testers are involved. In case issues arise or new projects are set up, people organise themselves to develop. Scrum of Scrum meetings are also done, however they do not have the desired effect since the follow up is sometimes lacking. In case people run in to a topic which needs attention, they form a group with other people who are interested in that topic as well and try to improve it. This is comparable to Communities of Practice. The cuddling session, where the teams tell each other what they will work on for the next Sprint, is specific for the Customer X teams.

Confluence is a platform for team collaboration. Documents can be stored and information can be shared. There is the possibility to organise information in categories, for instance per team or per project. Additionally questions can be posted on Confluence so that other colleagues can answer those. This way is preferred over informal watercooler Q&A because when questions are logged in Confluence this information is available for the entire organisation. Confluence is the latest addition at Topicus to stimulate knowledge sharing. It is supposed to store information in a more centralised manner and to replace all the other ways which cause the information to be stored in a dispersed way.

Hipchat, is a chat box where the entire Topicus Finance department has access to. In this chat box people talk to each other individually or group chats can be formed.  Hipchat, Jira and Confluence are all tools from Atlassian.

Jira is a tool to support project management, an overview is given of the tasks which need to be done and by whom. Additionally the velocity of the teams can be checked, as well as the burndown chart. Jira is linked to Hipchat, employees are able to set different rules in Jira in what case notifications in Hipchat should be given.

The Wiki is a platform where information can be stored and shared. The Wiki will be replaced by Confluence.

Additionally I observed there is face to face contact between the teams. This is usually among the teams working for the same customer. Although the Pakket teams sometimes contact the Customer X teams to help them out. The Pakket and Customer X teams are located on the same floor in the same wing of the building. The Customer Y team works at the  other side of the building and is therefore also less involved with the other teams. Furthermore, a part of the Customer X teams are working at the Customer X office one day a week. The Customer Y teams are located at the Customer Y office three days a week. Moreover the culture within each team differs. Some teams are calm and quiet, others are playing music and enjoying a table football game every once in a while.

Moving on with the results from the closed questions. First the knowledge among the customer teams are discussed (Table 6). Second we take a look at what the different customer teams know of the other customer teams (Table 7). The numbers given in Table 6 and Table 7 are averages based on a score of one to six. The distributions of these scores can be found in Appendix D.

The Customer X teams mentioned they know best what the other Customer X team is doing on average they scored an 4.9 on a scale of one to six. Additionally the Customer X teams mention they have sufficient knowledge of each other's teams to perform (5.3). This may be caused by their shared stand-ups and cuddling sessions. Pakket gave an 3.8 score about how much they know about the other Pakket teams. A possible cause might be that two of the four Pakket teams do not directly develop the PRODUCT D product, but are busy with the architecture and components. Although the score about what the teams know from each other is relatively low, this does not limit them in doing their daily tasks (5). Customer Y scores an average of 4.1 regarding knowledge of the other Customer Y teams. Moreover the employees working in the Customer Y teams score a 4.6 on whether they have enough information of the other Customer Y teams to perform well.

| | I know what other teams of my customer are doing | I have sufficient knowledge of my customers' other teams to do my work properly |
|---|---|---|
| Customer X | 4.9 | 5.3 |
| Pakket | 3.8 | 5.0 |
| Customer Y | 4.1 | 4.6 |

**Table 6: Knowledge sharing among customer teams**

When comparing the average scores on what the teams know about the teams of the other customers the results were less positive. Customer X again scored highest (3.3) on what they know about the other teams. However Customer X scored slightly positive (4) that they know sufficient to do their job. Pakket with 3.1 scores second on what they know about the other customer teams. With a 4.2 score on whether the knowledge is sufficient they scored the highest. Customer Y had the lowest score (2.5) on how much they  know about the other teams. On whether Customer Y teams know sufficient they scored a 3.7. The low score of Customer Y might be explained by the fact that the Customer Y teams are located on the other side of the building, as well as that the Customer Y teams are at the office of the customer three days a week.

| | I know what the teams of the other customers are doing | I have sufficient knowledge of the teams of the other customers to do my work properly |
|---|---|---|
| Customer X | 3.3 | 4.0 |
| Pakket | 3.1 | 4.2 |
| Customer Y | 2.5 | 3.7 |

**Table 7: Knowledge sharing between customer teams**

## 6.2 Post-Experiment

This section is about the post-experiment results. The questionnaire held after the experiment had a sample size of 51. The response rate was 65%. The sample size was smaller than with the pre-experiment questionnaire because people left the company. Furthermore, the new employees were not part of the sample group since they do not know about the "before" situation.  The response rate might be lower than with the pre-experiment questionnaire, since some team members were on vacation. Table 8 and Table 9 show the results of the level of knowledge sharing among and between the different teams. The numbers are averages on a scale of one to six. To make it easier to compare, the average numbers of the pre-experiment questionnaire are placed in the tables underneath in parentheses. The distribution of these numbers can be found in Appendix F.

| | I know what other teams of my customer are doing | I have sufficient knowledge of my customers' other teams to do my work properly |
|---|---|---|
| Customer X | (4.9) 5.3 | (5.3) 5.4 |
| Pakket | (3.8) 4.5 | (5.0) 4.8 |
| Customer Y | (4.1) 4.4 | (4.6) 4.6 |

**Table 8: Knowledge sharing among customer teams**

As can be seen in Table 8, for all teams there was progress in what they know about the other teams working for the same customer. However, the experiments were aimed at knowledge sharing between the teams working for different customers. In case of the Customer X teams a possible explanation for this improvement could be that the two teams have been merged during the experiment. Regarding the Pakket and Customer Y teams no plausible explanation could be found for the change in numbers. The numbers about whether the knowledge the teams have is sufficient, approximately stayed the same, which is according to the expectations.

| | I know what the teams of the other customers are doing | I have sufficient knowledge of the teams of the other customers to do my work properly |
|---|---|---|
| Customer X | (3.3) 3.6 | (4.0) 4.7 |
| Pakket | (3.1) 3.1 | (4.2) 4.3 |
| Customer Y | (2.5) 3.4 | (3.7) 3.9 |

**Table 9: Knowledge sharing between customer teams**

In Table 9 the results of the two experiments combined are shown. The results of the pre-experiment questionnaire and post-experiment questionnaire show a minor difference. It is therefore hard to say if the change in the level of knowledge and whether is knowledge is sufficient, is significant.

### 6.2.1 Chickens

Based on the questionnaire and the conversations with team members the experiences regarding the Chicken experiment differ. 22% of the respondents mentioned that they did not experience any advantages of the experiment. Arguments for this were that the Scrum of Scrums already provides the teams with the same information. Moreover, it was mentioned that the stand-ups are quite cryptic as an outsider, and therefore it was difficult learn something of the other teams. There was nothing organised to share the experiences of the Chicken within their own team, so not much is done with the new information. Another 31% did see the benefits of the Chickens, although only on process level, not the technical details since stand-ups are too short to understand the technical aspects. Moreover, it was mentioned that Chickens can be useful to see on a global level what other teams are doing, so when information is needed, the teams know better where to get this information. It was also mentioned that when joining a stand-up of a team where the employee or team share a dependency, useful information can be obtained from joining the stand-up. One of the employees mentioned, that with little effort big results can be achieved. 47% of the respondents were neutral about the experiment, this is because they either did not participated in the Chickens experiment or they did not experience any positive or negative effect about the Chickens. The feedback which was given is about the implementation of the Chickens. Team members indicated that they did not always know their dependencies. Although at the time the Chicken experiment was introduced, the teams were told to make an Environment Map, in order to get to know their dependencies. However, these Environment Maps were never created since teams were too busy. It was said that the Chickens were too voluntarily. This caused that not everybody felt the urge to be a Chicken.  However, according to the Scrum methodology and Agile philosophy teams are supposed to be autonomous. Therefore I expected that the teams would show initiative themselves and arrange to be a Chicken.  Additional feedback was that more guidance should have been provided.

### 6.2.2 Travellers

Based on the feedback conversation and the questionnaire, the two teams who participated in the Travellers experiment, overall had a positive experience. The people of Pakket gained more understanding of the people working in the Customer Y teams, since they have to visit the office of the customer two to three days a week. Moreover, technical knowledge is increased as well, since Customer Y works on a different part of the product. On the other hand, for the Customer Y team it was useful to have somebody available from the Pakket team to share their view. Because currently it happens, that the Customer Y teams build what works best for Customer Y but do not take into account whether this will work for Pakket as well. As a response to this observation a meeting with the Product Owner and Architect was planned to see how such issues can be solved. Having somebody with a fresh perspective within the team, who questions your way of working, helps the other team to improve or rethink why they do things in a certain way. From the Customer Y perspective is was very useful having an extra developer in the team to be able to discuss the work. Since normally this Customer Y team has only one developer. This positive outcome is possibly not directly related to the Travellers experiment. This effect was probably caused by having a peer to share ideas with. It is likely that this result would not have been there in case this Customer Y team had multiple developers in place. The availability of one developer at the Customer Y team, was next to the time constraint a reason why nobody of this Customer Y team joined the Pakket team. Both teams mentioned that due to this experiment the barrier to ask for information at the other team is lower. During the feedback meeting, the exchange employee of Pakket mentioned that the Customer Y teams function as islands, they lack a connection with the other teams. Moreover the Topicus people of the Customer Y team do not have their own stand-up meeting and retrospective, because they do the Scrum meetings together with the combined team. However information which is only relevant for the Topicus part of the team is not shared. Based on this feedback the Customer Y team started with their own stand-ups. The Pakket employee noticed that there is less interaction between the team members of Customer Y compared to her own team. The other team members of the Customer Y team acknowledge this, and suggest to increase communication and keep each other more up to date about their own tasks. The Customer Y team members noticed the critical feedback from an outsider is very useful as a confirmation of their assumptions. Based on this feedback additional actions can be taken to improve the team. One of the Customer Y team members said that the mobility between the projects is too low, and that the Management Team should do something about it, for instance by Travellers, so employees can be put to work in different environments.

Next to the work related experiences, the Traveller experiment also contributed to the personal growth of the Pakket employee. She mentioned changing teams was exciting, she was a bit nervous about what to expect from working at another team at another location. Also she felt a little insecure whether she would be able to do the work of the Customer Y team. So this entire experiment gave her more confidence to join new experiences in the future. She felt that one Sprint is long enough to get an idea of what the other team is doing.

Table 10 shows the results of the questionnaire. Pakket - CG and Customer Y - CG are the control groups for the Traveller experiment. This means that these so called CG teams only participated in the Chicken experiment. The results from the Chickens are also included in these numbers. The numbers between the parentheses are of the pre-experiment questionnaire. The numbers include what the teams know about all the teams working for the other customer.

| | I know what the teams of the other customers are doing | I have sufficient knowledge of the teams of the other customers to do my work properly |
|---|---|---|
| Pakket - CG | (3.0) 3.5 | (4.3) 4.2 |
| Pakket - Traveller | (3.3) 2.7 | (3.9) 4.3 |
| Customer Y - CG | (2.6) 3.3 | (3.8) 4.0 |
| Customer Y - Traveller | (2.3) 3.7 | (4.0) 3.7 |

**Table 10: Knowledge sharing between customer teams – Travellers**

The open questions in the questionnaire and the evaluation meetings with the Traveller teams indicated a much more positive outcome of the Travellers experiment than the numbers in Table 10 depict. Regarding the Pakket teams, the Traveller team shows a decrease in what they know about the other teams. This might be because of “the more you know the less you think you know” which is the opposite of the Dunning-Kruger effect (Kruger & Dunning, 1999). Although the Pakket Traveller team does show a bigger increase than the other Pakket teams in whether their knowledge of the other teams is sufficient to do their work properly. Considering the Customer Y teams it is hard to conclude if there were any significant changes in the level of knowledge of the other teams based on the numbers of the questionnaire. The Traveller teams shows a bigger increase in the amount of knowledge they have of the other teams, compared to the other Customer Y teams. However, the level of knowledge the Customer Y teams have to do their work properly, diminished. This could be caused by the fact that the Customer Y Traveller team is now aware that they used to think about the Customer Y specifications only, and not take into account what kind of effect this could have on the Pakket teams.

Suggestions of the two teams for further implementation is to make the suggestion for being a Traveller during a performance interview. Also instead of exchanging two people at the same time, a buddy system is suggested, where an employee of one team joins one person with the same role at another team for one Sprint. The next Sprint vice versa. A Management Team member indicated that Travellers can be planned far ahead so teams are able to prepare for the temporary change.

## 6.3 Options for further improvement

The options will be discussed on different levels. First the recommendations on the Chicken and Traveller experiments are given. Next, advice is given on knowledge sharing in general. Furthermore, there are some additional recommendations which came across during this research which are not directly related to knowledge sharing. At last the scientific recommendations will be given.

Because of the feedback on the Chicken experiment. The suggestion is to retry the Chicken experiment but in a stricter format. This means, first letting the teams and their Product Owners define their dependencies with other teams (e.g. environment mapping). After which they can decide for whom it might be useful to join a specific stand-up. Additionally there should be some time available during the stand-up for the people who went to another team to Chicken to share their experience. Furthermore a frequency standard must be set, for instance everybody should Chicken once every Sprint.

Although the idea of the Travellers concept in itself was simple. The execution and organisation of the exchange were not as easy. Since the teams will need to cut back a little in their productivity to share information, it is good to plan in advance so the teams can find a way to cope with the team member exchange. Based on this experiment I want to suggest that in order to make it successful, a few steps need to be taken in to account when in the starting phase of this Travellers concept.

- Exchange team members who voluntarily choose to join
- Plan one to two Sprints in advance
- Choose a topic which is easy to get started on (if possible)
- Let somebody from team A join team B for one Sprint, afterwards change
    - The people who switch places preferably have the same function (i.e. developer)
- After the exchange plan an evaluation with both teams to share experiences on wider level

Especially in the starting phase it is important to have volunteers participate, since obligated exchange does not fit with the Agile philosophy. Once a few successful exchanges have taken place other team members might get enthusiastic as well. Additionally, start with exchanging people within the same business line since there are more similarities and the people do know each other, people will feel less fearful and will more easily join the exchange. After several successful cases this can be expanded by exchanging people between different business lines. By planning several Sprints in advance the teams can plan their work in such a way that the impact on the possible decrease of productivity is minimal. This experiment showed that when a project or task which is still in the starting phase it is easier for the person joining, to understand the topic and is more likely to deliver a contribution. The original set up for the Travellers is that two team members are exchanged at the same time. However during the experiment is experienced that a set up where two people become buddies and join each other their teams for two Sprints is more valuable for knowledge exchange. It is advisable to link two team members which the same role (i.e. developer) so not only knowledge is shared on process level but also on technical level. Although the experiment was done with the exchange of developers, the other functions can also benefit of an exchange. During the exchange the other team members indirectly learn from the situation. However at the end of the two Sprints it is wise to take some time to share the experiences between the teams and take action where needed.

According to the teams, the right people to initiate the Travelling are the Scrum Masters and Product Owners, because they have a broader view on what other teams are doing. Although additional support from the Management Team is needed. A performance interview is an appropriate timing to discuss whether it will be beneficial for the personal development of a team member to be a Traveller for two Sprints.

It should be kept in mind that Travellers and Chickens are means to the goal of knowledge sharing. The Travellers and Chickens should not be seen as a goal in itself. Since LeSS offers other coordination methods which are focused on knowledge sharing these can be tried as well. Of these methods communication in code is not recommended since not everybody is able to read and write code. Also Town Hall is not recommended because this is too voluntarily and is therefore not likely to improve the coordination between the teams. I would recommend to next start with the increase of shared space, since this is also mentioned by the team members as a necessity. Furthermore I would like to suggest to try Joint Sprint Review Bazaar and Open Space, because this allows people to efficiently share information for those who are interested.

Next to the recommendations related to the experiment, some other recommendations can be made based on the replies of the questionnaires and the interviews. First of all multiple people indicated that the translation of the strategy and vision of Topicus Finance on an operational level is missing. Additionally the Customer Y teams are located at the other side of the building, this is too far for easy and convenient knowledge sharing. Especially since the Customer Y teams are not in Deventer for the majority of the week. I would suggest looking for opportunities to locate them closer to the other teams. Furthermore the composition of the Customer Y teams has changed too often, this had influence on the team morale. It would be better if the Customer Y teams are more stable and cohesive teams can be formed. Due to time constraints the focus for this research was Agile Scaling on team level and specifically on knowledge sharing. However Agile Scaling is more than knowledge sharing. In order to implement Agile Scaling in the entire organisation, I would recommend looking at SAFe® since this framework takes all different levels of the organisation into account. SAFe® requires Topicus to include the company strategy and vision on product level, which was indicated as one of the desires of the Product Owners and Scrum Masters. Additionally SAFe® provides structure to the integration and releases of software with the Agile Release Train. Company B has implemented some good practices on how to manage working on different projects, this include prioritisation of the different epics, features, stories and tasks as well as distribution of the work across the different teams. This way of planning can be beneficial for Topicus as well. In order to have all the Agile Scaling practices work, it is advised to get more standardisation across the teams. This means the teams should have the same start/stop day. Additionally all teams should work of the same Product Backlog. Similar quality should be guaranteed by having a shared Definition of Done and Definition of Ready.

## 6.4 Validity

In this research there were several threats to validity. These are split in internal and external threats to validity.  Internal validity is concerning to what extend the outcome of the dependent variable is caused by the independent variable rather than by another (unknown) variable (Brewer, 2000). External validity is regarding the generalisability of the study to other situations and other people (Shadish, Cook, & Campbell, 2002).

At least the following threats exist to the internal validity; selection bias, history, attrition and diffusion.  The selection bias was caused by the fact that the teams for the Travellers experiment are selected by availability. The team members directly involved in the switch were volunteers. Additionally during the Chicken experiment the Chickens were also on voluntary bases. In case of this research this could not have been prevented, since Scrum and Agility is about support of the employees. History also played a role. During the experiment team compositions have changed. The two Customer X teams were merged into one team. Furthermore some ad hoc circumstances within the teams, which required more focus on their regular work, caused that less focus was put on the Chicken experiment. Attrition was a result of people leaving the company and therefore were not able to finish the experiment. Additionally people were on holiday an therefore not able to finish the questionnaire and take part in the (entire) experiment. Diffusion could have been the case during the Travellers experiment. This happens when the teams which are not directly involved in the Travellers experiment  are influenced by the experiences of the experiment. This may have caused that the control group also shows improvement in their level of knowledge. Diffusion could have been prevented by separating the different Customer Y teams, and not allowing them to communicate, however, this was not possible since this communication between the teams is needed to properly execute their job.

The threat to external validity for the Travellers experiment was caused by the sample size of the experiment. Since only two teams were involved in this experiment it is hard to generalise the results.

The Travellers have only been tested for one Sprint. It was planned to be longer, but due to the external circumstances the start has been postponed several times. In fact the completion of this thesis was extended with four weeks so as to include the first results.

Regarding the questionnaire of the post-experiment, it seems like the closed questions about how the two experiments have influenced certain aspects, was not clear. This resulted in the fact that the majority of the people answered  these questions with non-applicable, even though the described situation was applicable and therefore less relevant answers were collected. Furthermore, the questions about how much the teams know about the other teams working for a different customer, might have resulted in more specific data when the question was posed differently. This different option would be that the same question is asked, but the question should be answered per different customer separately. In this way it is more likely that the numbers give a better reflection on the outcome of the experiment.

Because the two experiments took place at the same time it is hard to justify which experiment has caused which outcome. In the future, the experiments should be done by different sample groups or at different points in time. Additionally, none of the Agile Scaling frameworks have scientific proof that they are successful. Therefore it is not possible to scientifically proof that the results of this research are valid.

*In conclusion, in this chapter the results from the experiments are discussed. The opinions about the success of the Chicken experiment varied. It was advised to retry this experiment but with more structure and guidance, so that it is less voluntarily. The Traveller experiment has showed positive results, and it was therefore recommended to continue with this method to improve knowledge sharing. Next to the recommendation about the experiments, additional recommendations were done regarding location of the teams, operationalisation of the strategy and vision and the usage of other Agile Scaling frameworks. Moreover the validity of this research has been discussed.*

# 7 Conclusions and Recommendations

*This is the final chapter of this research. The first section will provide brief answers to all the research questions. In Section 7.2, the main conclusions of this research will be given. The final section will give the recommendations.*

## 7.1 Answers to research questions

In order to achieve the research goal, several research questions were formulated. The research goal and brief answers to the research questions are given below.

*Propose, test and partly implement an Agile Scaling framework to improve the functioning of the Scrum teams working on PRODUCT D version 3*

1. *Which Agile Scaling problems are experienced within Topicus Finance?*

The scaling problems experienced can be categorised in product development, strategy and vision of PRODUCT D and execution of Scrum.

2. *What are the requirements for Topicus Finance for the Agile Scaling framework based on the problems experienced?*

The requirements for the Agile Scaling framework were that it should help to improve knowledge sharing between the teams, additionally it should be possible to test during the timeframe of this research and therefore provide quick results.

3. *Which different Agile Scaling methods exist and how are the different characteristics of the models interrelated.*

In this study 15 Agile Scaling frameworks have been found. This list has been reduced to the top three. This resulted in three frameworks, Large Scale Scrum (LeSS), Scaled Agile Framework® (SAFE®) and Disciplined Agile Delivery (DAD). These three frameworks were analysed and compared. In general LeSS its focus is on Agile Scaling at team level with focus on coordination between the development teams. SAFe® on the other hand, focusses on adopting Agile Scaling throughout the entire organisation from the top of the organisation to the team level at the bottom of the organisation. DAD is meant for the development of software in project format. This framework works with a clear start and end, with usage of Scrum. A few suggestions for scaling are done, but this is not the main goal of DAD. Additionally, information was obtained at Company B who already successfully adopted Agile Scaling practices. At Company B not one complete single framework was implemented. They have chosen different options which suits their organisation.

4. *Which (part) of these models or a combination of these models can be applied to the current situation in such a way that it will improve the current process?*

Based on the requirement and the analysis of the three different Agile Scaling frameworks the choice was made to adopt practices from LeSS. LeSS offers several coordination methods which enhance knowledge sharing. Of these eleven coordination methods the Chickens and Travellers were chosen. With the Chickens people from the teams visited each other's stand-ups. With the Travellers team members were exchanged between the teams for a Sprint.

5. *Is the proposed model valid for Topicus Finance?*

It is not possible to draw conclusions regarding the validity of this research, since the sample size for the Travellers was one, and the execution of the Chicken experiment was not as intended.

## 7.2 Main conclusions

This research shows that, of the three researched frameworks, there is no single Agile Scaling framework which offers a complete solution for all the challenges faced by Topicus. As part of this research we did a case study at Company B Department C, where we observed that not one entire framework had been implemented, but several different practices are used to fit their specific context. Based on the two examples of Topicus and Company B it is likely that there is not one single answer to the Agile Scaling challenges which companies face. Therefore in the future companies should clearly investigate their Agile Scaling challenges, and select the practices from the frameworks which best fit their organisation.

Furthermore, the adoption of Agile Scaling frameworks requires changes in the way of working of the employees as well as in the structure of the organisation. Although the employees acknowledged that change is needed and they were involved in the decision making, the actual change was challenging. So knowledge from the change management field is very useful for the adoption of Agile Scaling frameworks. While, there were some difficulties in implementing the Travellers. However at the end of the experiment the team members and especially the Product Owner were enthusiastic about the Travellers method and its results.

Additionally, the chosen Large Scale Scrum (LeSS) practice, Travellers, could not be implemented in the exact way as described by the literature. The idea of exchanging teams was kept, but the shape of the Travellers concept had been altered to fit the context of Topicus.

Next to the implied effect of knowledge sharing, the Travellers experiment also lead to mutual respect and understanding for the differences between the internal and external customer teams. Also, this experiment lowered barriers between the two teams, and team members indicated that it is more likely to approach each other in case help is needed. Lastly, the team member of Pakket who had joined the Customer Y team, noticed personal growth by stepping out of her comfort zone.

## 7.3 Recommendations

The practical recommendations for Topicus are split in what should be done now, in the near future and in the long run. At last recommendations for the scientific community are given.

### 7.3.1 Now

The recommendation on the experiments is to repeat the Chicken experiment but with more structure and guidance to see whether this will result in enhanced knowledge sharing, since according to the team members this was lacking. This experiment could probably have given better results in case there were stricter rules and the dependencies were clear beforehand. This means that first an Environment Map should be created by the Product Owners together with (some of ) the team members to get the dependencies of the teams visible. Based on this Environment Map the teams can together with their Scrum Masters and/or Product Owners, plan who needs to visit which stand-up. Also during the stand-up time should be created, so that the Chicken can share the new information with his/her team members. Agreements should be made on how often team members have to Chicken, I would suggest each team should visit another stand-up twice a week. Furthermore, trying the Chickens for another six weeks with this stricter format should give an indication whether this concept is beneficial or not. In case positive results are lacking, the Chickens should not be continued. It should be kept in mind that these practices are means to a goal, and not the goal itself.

The Travellers should be continued in the format used with the experiment. Only that somebody of team A joins team B for a Sprint and the next Sprint vice versa. Instead of one team member joining another team for one Sprint, as done in the experiment. During the evaluation the Travellers indicated that a two-way exchange will probably be most beneficial for optimal knowledge sharing. Team members can either initiate Travelling themselves, or the coaches of the team members can suggest during a coaching session, that is might be useful for a person to widen their horizon by being a Traveller. These coaches are usually the Scrum Masters. During the bi-weekly Scrum Master meeting, the Scrum Masters can announce which team member they have available for Travelling and an appropriate exchange can be arranged this way so it fits with the schedule of the teams. For designers, an exchange between PRODUCT D teams will probably not be beneficial to gain technical knowledge since there is only one team which has designers. However it might be useful for the other teams if a designer joins their team, so they can learn to incorporate the designers point of view during their work, which will result in a better looking product according to one of the designers. For technical knowledge exchange the designer can do exchanges with designers at other business lines. This also counts for the architects, because they already work together and are not spread across teams. The exchange of people between business lines is a bit more challenging since this concept is not (yet) introduced. I want to suggest that the Management Team takes responsibility for introducing the Travellers concept at the Scrum Masters and Product Owners of the other business lines, so they can look for Traveller opportunities within their teams as well.

So briefly it can be stated that the Travellers experiment needs to be executed in the following way.

- Exchange team members who voluntarily choose to join
- Plan one to two Sprints in advance
- Choose a topic which is easy to get started on (if possible)
- Let somebody from team A join team B for one Sprint, afterwards change
    - The people who switch places preferably have the same function (i.e. developer)
- After the exchange plan an evaluation with both teams to share experiences on wider level

### 7.3.2 Short-term

LeSS also offers other coordination methods which are focused on knowledge sharing these can be tried as well. Of these methods communication in code is not recommended since not everybody is able to read and write code. Also Town Hall is not recommended because this is too voluntarily and is therefore not likely to improve the coordination between the teams.

I would recommend to next start with the increase of shared space, which is a LeSS practice, since this is also mentioned by the team members as a necessity. This means that the Customer Y teams have to move at least to the same side of the building as where the other PRODUCT D teams are located. Currently there is no space available. Other business lines should be requested if they are willing to move, so the PRODUCT D teams can be located closer to each other.

Furthermore, I would like to suggest to try Joint Sprint Review Bazaar. This can be done in addition to the team their individual Sprint Review. During this bazaar each team requires their own space where they can explain and demonstrate what they have been doing the previous Sprint. Just like a regular fair people walk around and stop by to ask questions and give feedback. By coordinating the Sprint Review this way, only the necessary information can be shared making it more efficient than the demos held previously. For this set up a big area is needed such as the canteen or VK1 Ada Lovelace (meeting room). Although the LeSS literature does not provide a timeframe for this concept, I think a session of one hour should be sufficient to share information.

Next I would like to recommend using Open Space. This is a meeting where team member have the opportunity to share their concerns with people of other teams. At the start of the meeting people share what they want to discuss, and a schedule like the one given in Table 11 is created. Each timeslot lasts 20 minutes. By organising the meeting like this team member have the opportunity to efficiently contribute to knowledge sharing by joining the meetings which are useful.

|  | Area 1 | Area 2 | Area 3 |
|---|---|---|---|
| **Time slot 1** | Topic A | Topic B | Topic C |
| **Time slot 2** | Topic D | Topic E | Topic F |
| **Time slot 3** | Topic G | Topic H | Topic I |

**Table 11: Example Open Space planner**

Especially in the beginning it will be useful if a Scrum Master or Product Owner takes the lead in this meeting. How often this meeting should be done is not given in the literature, but I would start with every Sprint. After several Sprints this session should be evaluated to decide if and how to continue with this meeting.

In order to make joint meetings possible, all the teams need to have their start/stop day on the same day. Since some of the Topicus teams are combined with customer teams, this might be a challenge. The Scrum Masters or Product Owners should try to negotiate with the customer teams whether it is possible to shift the start/stop day.

### 7.3.3 Long-term

In the long run, more uniformity and standardisation across teams is needed in order to be more flexible with the planning of work, which was mentioned by the Management Team. When teams are more flexible the occupancy of teams are likely to increase. This means teams should have the same start/stop day as described in the previous section. But this also means that there should be one single Product Backlog for the entire PRODUCT D product. Additionally, all teams should have the same quality standard, this can be achieved by having a standardised Definition of Ready and a Definition of Done. Since teams have gained more knowledge by having adopted the practices explained in the previous sections, teams are more flexible in what work to do.

When there is one Product Backlog and teams have the same start/stop day, it is possible to start using the Sprint Planning as done by Company B. How Company B executes their Sprint Planning is described in Section 3.5.1.1. Furthermore I would suggest to contact one of the cluster managers of the Company B Department C to share more information on how they have adopted this way of planning. By having flexible teams and working of one Product Backlog the Sprints can be planned more effective.

As indicated by the Product Owners, Scrum Masters and team members there is a demand for a translation of the strategy and vision towards a more operational level, so decisions regarding the development of the product can be made so it supports the strategy and vision of the company. LeSS focuses on practical coordination tools on team level. SAFe® on the other hand takes the entire organisation in to account. SAFe® describes Agile Scaling on team, portfolio and program level. Therefore LeSS and SAFe® complement each other. So when the coordination on team level is successful, the next step to bring Agile Scaling to a higher level could be done by SAFe®. SAFe® uses the strategy, vision and roadmap of the company as input for the Product Backlog. Since SAFe® is large and complex framework I would like to recommend looking for a consultancy organisation to help to adopt this framework at Topicus.

### 7.3.4 Scientific

During this research I noticed scientific proof of the success of the different Agile Scaling frameworks is lacking. For the majority of these frameworks cases are available which can be used for further research to get scientific proof of these frameworks. Furthermore, additional research should be done on what framework is best in which kind of situations. So, when organisations have defined their Agile Scaling challenges, it is easier to select the most suitable framework.

## References

Accelebrate. (2013). *Agile Training: Agile with Scrumban*. Retrieved from Accelebrate: https://www.accelebrate.com/training/agile-scrumban

Agile42. (2015). *The Enterprise Transition Framework*. Retrieved from agile42: http://www.agile42.com/en/agile-transition/etf/

AgiliX. (2013). *Overview - Wat is Scrum?* Retrieved from agilix: http://agilix.nl/agilixblog/wat-is-scrum

Ambler, S. (2009). The Agile Scaling Model (ASM): Adapting Agile Methods for Complex Environments. *IBM Corporation*, 1-35.

Ambler, S., & Lines, M. (2010). Disciplined Agile Delivery: The Foundation for Scaling Agile. *CrossTalk*, 7-11.

Babinet, E., & Ramanathan, R. (2008). Dependency Management in a Large Agile Environment. *IEEE Computer Society*, 401-406.

Barton, B. (2007). Establishing and Maintaining Top to Bottom Transparency Using the Meta-Scrum. *Agile Journal*, 1-5.

Beck, K., Beedle, M., Bennekum, A. v., Cockburn, A., W.Cunningham, Fowler, M., . . . Thomas, D. (2001). *Manifesto for Agile Software Development*. Retrieved from agile manifesto: http://www.agilemanifesto.org/

Beck, P., Gärtner, M., Mathis, C., Roock, S., & Schliep, A. (2015). *ScALeD Agile Lean Development - The Principles*. Retrieved from scaled principles: http://scaledprinciples.org/

Beedle, M. (2015). *Enterprise Scrum- Executive Summary.* Agile Management for the 21st Century.

Berends, H., van Aken, J., & van der Bij, H. (2012). *Problem Solving in Organizations.* New York: Cambridge University Press.

Bird, C., & Davies, R. (2007). Scrum Gathering. *Scaling Scrum.* London.

Boyer, K., & McDermott, C. (1999). Strategic consensus in operations strategy. *Journal of Operations Management*, 289-305.

Brewer, M. (2000). Handbook of Research Methods in Social and Personality Psychology. In H. Reis, & C. Judd, *Handbook of Research Methods in Social and Personality Psychology* (p. 13). New York: Cambridge University Press.

Burm, D. (2013, August 12). *New Scrum Process Overview - Update.* Retrieved from blog.xebia: http://blog.xebia.com/2013/08/12/new-scrum-process-overview-update/

Chow, T., & Cao, D.-B. (2008). A survey study of critical success factors in agile software projects. *The Journal of Systems and Software*(81), 961-971.

Cohn, M. (2007, May 7). *Advice on Conducting the Scrum of Scrum Meeting*. Retrieved from Scrum Alliance: https://www.scrumalliance.org/community/articles/2007/may/advice-on-conducting-the-scrum-of-scrums-meeting

Continious Agile. (2015, February 2). *Continuous Enterprise: Matrix of Services*. Retrieved from Unblock! A guide to the New Continious Agile: http://www.continuousagile.com/unblock/ea_matrix.html

Dolman, R., & Spearman, S. (2014). *Links*. Retrieved from Agile Scaling: http://www.agilescaling.org/links.html

DSDM Consortium. (2015). *DSDM*. Retrieved from DSDM: http://www.dsdm.org/journey-so-far

Franken, M. (2013). Hoofdstuk 14 - Scrum met meerdere teams. In M. Franken, *Scrum voor Dummies* (pp. 157-162). Amsterdam: Pearson Benelux.

Franken, M. (2013). *Scrum voor Dummies.* Amsterdam: Pearson Benelux.

Goldman, S., Nagel, R., & Preiss, K. (1995). Agile Competitors and Virtual Organizations: Strategies for Enriching the Customer.

Greening, D. (2010). Enterprise Scrum: Scaling Scrum to the Executive Level. *2010 43rd Hawaii International Conference on Systems Sciences (HICSS)* (pp. 1-10). Honolulu: IEEE.

Harasymczuk, M., & Kniberg, H. (Directors). (2014). *Spotify Engineering Culture part 1 (Agile Enterprise Transition with Scrum and Kanban* [Motion Picture].

Harasymczuk, M., & Kniberg, H. (Directors). (2014). *Spotify Engineering Culture part 2 (Agile Enterprise Transition with Scrum and Kanban)* [Motion Picture].

Heikkilä, V., Paasivaara, M., Lassenius, C., & Engblom, C. (2013). Continious Release Planning in a Large-Scale Scrum Development Organization at Ericsson. In H. Baumeister, & B. Weber, *Agile Processes in Software Engineering and Extreme Programming* (pp. 195-209). Berlin: Springer-Verlag.

Henver, A., Salvatore, T., & Park, J. (2004). Design Science in Information Systems Research. *MIS Quarterly*, 75-105.

Joshi, M., Kathuria, R., & Porth, S. (2003). Alignment of strategic priorities and performance: an integration of operations and strategic management perspectives. *Journal of Operations Management*, 353-369.

Kester, J. v. (2013, April 25). *Scrum: ook geschikt voor kleine bedrijven & projecten*. Retrieved from Frankwatching: http://www.frankwatching.com/archive/2013/04/25/scrum-ook-geschikt-voor-kleine-bedrijven-projecten/

Kettunen, P., & Laanti, M. (2008, March). Combining Agile Software Projects and Large-scale Origanizational Agility. *Software Process Improvement and Practice*, 183-193.

Kniberg, H., & Ivarson, A. (2012). *Scaling Agile @ Spotify with Tribes; Squads, Chapters and Guilds.*

Kruger, J., & Dunning, D. (1999). Unskilled and Unaware of It: How Difficulties in Regocnizing One's Own Incompetence Lead to Inflated Sef-Assessments. *Journal of Personality and Social Psychology*(77(6)), 1121-1134.

Ladas, C. (2008). *Scrumban: Essays on Kanban Systems for Lean Software Development.* Seattle: Modus Cooperandi Press.

Larman, C., & Vodde, B. (2010). Chapter 6: Coordination. In C. Larman, & B.Vodde, *Practices for Scaling Lean & Agile Development: Large, Multisite and Offshore Product Development with Large-Scale Scrum* (pp. 189-214). Boston : Pearson Education, Inc.

Larman, C., & Vodde, B. (2014). *Large-Scale Scrum*. Retrieved from LeSS More with LeSS: http://less.works/

Leffingwell, D. (2007). *Scaling Software Agility.* Boston: Pearson Education.

Leffingwell, D. (2011). Chapter 3: Agile Requirements for the Team. In *Agile Software Requirements: Lean Requirement Practices for Teams, Programs and the Enterprise* (pp. 47-61). Boston: Pearson Education Inc.

Leffingwell, D. (2011). Chapter 4: Agile Requirements for the Program. In D. Leffingwell, *Agile Software Requirements: Lean Requirements Practices for Teams, Programs and the Enterprise* (pp. 63-82). Boston: Pearson Education Inc.

Leffingwell, D. (2015). *Scaled Agile Framework 3.0*. Retrieved from Scaled Agile Framework: http://www.scaledagileframework.com/

Lewitz, O. (2014, October 26). De-Scaling Your Organization and Using Temenos. (T. Charron, Interviewer)

Lewitz, O. (2015, January 28). *OOP Descale Your Organisation!* Retrieved from The Trust Artist: http://trustartist.com/2015/01/28/oop-descale-organisation/

Lindvall, M., Basili, V., Boehm, B., Costa, P., Dangle, K., Shull, F., . . . Zelkowitz, M. (2002). Emperical findings in Agile Methods. *Extreme Programming and Agile Methods - XP/Agile Universe*, 197-207.

Lines, M., & Ambler, S. (2015). *Disciplined Agile Delivery*. Retrieved from Disciplined Agile Delivery: http://disciplinedagiledelivery.com/

Lines, M., & Ambler, S. (2015, March 13). *Introduction to DAD.* Retrieved from Disciplined Agile Delivery: http://disciplinedagiledelivery.com/

Lines, M., & Ambler, S. (2015, March 13). *Large Agile Teams.* Retrieved from Disciplined Agile Delivery: https://disciplinedagiledelivery.wordpress.com/agility-at-scale/large-agile-teams/

Maranzato, R., Neubert, M., & Herculano, P. (2012). 2012 Agile Conference. *Scaling Scrum Step by Step: "The Mega Framework"* (pp. 79-85). Conference Publishing Services.

McNiff, J., & Whitehead, J. (2009). *Doing and Writing Action Research.* London: SAGE Publications Ltd.

Mountain Goat Software. (2015, July 23). *Sprint Review Meeting*. Retrieved from Topics in Scrum: https://www.mountaingoatsoftware.com/agile/scrum/sprint-review-meeting

Net Objectives. (2013). *Scrumban Resources*. Retrieved from Net Objectives: http://www.netobjectives.com/resources/scrumban

Oxford Dictionary. (2015, February 4). *British & World English*. Retrieved from Oxford Dictionaries: http://www.oxforddictionaries.com/definition/english/agile?searchDictCode=all

Paasivaara, M., & Lassenius, C. (2014). Communities of practice in large distributed agile software development organization - Case Ericsson. *Information and Software Technology*, 1556-1577.

Peltier, J. (Director). (2013). *From Scrum to Scrumban* [Motion Picture].

Qumer, A., & Henderson-Sellers, B. (2007). Construction of an Agile Software Product-Enhancement Process by Using an Agile Software Solution Framework (ASSF) and Situational Method Engineering. *IEEE* .

Ramos, C., & Roijakkers, S. (2015). *Scaling Scrum @ Company B with Emergent Innovation.* Scrum.org.

Ramos, C., & Roijakkers, S. (2015). Scaling Scrum @ Company B with Emergent Innovation. *Scrum Day Europe.*

Razzak, A., & Ahmed, R. (2014). Knowledge Sharing in Distributed Agile Projects: Techniques, Strategies and Challenges. *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems*, 1431-1440.

Richardson, I., Casey, V., O'Riordan, M., Meehan, B., & Mistrík, I. (2009). Knowledge Management in the Global Software Engineering Environment. *Fourth IEEE International Conference on Global Software Engineering*, 367-369.

Roijakkers, S. (2014, September 10). Company B - Sensors - TU Processing How we implement agile.

Rubin, K. (2013). Chapter 1 - Introduction. In K. Rubin, *Essential Srucm: A practical guide to the most popular agile process* (pp. 1-12). Pearson Education US.

Rubin, K. (2013). Chapter 2 - Scrum Framework. In K. Rubin, *Essential Scrum, A practical guide to the most popular agile process* (pp. 13-28). Pearson Education US.

Rus, I., & Lindvall, M. (2002, May/June). Knowledge Management in Software Engineering. *IEEE Software*, 26-38.

Salmela, H., Tapanainen, T., Baiyere, A., Hallanoro, M., & Galliers, R. (2015). IS Agility research: An assesment and future directions. *Twenty-Third European Conference on Information Systems.* Münster.

Schwaber, K. (2014). *Evidence-Based Change Framework*. Retrieved from EBMgt: http://www.ebmgt.org/Agility-Path-Framework

Schwaber, K. (2014). *The Ability Guide to Evidence-Based Change: Using Scrum to Transform Your Enterprise.* Scrum.org.

Scrum Inc (Director). (2014). *Scrum at Scale: Explained* [Motion Picture].

Scrum.org. (2014). *Agility Path: Continious Improvement. Competitive Advantage.* Scrum.org.

Sekitoleko, N., Evbota, F., Knauss, E., Sandberg, A., Chaudron, M., & Olsson, H. (2014). *Technical Dependency Challenges in Large-Scale Agile Software Development.* Rome: 15th International Conference on Agile Software Development.

Shadish, W., Cook, T., & Campbell, D. (2002). Experimental and Quasi-Experimental Designs for Generalized Causal Inference. Boston, New York: Houghton Mifflin Company.

Shao, B., Yin, P., & Chen, A. (2014). Organizing knowledge workProduct D for specified iterative software development tasks. *Decision Support Systems*, 59 (15-27).

Singh, G. P. (2013, July 3). *What is Scrum?* Retrieved from Agile and Related Methodologies: https://gurtejpsingh.wordpress.com/2013/07/03/what-is-scrum/

Singleton, A. (2014, August 24). *Agile 2013 appearance for "MAXOS - the new continuous agile"*. Retrieved from anysingleton: http://andysingleton.com/agile-2014-appearance-for-maxos-the-new-continuous-agile/?__hstc=174222397.ec8a3a65e8f39091bb3268c24c12e566.1412206673753.1412206673753.1412206673753.1&__hssc=174222397.1.1412206673753&__hsfp=3135150724

Solingen, R. v., & Rustenburg, E. (2012). *De kracht van Scrum: Een inspirerend verhaal over een revolutionaire projectmanagementmethode.* Amsterdam: Pearson Benelux.

Sutherland, J. (2001). Agile Can Scale: Inventing and Reinventing SCRUM in Five Companies. *Cutter IT Journal*, 5-11.

Sutherland, J. (2014). *scruminc*. Retrieved from scruminc: http://www.scruminc.com/

Takeuchi, H., & Nonaka, I. (1986). The New New Product Development Game. *Harvard Business Review*, 1-11.

Thompsom, K. (2013). *Recipes for Agile Governance in the Enterprise.* Foster City, Los Angeles: cPrime, Inc - Agile Practice Lead.

Toebes, J. (2014, August 6). *Scrumban*. Retrieved from Xebia: http://blog.xebia.com/2014/08/06/scrumban/

Tolstoy, L. (1900). *Pamphlets: Three Methods Of Reform.*

Topicus. (2015, February 3). *Finance*. Retrieved from topicus: http://topicus.nl/finance/

Topicus. (2015, February 3). *Product D Hypotheken*. Retrieved from Topicus: http://topicus.nl/finance/hypotheken/product/Product D-hypotheken

Topicus. (2015, February 3). *over topicus*. Retrieved from topicus: http://topicus.nl/over-topicus/

Trapani, K. (2014, June 4). *How to Develop Your Own Recipe for Agile Governance*. Retrieved from cprime: https://www.cprime.com/2014/06/how-to-develop-your-own-recipe-for-agile-governance/

Weelwright, S. (1984). Manufacturing Strategy: Defining the Missing Link. *Strategic Management Journal*(Volume 5), 77-91.

Wieringa, R. (2014). *Design Science Methodology for Information Systems and Software Engineering.* Berlin Heidelberg: Springer-Verlag Berlin Heidelberg.

Xebia. (2014). *Dutch Agile; Survery report 2014.* Xebia Agile Consulting.

## Appendix A – Interview Questions

Name

Function/Role

Responsible for which Scrum team(s)?

How is Scrum used within your team(s)?

Do you encounter any problems regarding usage of Scrum?

What aspects of Scrum are working really well?

How is the product management currently organised?

    How does the work/tasks reach your team?
    Are the values of the stories known/is it known why certain stories are important?

How is your involvement with other teams?

    Can you give examples?
    What do you know about other teams?
    How do you know about the other teams?

What are the positive/negative aspects of the current “system”?

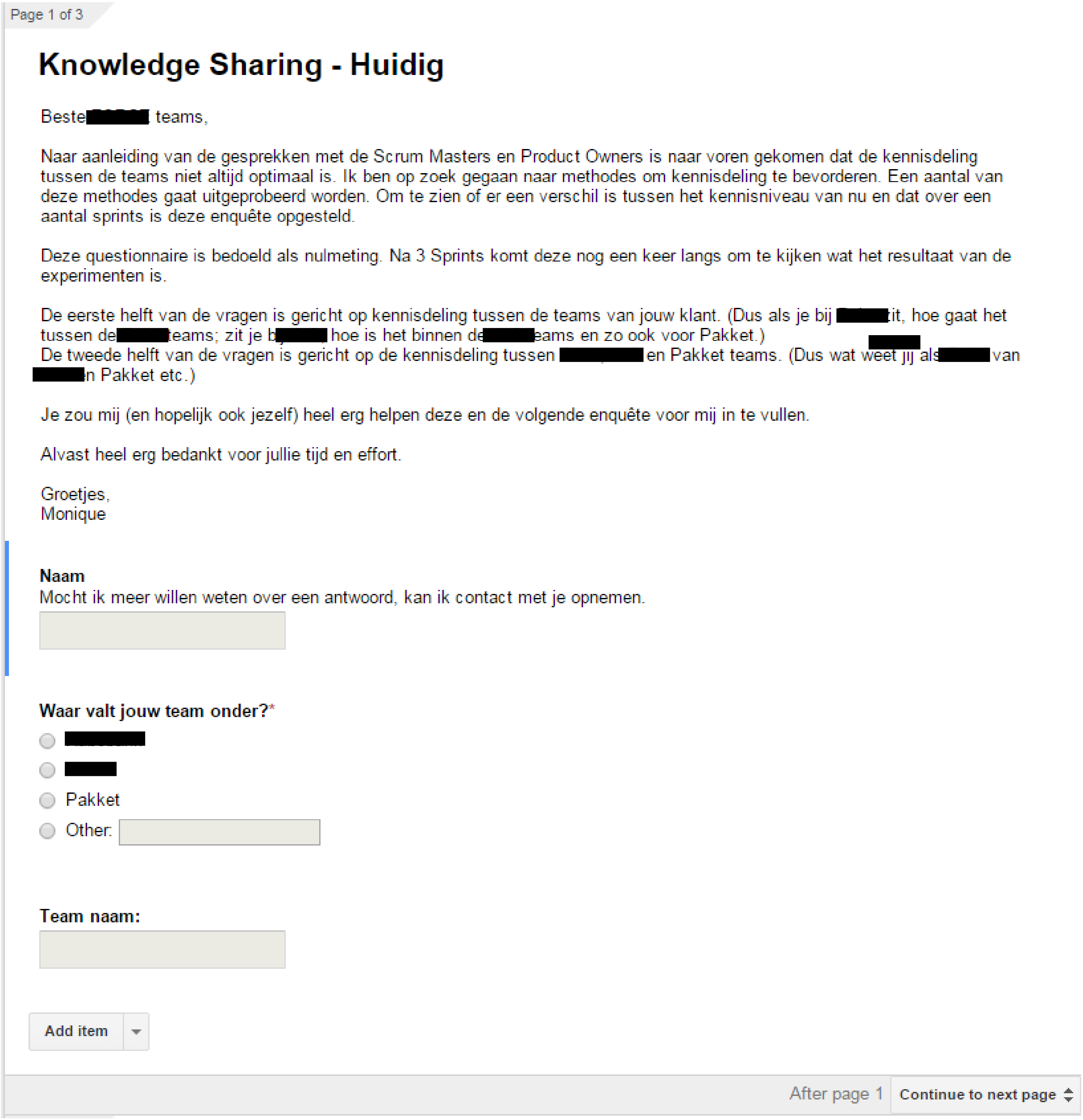
What is most important for improved product/portfolio management according to you?

What is an absolute no-go within a new product/portfolio management framework?

Do you have any suggestions for my further research?

Are you willing to help me more often?

# Appendix B – Questionnaire Pre-Experiment



Page 1 of 3

## Knowledge Sharing - Huidig

Beste [redacted] teams,

Naar aanleiding van de gesprekken met de Scrum Masters en Product Owners is naar voren gekomen dat de kennisdeling tussen de teams niet altijd optimaal is. Ik ben op zoek gegaan naar methodes om kennisdeling te bevorderen. Een aantal van deze methodes gaat uitgeprobeerd worden. Om te zien of er een verschil is tussen het kennisniveau van nu en dat over een aantal sprints is deze enquête opgesteld.

Deze questionnaire is bedoeld als nulmeting. Na 3 Sprints komt deze nog een keer langs om te kijken wat het resultaat van de experimenten is.

De eerste helft van de vragen is gericht op kennisdeling tussen de teams van jouw klant. (Dus als je bij [redacted]it, hoe gaat het tussen de [redacted]teams; zit je b[redacted] hoe is het binnen de[redacted]eams en zo ook voor Pakket.) [redacted]
De tweede helft van de vragen is gericht op de kennisdeling tussen [redacted] en Pakket teams. (Dus wat weet jij als[redacted] van [redacted]n Pakket etc.)

Je zou mij (en hopelijk ook jezelf) heel erg helpen deze en de volgende enquête voor mij in te vullen.

Alvast heel erg bedankt voor jullie tijd en effort.

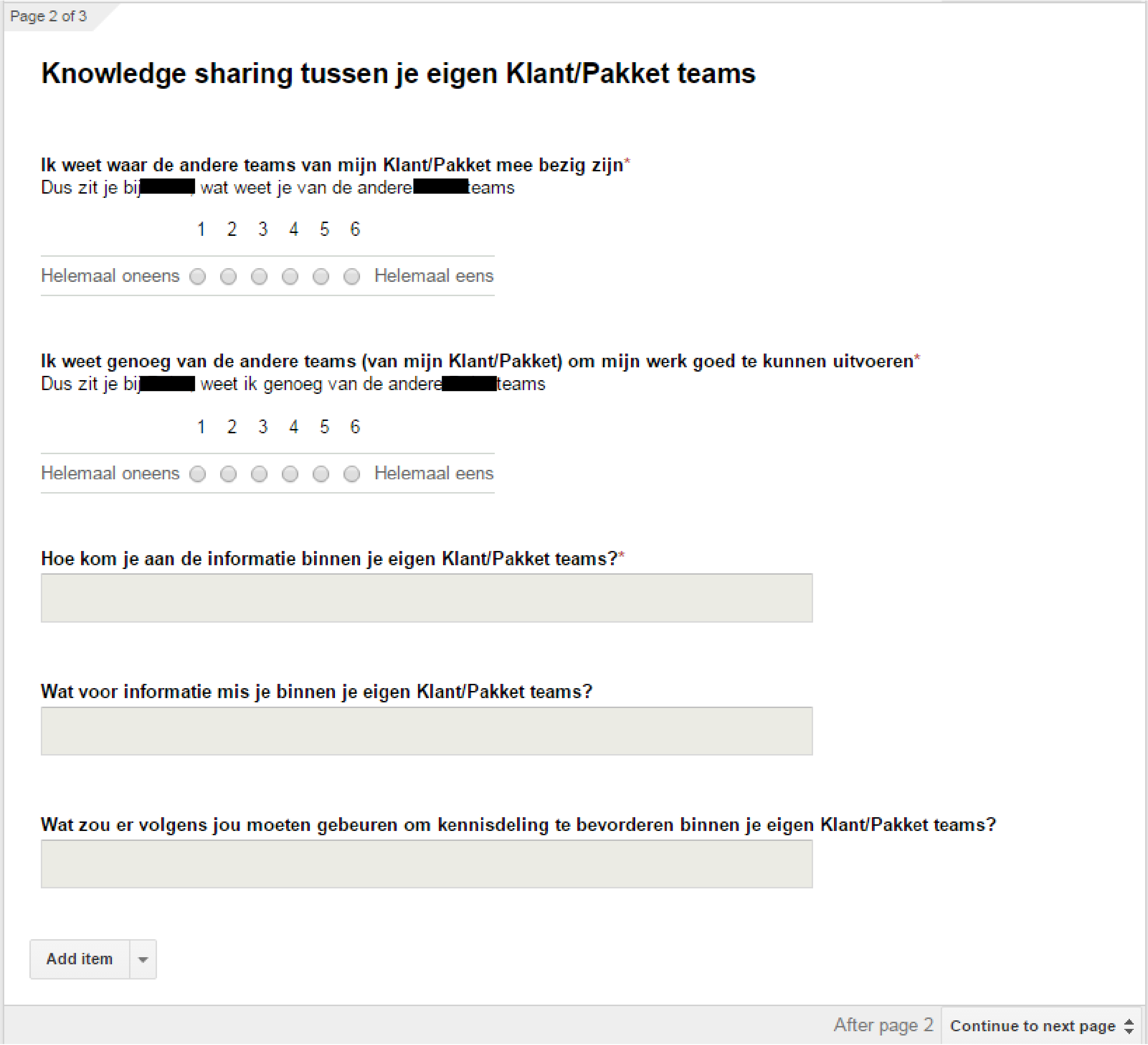
Groetjes,
Monique

**Naam**
Mocht ik meer willen weten over een antwoord, kan ik contact met je opnemen.

**Waar valt jouw team onder?***

○ [redacted]
○ [redacted]
○ Pakket
○ Other:

**Team naam:**



Page 2 of 3

## Knowledge sharing tussen je eigen Klant/Pakket teams

**Ik weet waar de andere teams van mijn Klant/Pakket mee bezig zijn***
Dus zit je bij[redacted] wat weet je van de andere[redacted]eams

| | 1 | 2 | 3 | 4 | 5 | 6 | |
|---|---|---|---|---|---|---|---|
| Helemaal oneens | ○ | ○ | ○ | ○ | ○ | ○ | Helemaal eens |

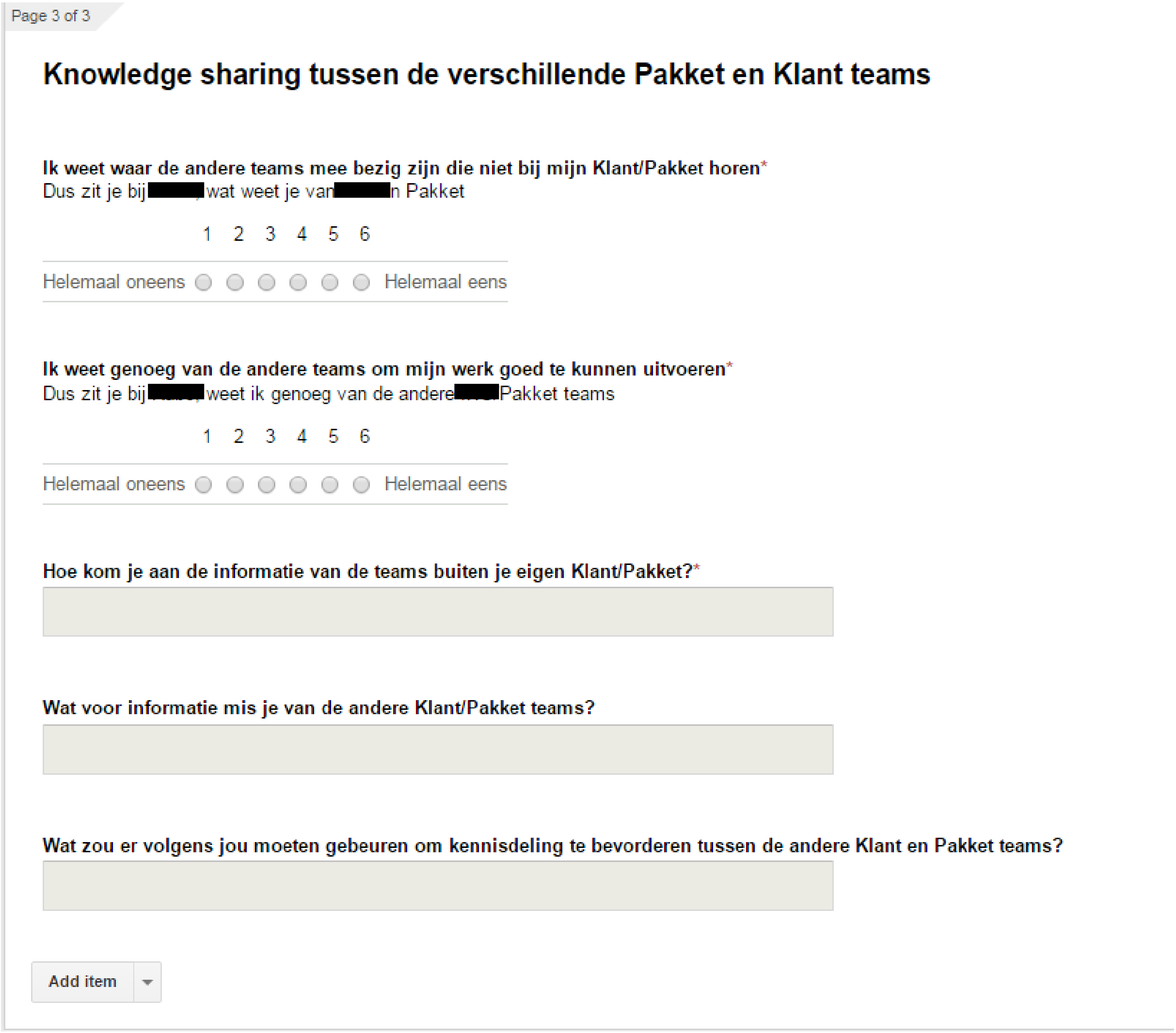
**Ik weet genoeg van de andere teams (van mijn Klant/Pakket) om mijn werk goed te kunnen uitvoeren***
Dus zit je bij[redacted] weet ik genoeg van de andere[redacted]teams

| | 1 | 2 | 3 | 4 | 5 | 6 | |
|---|---|---|---|---|---|---|---|
| Helemaal oneens | ○ | ○ | ○ | ○ | ○ | ○ | Helemaal eens |

**Hoe kom je aan de informatie binnen je eigen Klant/Pakket teams?***

**Wat voor informatie mis je binnen je eigen Klant/Pakket teams?**

**Wat zou er volgens jou moeten gebeuren om kennisdeling te bevorderen binnen je eigen Klant/Pakket teams?**



Page 3 of 3

## Knowledge sharing tussen de verschillende Pakket en Klant teams

**Ik weet waar de andere teams mee bezig zijn die niet bij mijn Klant/Pakket horen***
Dus zit je bij [redacted] wat weet je van [redacted] Pakket

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | |
| Helemaal oneens | ○ | ○ | ○ | ○ | ○ | ○ | Helemaal eens |

**Ik weet genoeg van de andere teams om mijn werk goed te kunnen uitvoeren***
Dus zit je bij [redacted] weet ik genoeg van de andere [redacted] Pakket teams

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | |
| Helemaal oneens | ○ | ○ | ○ | ○ | ○ | ○ | Helemaal eens |

**Hoe kom je aan de informatie van de teams buiten je eigen Klant/Pakket?***

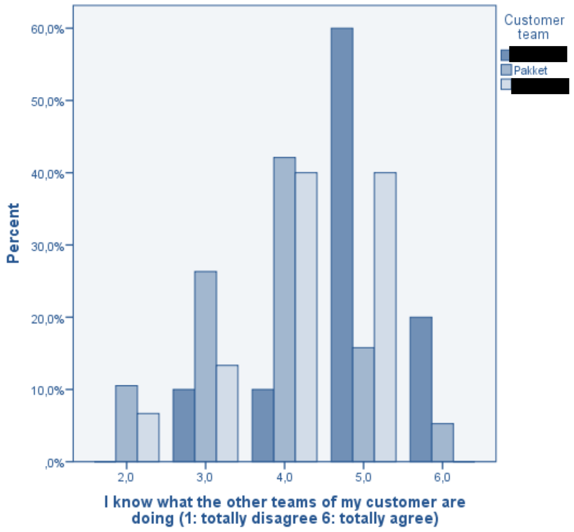
**Wat voor informatie mis je van de andere Klant/Pakket teams?**

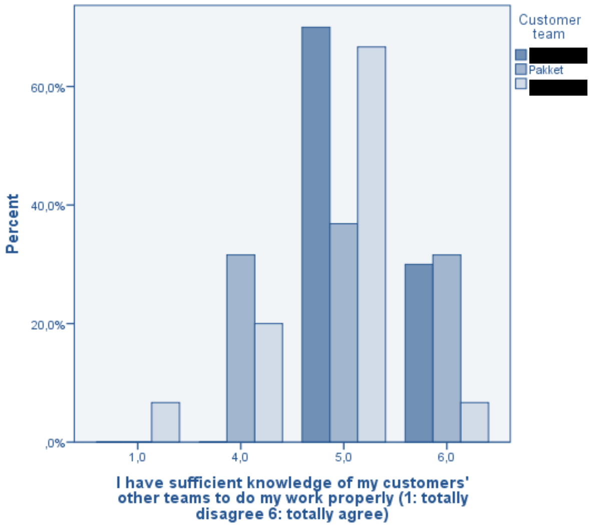
**Wat zou er volgens jou moeten gebeuren om kennisdeling te bevorderen tussen de andere Klant en Pakket teams?**

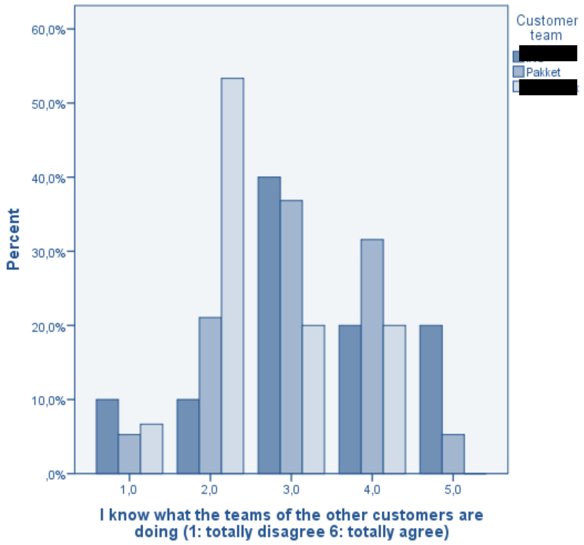
## Appendix C – Clustering Knowledge Sharing Methods

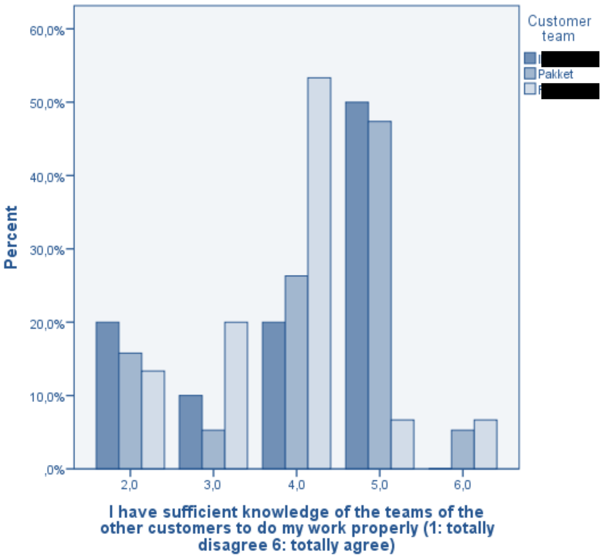
| | **Within teams working for 1 customer** | **Between teams working for different customers** |
|---|---|---|
| **∑Key figures** | **10** | **9** |
| Product Owner | 6 | 5 |
| Customer | 3 | 0 |
| Architect | 1 | 0 |
| Scrum Master | 0 | 1 |
| Information Analist | 0 | 2 |
| Tech Lead | 0 | 1 |
| **∑Meeting** | **61** | **60** |
| Function meeting | 13 | 10 |
| Scrum of scrums | 13 | 26 |
| Stand-up | 10 | 1 |
| Integration planning | 7 | 10 |
| Demo session | 5 | 0 |
| Cuddling session | 5 | 0 |
| Release planning | 2 | 6 |
| Pokersession | 1 | 0 |
| Knowledge sharing session | 1 | 2 |
| Plenair | 1 | 1 |
| Shared start/stop | 1 | 0 |
| Planning | 1 | 0 |
| Internal retrospective | 1 | 0 |
| Regular meetings | 0 | 3 |
| Cross project meeting | 0 | 1 |
| **∑Medium** | **28** | **18** |
| Jira | 8 | 2 |
| Wiki | 7 | 4 |
| Confluence | 6 | 6 |
| E-mail | 3 | 1 |
| Hipchat | 3 | 3 |
| Internet | 1 | 0 |
| Feature Integration List | 0 | 1 |
| File Share | 0 | 1 |
| **∑Personal contact** | **31** | **18** |
| Ask/drop by | 20 | 13 |
| Coffee corner/hallway | 11 | 5 |
| **∑Others** | **1** | **6** |
| Functional Design | 0 | 1 |
| Interfere with everything | 1 | 1 |
| Integration together with other teams | 0 | 4 |

## Appendix D – Distribution Pre-Experiment Closed Questions

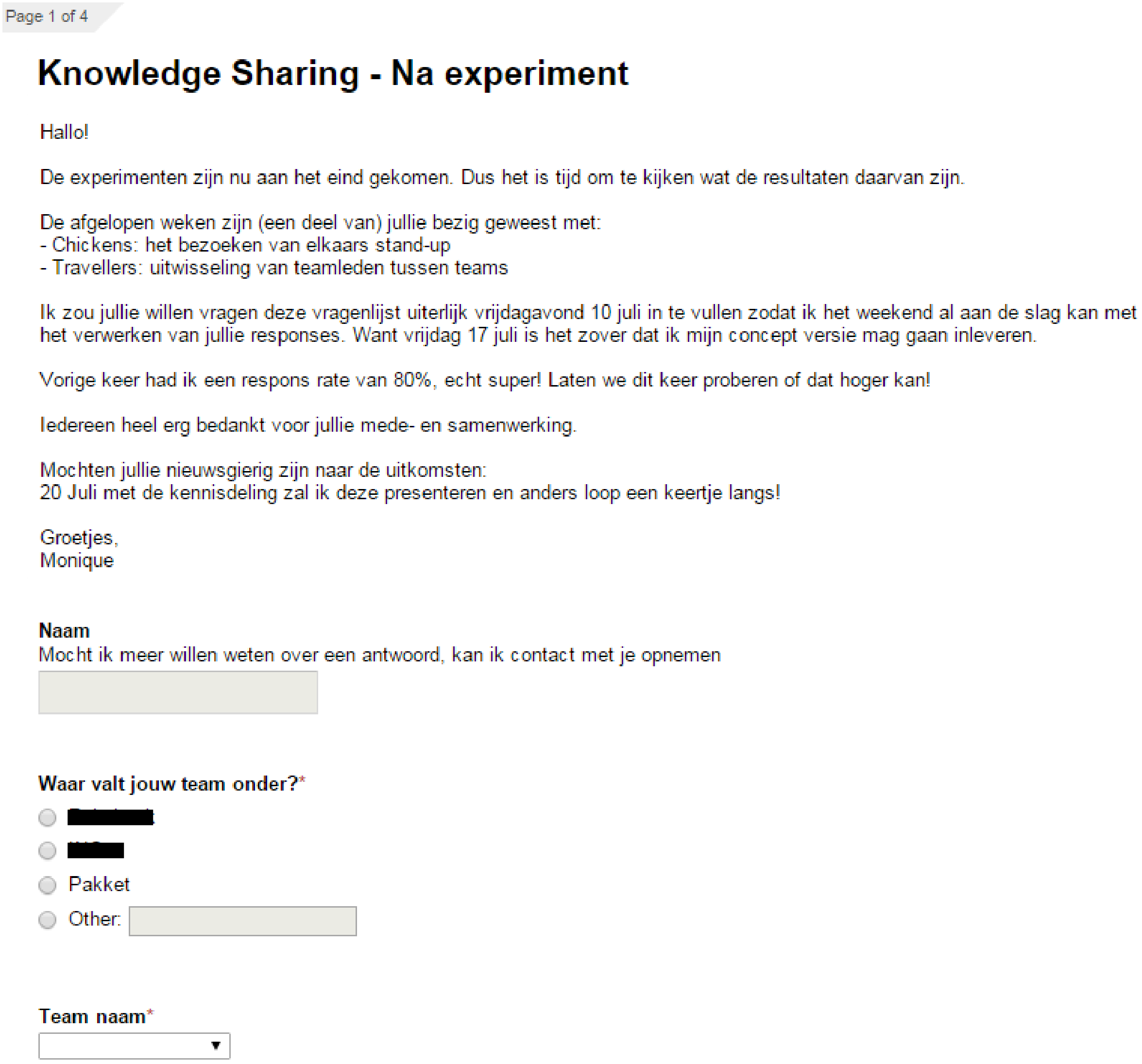








# Appendix E – Questionnaire Post-Experiment



Page 1 of 4

## Knowledge Sharing - Na experiment

Hallo!

De experimenten zijn nu aan het eind gekomen. Dus het is tijd om te kijken wat de resultaten daarvan zijn.

De afgelopen weken zijn (een deel van) jullie bezig geweest met:
- Chickens: het bezoeken van elkaars stand-up
- Travellers: uitwisseling van teamleden tussen teams

Ik zou jullie willen vragen deze vragenlijst uiterlijk vrijdagavond 10 juli in te vullen zodat ik het weekend al aan de slag kan met het verwerken van jullie responses. Want vrijdag 17 juli is het zover dat ik mijn concept versie mag gaan inleveren.

Vorige keer had ik een respons rate van 80%, echt super! Laten we dit keer proberen of dat hoger kan!

Iedereen heel erg bedankt voor jullie mede- en samenwerking.

Mochten jullie nieuwsgierig zijn naar de uitkomsten:
20 Juli met de kennisdeling zal ik deze presenteren en anders loop een keertje langs!

Groetjes,
Monique

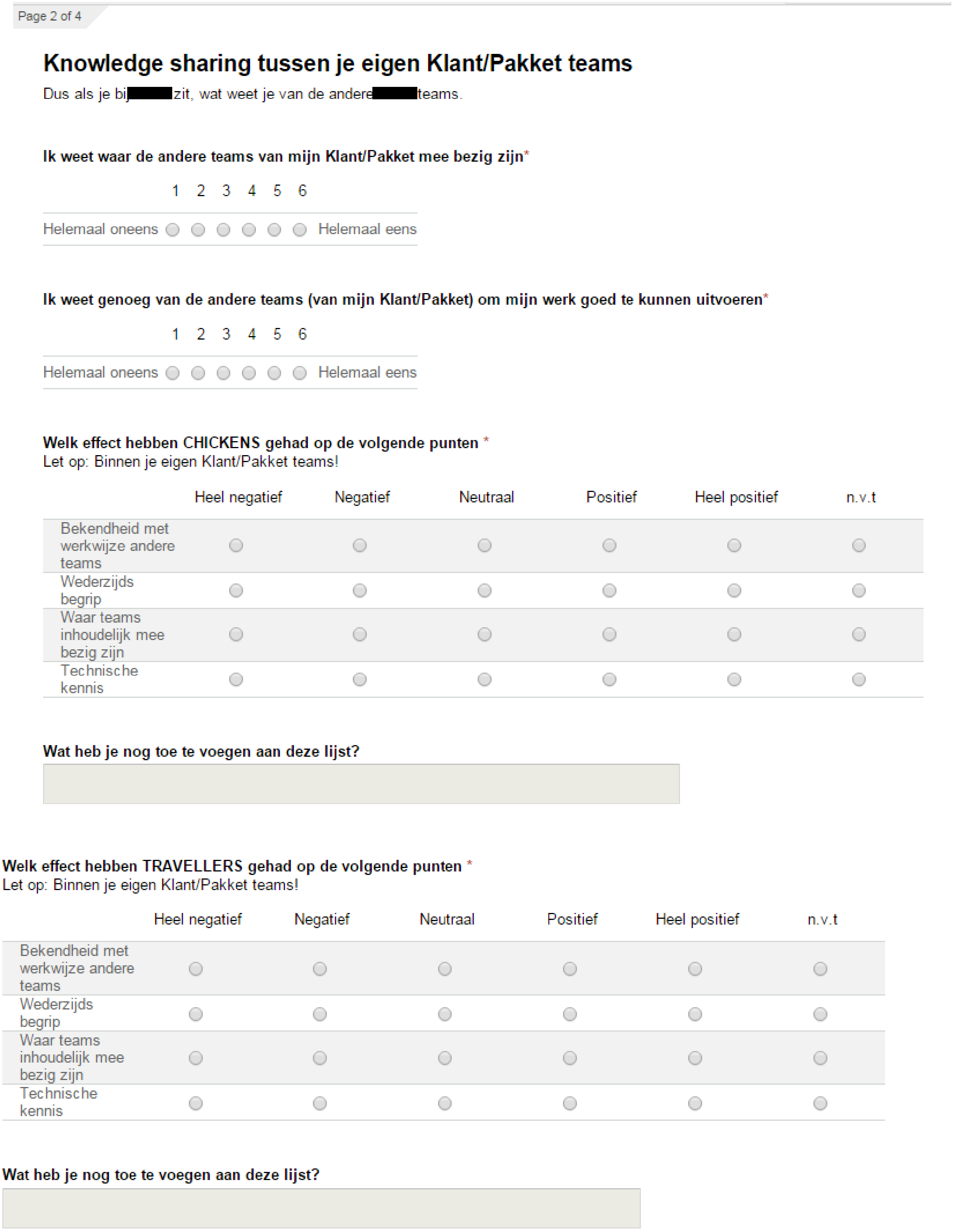
**Naam**
Mocht ik meer willen weten over een antwoord, kan ik contact met je opnemen

**Waar valt jouw team onder?***

○ [redacted]
○ [redacted]
○ Pakket
○ Other:

**Team naam***

▼



Page 2 of 4

## Knowledge sharing tussen je eigen Klant/Pakket teams

Dus als je bij [redacted] zit, wat weet je van de andere [redacted] teams.

**Ik weet waar de andere teams van mijn Klant/Pakket mee bezig zijn***

|  | 1 | 2 | 3 | 4 | 5 | 6 |  |
|---|---|---|---|---|---|---|---|
| Helemaal oneens | ○ | ○ | ○ | ○ | ○ | ○ | Helemaal eens |

**Ik weet genoeg van de andere teams (van mijn Klant/Pakket) om mijn werk goed te kunnen uitvoeren***

|  | 1 | 2 | 3 | 4 | 5 | 6 |  |
|---|---|---|---|---|---|---|---|
| Helemaal oneens | ○ | ○ | ○ | ○ | ○ | ○ | Helemaal eens |

**Welk effect hebben CHICKENS gehad op de volgende punten** *

Let op: Binnen je eigen Klant/Pakket teams!

|  | Heel negatief | Negatief | Neutraal | Positief | Heel positief | n.v.t |
|---|---|---|---|---|---|---|
| Bekendheid met werkwijze andere teams | ○ | ○ | ○ | ○ | ○ | ○ |
| Wederzijds begrip | ○ | ○ | ○ | ○ | ○ | ○ |
| Waar teams inhoudelijk mee bezig zijn | ○ | ○ | ○ | ○ | ○ | ○ |
| Technische kennis | ○ | ○ | ○ | ○ | ○ | ○ |

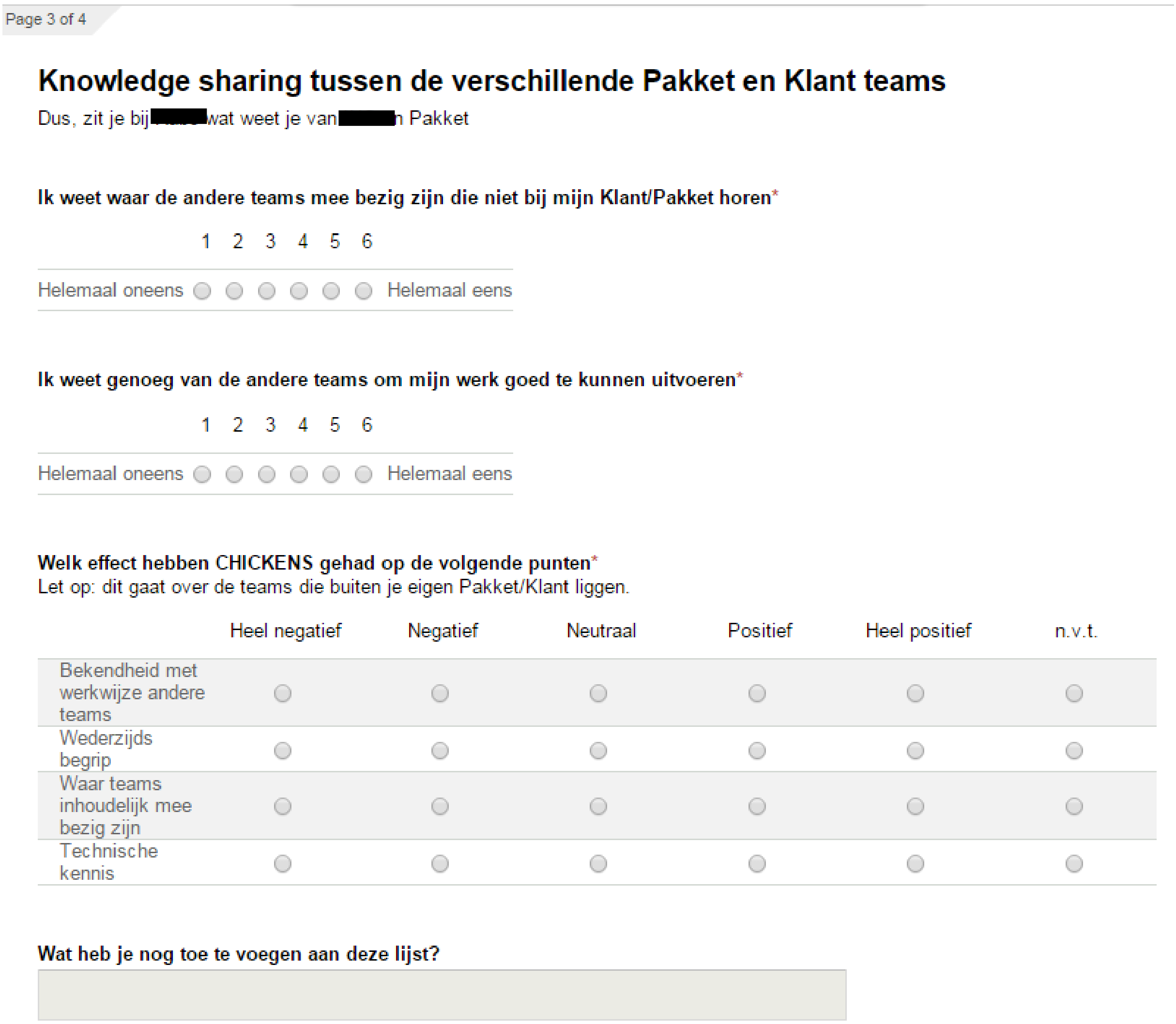
**Wat heb je nog toe te voegen aan deze lijst?**

**Welk effect hebben TRAVELLERS gehad op de volgende punten** *

Let op: Binnen je eigen Klant/Pakket teams!

|  | Heel negatief | Negatief | Neutraal | Positief | Heel positief | n.v.t |
|---|---|---|---|---|---|---|
| Bekendheid met werkwijze andere teams | ○ | ○ | ○ | ○ | ○ | ○ |
| Wederzijds begrip | ○ | ○ | ○ | ○ | ○ | ○ |
| Waar teams inhoudelijk mee bezig zijn | ○ | ○ | ○ | ○ | ○ | ○ |
| Technische kennis | ○ | ○ | ○ | ○ | ○ | ○ |

**Wat heb je nog toe te voegen aan deze lijst?**



Page 3 of 4

## Knowledge sharing tussen de verschillende Pakket en Klant teams

Dus, zit je bij [redacted] wat weet je van [redacted]n Pakket

**Ik weet waar de andere teams mee bezig zijn die niet bij mijn Klant/Pakket horen***

| | 1 | 2 | 3 | 4 | 5 | 6 | |
|---|---|---|---|---|---|---|---|
| Helemaal oneens | ○ | ○ | ○ | ○ | ○ | ○ | Helemaal eens |

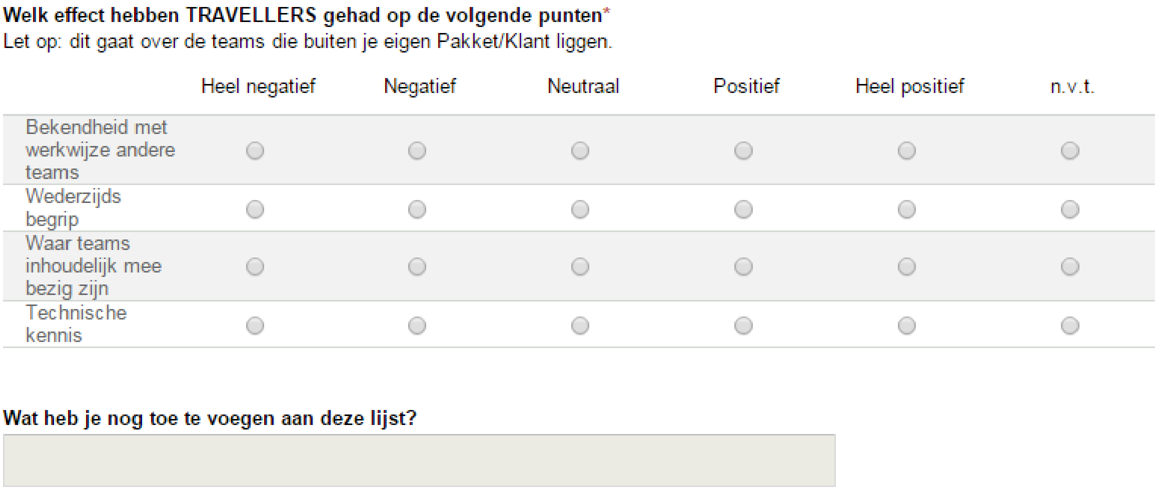
**Ik weet genoeg van de andere teams om mijn werk goed te kunnen uitvoeren***

| | 1 | 2 | 3 | 4 | 5 | 6 | |
|---|---|---|---|---|---|---|---|
| Helemaal oneens | ○ | ○ | ○ | ○ | ○ | ○ | Helemaal eens |

**Welk effect hebben CHICKENS gehad op de volgende punten***
Let op: dit gaat over de teams die buiten je eigen Pakket/Klant liggen.

| | Heel negatief | Negatief | Neutraal | Positief | Heel positief | n.v.t. |
|---|---|---|---|---|---|---|
| Bekendheid met werkwijze andere teams | ○ | ○ | ○ | ○ | ○ | ○ |
| Wederzijds begrip | ○ | ○ | ○ | ○ | ○ | ○ |
| Waar teams inhoudelijk mee bezig zijn | ○ | ○ | ○ | ○ | ○ | ○ |
| Technische kennis | ○ | ○ | ○ | ○ | ○ | ○ |

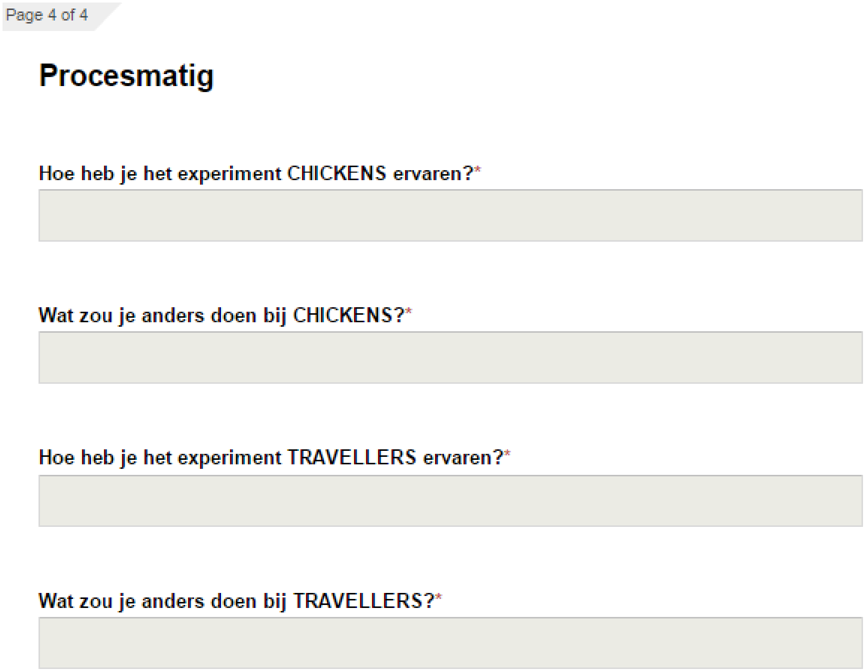
**Wat heb je nog toe te voegen aan deze lijst?**



**Welk effect hebben TRAVELLERS gehad op de volgende punten***
Let op: dit gaat over de teams die buiten je eigen Pakket/Klant liggen.

| | Heel negatief | Negatief | Neutraal | Positief | Heel positief | n.v.t. |
|---|---|---|---|---|---|---|
| Bekendheid met werkwijze andere teams | ○ | ○ | ○ | ○ | ○ | ○ |
| Wederzijds begrip | ○ | ○ | ○ | ○ | ○ | ○ |
| Waar teams inhoudelijk mee bezig zijn | ○ | ○ | ○ | ○ | ○ | ○ |
| Technische kennis | ○ | ○ | ○ | ○ | ○ | ○ |

**Wat heb je nog toe te voegen aan deze lijst?**



Page 4 of 4

## Procesmatig

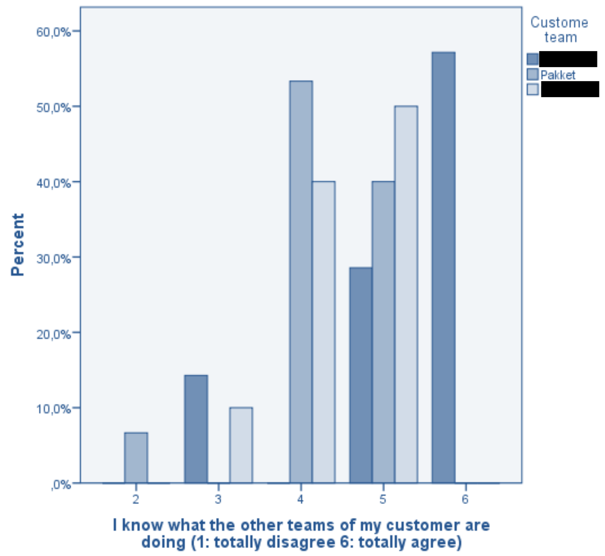
Hoe heb je het experiment CHICKENS ervaren?*

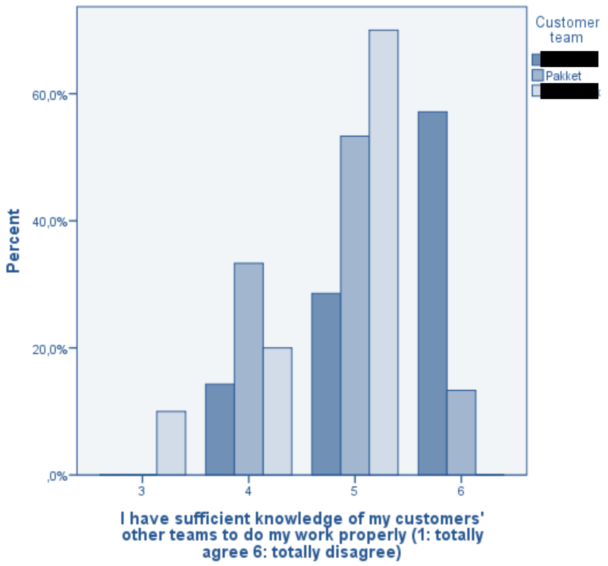
Wat zou je anders doen bij CHICKENS?*

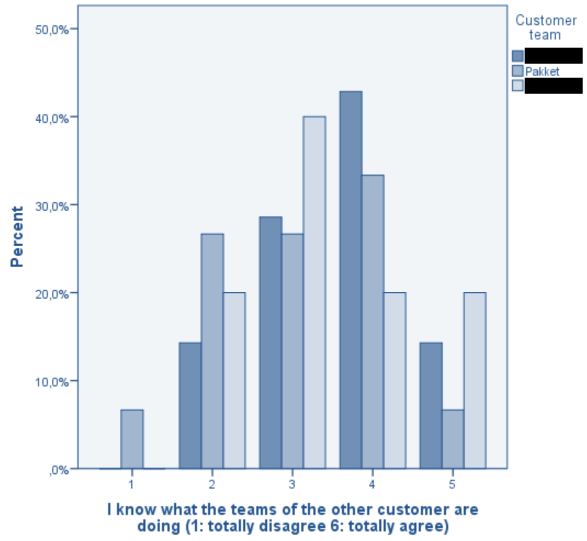
Hoe heb je het experiment TRAVELLERS ervaren?*

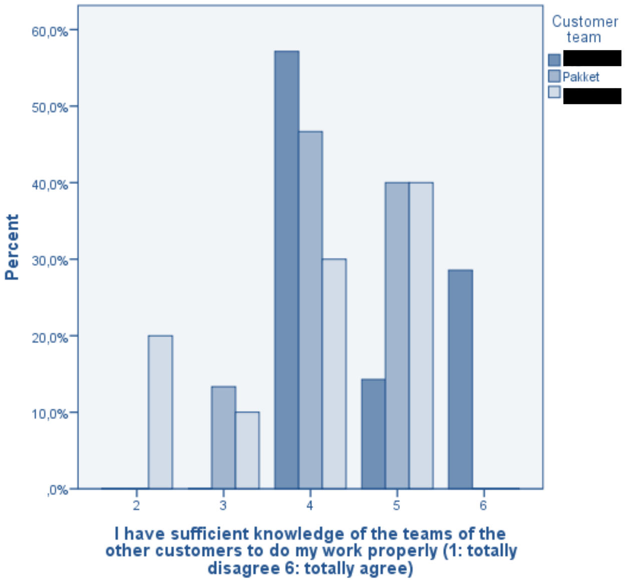
Wat zou je anders doen bij TRAVELLERS?*

# Appendix F – Distribution Post-Experiment Closed Questions


Percent
Custome team
[redacted]
Pakket
[redacted]
I know what the other teams of my customer are doing (1: totally disagree 6: totally agree)


Percent
Customer team
[redacted]
Pakket
[redacted]
I have sufficient knowledge of my customers' other teams to do my work properly (1: totally agree 6: totally disagree)


Percent
Customer team
[redacted]
Pakket
[redacted]
I know what the teams of the other customer are doing (1: totally disagree 6: totally agree)


Percent
Customer team
[redacted]
Pakket
[redacted]
I have sufficient knowledge of the teams of the other customers to do my work properly (1: totally disagree 6: totally agree)