Stakeholder & Developer Communication Powered by Product Owners

JENNIFER PERSSON
Stakeholder & Developer Communication
Powered by Product Owners

by

Jennifer Persson
Produktägarledd Kommunikation mellan Stakeholder & Mjukvaruutvecklare

Jennifer Persson
Abstract

The purpose of this study is to investigate the challenges and opportunities in the communication between software developers and their stakeholders, as well as the impact that the product owners have on the communication. The research was conducted using a case study approach, and data was mainly collected through observations and interviews with product owners and software developers at the case company Kry. The findings show that the challenges in communication are the technical language of software developers, difficulty for stakeholders to understand the software development process and difficulty for developers to understand the underlying context of a feature request. The opportunities in their communication are that the developers can better understand the reasons of a request and thus feel more engaged in the product and more satisfied towards the company; Good communication can cut development time and cost. The product owner is shown to be almost irreplaceable due to the large impact that they have. They become translators between the developers and stakeholders, enabling them to communicate better. Furthermore, they are the main point of contact for the stakeholders, thus allowing the developers to focus more on doing development work. However, this might sometimes result in that the developers get involved too late in the process, making changes costly when they discover something faulty. As a conclusion, the communication between stakeholders and developers powered by product owners greatly enhance the possibility to build good, maintainable software and lead to a more sustainable work environment for all involved.

Key words: Communication, Software Development, Inter-professional Collaboration, Professional Language, Product Owner, Stakeholder, Developer
Sammanfattning

Syftet med studien är att undersöka utmaningar och möjligheter i kommunikationen mellan mjukvaruutvecklare och stakeholders, samt den påverkan som produktägare har på kommunikationen. Detta gjordes genom en fallstudie och data är i huvudsak insamlad genom observationer och intervjuer med produktägare och utvecklare på företaget Kry. Resultatet visar att utmaningar i kommunikationen är mjukvaruutvecklarnas tekniska språk, svårigheter för stakeholders att förstå mjukvaruutvecklingsprocessen och för utvecklarna att förstå de bakomliggande anledningarna för en kravändring. Möjligheter i kommunikationen är att utvecklarna bättre kan förstå de här anledningarna och därför bli mer engagerade i produkten och känna sig mer nöjda med företaget; bra kommunikation kan minska utvecklingskostnader och tid. Produktägare visar sig vara nästan oersättliga på grund av den stora påverkan de har. De blir översättare mellan utvecklare och stakeholders, och hjälper dem att kommunicera bättre. De blir också den huvudsakliga kontaktpersonen för stakeholders, vilket ger utvecklarna mer möjligheter att fokusera på deras utvecklingsarbete. Dock kan det resultera i att utvecklarna ibland blir inblandade för sent i processen, vilket gör potentiella behov av förändringar dyra när de upptäcker något som är fel. Sammanfattningssvis, när kommunikationen mellan stakeholder och utvecklare är faciliterad av produktägare så ökar möjligheterna för att bygga bra mjukvara och det leder till en mer hållbar arbetsmiljö för alla inblandade.

Nyckelord: Kommunikation, Mjukvaruutveckling, Interprofessionellt Samarbete, Yrkesspråk, Produktägare, Stakeholder, Utvecklare
Acknowledgments

Firstly, I want to thank my supervisor Erika Lokatt for the great knowledge and guidance throughout the research and writing of the thesis. She was a great support and made the whole process easier.

Secondly, I want to thank the opponent Anton Robertsson and the examiner of the thesis Johann Packendorff. They had valuable input that helped me further improve the final version of the thesis.

Thirdly, I want to thank everyone at the case company Kry for their support and encouragement throughout the process. Especially the interviewees that took time out of their schedule to help me with my research.

Lastly, I also want to give extra acknowledgement to my supervisor at the case company, Jens Arvidsson. He has provided me with great insight and opportunities, as well as a lot of laughs during our regular meetings.

Thank you all for your participation and support!
## Contents

1 Introduction 10  
1.1 Background ................................. 10  
1.2 Problem Formulation ......................... 12  
1.3 Research Purpose & Question ................. 12  
1.4 Delimitations ............................... 13  
1.5 Outline of thesis ............................ 13  

2 Theoretical Framework 15  
2.1 Software development methods & practices .......... 15  
2.1.1 Agile Methods ............................ 16  
2.2 Communication in the workplace .................. 21  
2.3 Professional collaboration ..................... 22  
2.3.1 Professional Language ...................... 22  
2.3.2 Inter-professional collaboration ............... 23  

3 Method 24  
3.1 Research Approach ............................ 24  
3.2 The Case Company: Kry ........................ 25  
3.3 Data Collection .............................. 25  
3.3.1 Observations .............................. 26  
3.3.2 Interviews ............................... 27  
3.3.3 Ethical Considerations ..................... 29  
3.4 Literature search and review .................... 29  
3.5 Research Quality ............................. 30  
3.5.1 Reliability ............................... 30  
3.5.2 Validity ................................. 31  
3.5.3 Generalizability ........................... 32
3.5.4 Complementary evaluation criteria ............... 32
3.6 Research Process .................................... 33

4 Results & Analysis ................................. 35
   4.1 Product Owner as a Translator .................. 35
   4.2 Point of Contact ................................. 40
   4.3 The Art of Saying “No” ......................... 43

5 Discussion .......................................... 48
   5.1 Reflections on Sustainability ................... 54
   5.2 Limitations & Future work ...................... 55

6 Conclusion .......................................... 56

Bibliography ......................................... 59
   A Observation Template ............................ 64
   B Interview - Product Owner ...................... 65
   C Interview - Developer ............................ 67
Chapter 1

Introduction

This introductory chapter presents the intent of this report. The first section provides a brief background to the subject in question and introduce the problem. Moreover, the chapter presents the research purpose, questions and delimitations needed to comprehend the report. Lastly, the disposition of the report is introduced.

1.1 Background

Communication is an essential part of an organization (Conrad 2014), and good collaboration between teams can be a factor of success (Hansen & Press 2009). This is nonetheless true in software development and a software development organization; with the rise of agile software development methods the communication with stakeholders and users have become part of the daily work for the developers (Beck et al. 2001). However, the communication does not come without challenges, and poor communication with stakeholders is one of the main factors for why software fails (Charette 2005).

Due to popular agile software development methods such as Scrum, there exists a third party in the communication between software developers and stakeholders. The product owner is considered part of the product development team, but is mainly responsible for being the stakeholder representative towards the team (Sverrisdottir et al. 2014). Despite this important mission, there exist little information on
how the product owner affect the communication between stakeholders and developers and the software development process.

Moreover, as the world becomes more digital an increasing amount of companies and industries that traditionally have not been software dependant now use it to enhance their product and/or service offerings. This digitization has lead to a dramatic increase in costs for software development in organizations in all kinds of business sectors. It is often a race to become the winner of the best digital solutions in which time to market, rich user experience and broad functionality are all needed. The success of companies now more than ever may depend on their ability in releasing the right software offering sooner rather than later. To be successful, it is of utmost importance that the stakeholders of a product can describe what they need in a way that the software teams can understand and where the software teams can explain what the new disruptive technologies can make possible. The winners of tomorrow are the organizations that can understand what the new technologies offers that can provide dramatic improvements and get amazing products out the door. In order to make this possible, the communication between stakeholders and software developers is absolutely crucial.

An example of this is the banking industry that is successfully digitizing it’s business (Cuesta et al. 2015). Another traditionally non-digital industry is the healthcare industry, where more actors providing digital healthcare continue to emerge, such as Kry (n.d.) and Babylon (n.d.). This creates instances where previously unrelated professions such as doctors and software developers now have to interact and together create value for the users.

By considering software developer as a profession, it complicates their communication as the professional language may become an obstacle for people outside of the profession to understand them. Professional language is something all professions develop to facilitate the communication between members of a profession (Ironside 1989), however it may make it difficult for other occupations to understand the highly technical jargon. This often becomes a barrier for representatives of different professions to understand each others perspectives (Geijer 2003).
1.2 Problem Formulation

An increasing amount of companies and industries that traditionally have not been software dependant now use software to enhance their product and/or service offerings (Cuesta et al. 2015)(Kry n.d.). This increase illuminates the need for efficient communication between software developers and their stakeholders. However, communication between the professions are commonly hindered by professional languages and difficulties in understanding each others perspectives (Geijer 2003).

Ensuring that the collaboration between teams work seamlessly is of great importance as it has been proven to be a major factor of success (Hansen & Press 2009). This is true also for software development since poor communication between developers and stakeholders are one of the most common factors of why software fails (Charette 2005). Despite this, there is little research on the communication between stakeholders and software developers in an agile setting, and there exist a gap in research that evaluates what role the product owners play in this communication. Furthermore, it is also important to understand how the impact of product owners on the communication affects the software development process. By understanding the challenges and opportunities in the communication between software developers and their stakeholders and the impact of product owners, the communication can be dramatically improved. This would have the potential of being the difference between success and failure in software development projects.

1.3 Research Purpose & Question

This study will investigate the communication between software developers and their stakeholders, and the effects that the product owner have on this communication. The purpose of the study is to get a better understanding of the impact that product owners have, and to contribute to a theoretical basis so that future research can build on top of this to further improve the communication.

The research questions are posed as follows:
• What are challenges and opportunities in the communication between a software development team and stakeholders?

• What are challenges and opportunities with the product owner facilitating the communication?

• How does the above affect the software development process?

1.4 Delimitations

The scope of this study is delimited to what Blomkvist & Hallin (2015) refer to as the individual level and the functional level. The individual level means the perspective of the management and employees and the functional level refers to the perspective of process. The scope will be a mixture of these levels, as this thesis will investigate the impact that the communication has on the software development process at the case company and interview the employees to understand their perspective on the phenomenon. The industrial level will not receive attention in this study. Furthermore, the stakeholder communication is delimited to internal stakeholder communication, i.e. the communication with stakeholders within the organization.

1.5 Outline of thesis

The disposition of the thesis is as follows:

1 Introduction - The first chapter gives an introduction to the study by introducing the phenomenon, formulates the problem and presents the purpose of the research.

2 Theoretical Framework - The second chapter introduces the theoretical framework. It focuses on software development methods, communication in the workplace and professional collaboration.

3 Method - The third chapter explains the research approach and the data collection. It also introduces the case company and discuss the quality of the research.
4 **Results & Analysis** - The fourth chapter presents the results of the case study divided into four areas that was identified and analyzed in the empiric.

5 **Discussion** - The fifth chapter discusses findings from the previous chapter, including a reflection on sustainability. The limitations of the study is also discussed, as well as suggestions on topics for future studies.

6 **Conclusion** - The sixth chapter summarize the result and discussion of the study.
Chapter 2

Theoretical Framework

This chapter presents the framework that relate the subject of software development, communication in the workplace and professional collaboration. Firstly, a deep dive into the communication in agile development and the role product owner. This is then followed by descriptions of barriers of effective communication in organizations. Lastly, professional language and its effects on communication is explained, together with theory on inter-professional collaboration.

2.1 Software development methods & practices

Karam & Casselman (1993) define software development methods as systems of "technical procedures and notational conventions for the organized construction of software based-systems". Jacobson et al. (2019) express it more colloquial as something that provides "guidance for all things that you need to do when developing and sustaining software" (Jacobson et al. 2019, p.27). Thus, the methods are suggestions for ways of working with all the different aspects of software engineering. This includes how to manage the client and stakeholder communication, how to organize the teams, testing etc. Jacobson et al. (2019) further make a distinction between methods and practices, where practices are mini-methods that give guidance to some parts of the software development process. According to them, a method is composed of a collection of
practices, all practices are reusable and new combinations can create new methods (Jacobson et al. 2019).

The two most common approaches to software development are traditional and agile software development. Traditional software development includes methodologies such as the Waterfall method, V-model and RUP (Leau et al. 2012). The traditional methods are based on a sequential series of steps, e.g. requirements definitions, solution building, testing and deploying. Furthermore, these methods also require a lot of documentation, especially in the form of requirements for the program.

In a reaction to the inflexibility of the traditional methods (Bygstad et al. 2008), the agile approach has gained popularity and become the most applied by addressing the unpredictability of the world through putting focus on individuals and their creativity rather than the software development process (Dybå & Dingsøyr 2008). In this study, the employees of the Company work agile (specifically with Scrum and Dual Track Agile) and the agile practices will thus be explained in more detail in the following section.

2.1.1 Agile Methods

In contrast to traditional methods, agile software development is incremental and iterative (Leau et al. 2012). Williams & Cockburn (2003) further state that agile development is about embracing feedback and change. According to the agile manifesto the software development should focus on four main values (Beck et al. 2001):

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

In the context of communication, agile practices are considered to have a positive effect on the internal communication of a development team, where informal communication reduces the need for documentation and therefore facilitate more productive teams (Pikkarainen et al. 2008).
In their study, Pikkarainen et al. (2008) also found that agile methods have a positive impact on some external communication, however some communication hurdles are still present in the communication with the stakeholders. Their results showed that some popular agile practices, e.g. Scrum, do not provide enough of communication mechanisms and thus need extra process practices to ensure that the developers and stakeholders interact regularly (Pikkarainen et al. 2008). In the study of Svensson & Höst (2005), they got similar results and reached the conclusion that agile overall improved the communication between the development team and the rest of the organization.

Moreover, Holmström et al. (2006) found in their research that agile practices can help the communication in a global software development environment. Particularly, the practice Scrum was shown to improve communication in a globally distributed development team and decrease misunderstandings (Holmström et al. 2006). However, not much was mentioned on communication with stakeholders in their study.

**Scrum**

Scrum has become the de facto standard method used in software development (Larusdottir et al. 2017), and it is not only a software development method but also a management process that can be applied to a numerous amount of different activities (Holcombe 2008). In line with the agile manifesto, Scrum is based on flexibility, adaptability and productivity. A characteristic of Scrum is that a Scrum team is self-controlled (Sverrisdottir et al. 2014). This encourage people to suggest new ideas, resulting in transparency in decision-making and more freedom in the development process. Furthermore, the method encourage continuous review of the development processes, ensuring that the things developed are aligned to the ever changing environment and requirements (Sverrisdottir et al. 2014). The Scrum method is visually demonstrated in Figure 2.1.

In Scrum, everything is divided into features with assigned project value and estimated effort and cost (Holcombe 2008). The different phases are called sprints and each sprint should result in complete and usable software. By breaking down project requirements into sub-
goals and short activities, the work is prioritized and tracked in the product backlog (Holcombe 2008). Each sprint is planned by the team and tasks from the product backlog are moved to the sprint backlog. Everyday the team takes part of a Daily Scrum meeting (also called daily stand-up meetings), which is a short and focused meeting on current status and challenges in the sprint (Holcombe 2008). These meetings are held to monitor the progress of the development as well as encourage collaborations and knowledge sharing between the members, because although solutions are not supposed to be discussed during the stand-ups people are free to discuss them afterwards. This is something that increases the transparency of the process. At the end of each sprint, a retrospective is held (Holcombe 2008). It is a meeting that aims to review the process and result of the sprint, to encourage learning through out the development process.

A team in Scrum consists of three different roles: Product Owner, Scrum Master and Team Member (Sverrisdottir et al. 2014). The Scrum Master is responsible for ensuring that the process is applied, maximizing the the benefits of it and for supervising the communication within the team (Sverrisdottir et al. 2014) (Judy & Krumins-Beens 2008). The Scrum Master is a compliment to the product owner, where the product owner is responsible for "What" to do, the Scrum Master is responsible for "How" to do it.

The product owner is responsible for managing the product backlog
to maximize the business value of the product (Judy & Krumins-Beens 2008). This means representing all stakeholders, prioritizing all requests and ensure that the team is working in the right direction to create as much value as possible for the users to a cost as low as possible (Judy & Krumins-Beens 2008) (Sverrisdottir et al. 2014). Being the representative for all the stakeholders requires that the product owner communicates closely with them to understand their requirements, however it is imperative that the product owner rejects new requirements that are not necessary (Sverrisdottir et al. 2014).

There is little in research about opportunities and challenges of the product owners communication with the stakeholders. More studies have been made about the collaboration between the product owners and the software developers. In the research by Sverrisdottir et al. (2014) the role of the product owner in Scrum is studied and it is concluded that the role of product owner is very difficult. The reason for the high level of difficulty is the many factors that determine the success for the product owner; organizational culture, type of project, management approach, and the interaction within the team that is developing the product all effect the outcome (Sverrisdottir et al. 2014). One of the most important tasks of the product owner is deciding what to implement to ensure that the team is not wasting time on unnecessary things. Furthermore, the product owner needs unrestricted authority regarding the prioritization, otherwise there is a danger of misunderstanding the requirement and thus not meeting the expectations of the stakeholders (Sverrisdottir et al. 2014).

Judy & Krumins-Beens (2008) discuss the concept of collective product ownership, where there is no strict boundary between the product owner and the development team. The product owner keeps the accountability for the product, but shares the authority over vision, priorities and execution. The collaboration is built upon high-performance, mutual respect and deep trust. For the product owner, it becomes about engaging the team towards the product while guiding the product towards the vision and high-level goals (Judy & Krumins-Beens 2008). This results in a team that is passionate about their product and feel personally responsible for it’s success. However, collective product ownership takes a lot of time and has not been proven to be high-performing yet.
Dual Track Agile

Dual Track Agile is a method for project development in software development with a high focus on UX-design (Jungbeck 2018). The two tracks are the Discovery track and the Delivery track, and they work in parallel with each others. The purpose for having two different tracks is to give the product owner, designers and developers enough time to understand the requirements of new ideas or features before putting them in the backlog for the software developers to code, without obstructing the development of already existing coding tasks (de Lichtenberg 2017).

The Discovery track is about understanding the stakeholders to reveal use cases and goals and translate them into product requirements, i.e. discover what should be done. The Delivery track includes developing, testing and concept iteration, i.e. deliver an implemented design and idea. (Yang 2015)

Figure 2.2: A visualisation of the Dual Track Agile process (Yang 2015)

Yang (2015) emphasizes the importance to involve the stakeholders in the Discovery process, as it will help with understanding the requirements, prioritize features as well as cut development costs as it will reduce the amount of testing needed later on. In the Delivery track, it is important to have a clickable design prototype and clear pattern library, as it will help cut development time. User involvement is critical in this track, as it will prevent the building of features or designs without real use (Yang 2015). However, it is a relative new method with lack of extensive research on the subject and thus empiric of the effects that come from using this (Jungbeck 2018).
2.2 Communication in the workplace

"Communication is the lifeblood of an organization and, without it, things will not get done." (Conrad 2014)

Some of the common barriers of effective communication in organizations are described by Kelly (2000) as: Perception, communication depends on our perception, or how we perceive people, their motives, and intentions; Information overload, a by-product of the sheer volume of information and data that managers deal with on a daily basis; Technical and in-group language, different vocabularies make it hard for one individual or group to communicate with another; Absence of formal communication channels. Kelly (2000) advocates that "vision-directed communication help build a shared context by reinforcing values and beliefs about what is important in the organization. This shared context is essential for effective decision making and problem solving, as well as communication.". Thus, it is suggested that vision-directed communication can help to overcome the mentioned barriers for effective communication. The barrier of perception can be overcome through the process of communicating vision, as it can raise awareness of important organizational goals and thus provide additional information about goals and expectations. Regarding the barrier of information overload, more information is actually better, as long as it is the right kind of information, suggesting that the manager need to understand what information is actually needed for their work. The barrier of different in-group language can be minimized through the unifying effects of vision, aligning everyone behind one common goal, and images and metaphors can be used when communicating the vision to accentuate inclusion. (Kelly 2000)

Other challenges of corporate communication is transparency as a lot of organizations mistakenly assume that they are transparent internally; transparency is a necessary strategy for organizations to cope with inquisitive stakeholders (Thøger Christensen 2002). Moreover, De Ridder (2004) found that the higher quality of task-related information for the employees of an organization, the higher the satisfaction and thus commitment they feel towards the company; the higher the quality also results in more trust in management. Furthermore, the comparison made by Hansen & Press (2009) of initiatives of two
organization from the technology industry show that it is not only important with well-functioning cross-team collaboration, it is a factor that can determine the difference between success and failure.

### 2.3 Professional collaboration

Professions are occupational groups that base their work on scientific research (Brante 2009). This traditionally includes professions such as doctors, nurses and lawyers; nowadays software developers and management can also claim to be new kinds of professions - corporate professions (Marks & Scholarios 2007) (Muzio et al. 2011). Thus, it is of interest to include research on professional language and inter-professional collaboration to the theoretical framework of this study. This section will continue by presenting research covering these fields.

#### 2.3.1 Professional Language

Professional language refer to the "jargon", the specialized language, that professions develop to facilitate communication within the profession (Ironside 1989). It is something that all professions have, and it helps them to give names to different tools, processes and rules (Geijer 2003). While it is something that greatly improves the communication within the profession, it becomes an obstacle for people outside of the profession to understand. The professional language might appear highly technical and even incomprehensible, complicating the communication between professions and other occupations (Geijer 2003).

Geijer (2003) found in her research that professional language creates both challenges and opportunities in inter-professional collaboration. It may result in rejection of each others languages and competencies. However, she found that through continuous interactions that the professions would pick up each others languages and overcome the barriers of it (Geijer 2003).

Williams (2002) explored the use of art work to overcome the different professional languages in the healthcare sector and found that collage work can be used to enable participants at interprofessional meetings
to use a medium as a bridge between the different languages. It almost becomes a neutral common language that the different professions all understand and through the neutrality it enables clearer understanding of each other’s expectations (Williams 2002).

### 2.3.2 Inter-professional collaboration

All professions have different cultures, including values, beliefs, attitudes, customs and behaviours and these professional cultures complicates the inter-professional collaboration (Hall 2005). Schein (2016) believes that different cultures evolves in social contexts and to understand them it is required to understand their background. Through understanding the driving forces in a culture, or a specific group, it enables better understanding of each other and the collaboration between the groups (Schein 2016)(Hall 2005).

Lingard et al. (2005) introduced a checklist for healthcare professions to promote inter-professional communication in the operating room. This was done with the intent to improve the quality and safety of care. The checklist showed promise as a feasible and efficient tool that promotes information exchange and team cohesion (Lingard et al. 2005). A couple years after the study of Lingard et al. (2005) was published, World Health Organization (WHO) published a Surgical Safety Checklist that has since then been implemented in different countries (Semel et al. 2010). The Surgical Safety Checklist is divided into three different phases of the operation, with some specific checks directed to specific professions. The checklist has proved to not only reduce the cost but also improve the quality of care by preventing error (Semel et al. 2010).

Åkesson & Henell (2014) found that if knowledge-sharing occurred in a collaboration between different professions then this created a better understanding of each others professional language. Thus, decreasing the barrier of professional language and enabling better communication. Moreover, in the study of Abrahamsson & Gredevik-Heinebrün (2009) it was shown that by educating the inter-professional work team about communication the collaboration would improve.
Chapter 3

Method

In this chapter the study’s methodology is discussed. Firstly, the research approach is presented. The case company and its relevance for this study is thereafter introduced. This is followed by in-depth descriptions of the data collection and the literature research and review. The research quality is then discussed. Lastly, the chapter concludes with a summary of the research process.

3.1 Research Approach

To achieve the purpose of this thesis, the decision was made to conduct a case study of the company Kry. A case study can be described as a method that is used to explore a single phenomenon in a live, natural and specific setting (Collis & Hussey 2009). Blomkvist & Hallin (2015) state that it is a suitable method when the purpose is researching, explaining or describing. As this research aims to describe and get a better understanding of the communication between software developers and their stakeholders, and the product owners role in the communication, a case study was deemed the most fitting.
3.2 The Case Company: Kry

The case company Kry, hereafter referred to as the Company, is a digital healthcare provider that operates in Sweden, Norway, France and the United Kingdoms. The fast growing company was founded in 2014 and has at the time of writing 300 employees (and approximately 500 clinicians), with close to 80 employees in the RnD department. The RnD department consists of the CPO (Chief Product Officer), CTO (Chief Technology Officer), Head of Design as well as the development teams. The development teams consist of a Product Owner, software developers and if needed a designer.

The Company works agile in the RnD department with Scrum and Dual Track Agile practices. Their sprints are 1 week long and the teams have daily stand-up meetings, and sprint planning and retrospective once a week. The product owners have biweekly stakeholder meetings where all stakeholders of a team is invited to attend if they want to be informed about the projects of that development team. It is mainly in the Discovery meetings that the software developers interact with the stakeholders, and the product owner is commonly present during these meetings. In addition to the Discoveries, the development team and the stakeholders interact through Slack, a digital team-collaboration tool.

The Company is quickly growing and thus increasing the complexity of the organization as they add more development teams and even more stakeholders. This makes the Company a great case to study, as the complexity of the communication between the development teams and the stakeholders are very palpable and thus making a great example of the phenomenon studied. Furthermore, it is a modern company that works with popular and modern software development methods, thus making the case more relatable to other cases.

3.3 Data Collection

Collis & Hussey (2009) believe that to properly perform a case study, it is crucial to obtain an in-depth knowledge of the phenomenon and understand the context of the setting. This is achieved by collecting a
large volume of data, something that Blomkvist & Hallin (2015) claim to be a characteristic of a case study. The data can be collected through multiple methods, and in this case it was collected through observations and semi-structured interviews with employees at the Company.

### 3.3.1 Observations

Observing the company and its employees was used as a method to gain a thorough understanding of the context of the case study and to gather empiric. Firstly, the author was welcomed to the Company and allowed access to everything as any other employee. This enabled the author to understand the use of the public communication channels as well as the company culture. Secondly, the author attended “intro-days” at the company together with newly hired employees, where the CEO and rest of the management and department heads participated and held presentations and Q&As so that the new employees could get an overview of the company, its departments and the ways of working. In total there was three introductory days with additional presentations spread out on other days. Two days were dedicated to the whole company and the third day was specifically for the R&D department that the software development teams belong to. The observations in these situations contributed to the gathering of information about the context of the phenomenon and this study. It also resulted in personal relations between the author and the employees.

As these activities allowed the author to become as one of the employees, this then enabled for what Czarniawska-Joerges (2007) refers to as Shadowing. Shadowing is when the observer keeps the perspective of an outsider to remain unbiased, while still gaining first hand access to the organization’s processes. This is preferable to participating observations when the observer becomes one in the group to experience what the members of that group is experiencing. While this allows the observer to record the natural behaviour of the group and allows the gathering of data which could not be obtained unless the observer is considered a member of the group; the critic against participating observations is that the observer might have difficulties remaining objective (Kothari 2004). This was avoided in this study through not participating and only shadowing in the Discoveries and meetings attended,
thus remaining objective.

Thirdly, the empiric gathered through observations was collected by attending Discoveries and stakeholder meetings that the product owners invited the author to. A total of 9 Discoveries was attended and notes was taken according to a prepared template, see Appendix A. Both Discoveries with stakeholders and Tech Discoveries (where the development team themselves discuss the technical solution, sometimes with input from members of another development team) was observed. The choice to attend both kinds of Discoveries was taken as it was of interest to observe if and how the presence of a stakeholder affected the actions of the team members and the product owner. One discovery with the Product Owner not present was also observed. The Discoveries were between 30 min to 2 hours long, with an average of 1 hour.

Lastly, as the Company constantly strives to improve their processes, one employee (an engineering manager) conducted a workshop with members of the R&D department with the subject of improving their software development process. This was before the study had started, but the author was invited to participate in following meetings discussing the results and possible improvements to their process. In these meetings both managers, product owners and Head of Design participated, allowing for insights to their thought process on the context of the phenomenon.

3.3.2 Interviews

The semi-structured interview is the most commonly used interview methodology and is organized around a number of predetermined themes or question areas (Blomkvist & Hallin 2015). It is an advantageous method to explore a phenomenon while containing it to themes and question areas of interest. In this study the semi-structured interview methodology was chosen as it seemed to be an appropriate way to explore the communication between software developers and their stakeholders and the effects that the product owners have on it.

The interviews were conducted with the employees of the Company towards the end of the study. The participating interviewees were either a product owner or a software developer and they are presented
in Table 3.1. All interviewees were asked about predetermined themes and question areas depending on their profession, see Appendix B and C for the interview guides. Furthermore, the interviews were recorded with the permission of the interviewees and transcribed afterwards.

<table>
<thead>
<tr>
<th>Date</th>
<th>Profession</th>
<th>Background</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>2019-04-10</td>
<td>Product Owner</td>
<td>Scrum Master</td>
<td>50 min</td>
</tr>
<tr>
<td>2019-04-16</td>
<td>Product Owner</td>
<td>Product Owner</td>
<td>45 min</td>
</tr>
<tr>
<td>2019-04-17</td>
<td>Product Owner</td>
<td>Medical Doctor</td>
<td>50 min</td>
</tr>
<tr>
<td>2019-04-30</td>
<td>Software Developer</td>
<td>-</td>
<td>35 min</td>
</tr>
<tr>
<td>2019-04-30</td>
<td>Software Developer</td>
<td>-</td>
<td>45 min</td>
</tr>
<tr>
<td>2019-05-02</td>
<td>Software Developer</td>
<td>-</td>
<td>45 min</td>
</tr>
<tr>
<td>2019-05-06</td>
<td>Software Developer</td>
<td>-</td>
<td>45 min</td>
</tr>
</tbody>
</table>

Table 3.1: List of interviewed employees at the Company

It was decided to interview employees that have been with the Company for a longer time as they would be knowledgeable on the communication processes in the company and employees who had shown interest in the study. However, as the Company was founded in 2014 and is rapidly growing at the time of writing, “a longer time” at the company is still relatively short. All the product owners had been at the company for more than one year. While choosing product owners to interview, their background was also taken in regard as it was deemed interesting to talk to employees with different backgrounds to see if it affected their opinions of the phenomenon.

Lastly, no stakeholder was interviewed for this study. The decision to not interview any stakeholder was based on it being a diverse and large group of people consisting of many different professions, where the stakeholders had varied amount of contact with the developers and product owners. It was deemed problematic to find people with a lot of experience of being a stakeholder in these situations while also be able to sufficiently represent the whole group. Due to the difficulties to identify suitable interviewees it would be difficult to satisfyingly generalize the result and ensure the research quality; thus the stakeholders were omitted from the interviews. Moreover, while the perspective of the stakeholders on the challenges and opportunities in
the communication is interesting, their perspective would not aid in answering the third research question; thus further justifying omitting them from the interviews.

### 3.3.3 Ethical Considerations

The data gathering process of a qualitative study in social science most often includes methods which involves human beings, thus making it important to consider the ethical aspects. As this study was conducted in Sweden, the Swedish Research Council four ethical requirements for humanities and social science research was followed (God forskningssed 2017). All participants of this study were informed of the purpose of the study and their participation, they were asked for permission and they were kept anonymous throughout the report. The data collected is only used for the purpose that was explicitly given permission to.

### 3.4 Literature search and review

The literature was gathered throughout the whole research process. Initially it was to get an overview of the phenomenon and existing research, but as the research progressed the gathering of theory became more specific. It was important to find new perspectives from different research areas to combine them under the studied phenomenon.

Literature was searched mainly through Google Scholar and Web of Knowledge, but some was searched through the DiVA portal. Search terms with different spellings, modifiers, abbreviations and combinations used for the literature search included: agile development, agile communication, scrum, dual track agile, corporate communication, stakeholder communication, internal stakeholders, professions, professional language, professional communication, professional coordination, professional collaboration, software developers as a profession.

The Petersen & Ali (2011) suggestion for study selection was used as a guide to ensure good literature quality. However, Swedish sources were also used despite their criteria of only sources written in English.
Furthermore, as this thesis is not limited to the area of software engineering and computer science, research papers outside of this area were also collected. Peer reviewed articles and textbooks were preferred over other sources in order to maintain a high research quality.

Moreover, additional research fields were studied while creating the theoretical framework; they were however omitted along the research process as they were deemed unfitting for this case. An example of this is the field of linear communication that was deemed unfitting due to the product owner being a third party in the communication which is not included in that research field as they only consider one sender and one receiver in the communication.

### 3.5 Research Quality

This section covers the quality of this research. The three concepts of reliability, validity and generalizability provide a framework for the evaluation of the research and are thus discussed. Lastly, an alternative evaluation criteria is applied and discussed.

#### 3.5.1 Reliability

Gibbert et al. (2008) describes reliability as the absence of random error in the research; a study with high reliability would reach the same conclusions if it was repeated (Collis & Hussey 2009). Transparency and replication are key concepts of reliability, and enhancing them by careful documentation and clarification of the research process can improve the reliability (Gibbert et al. 2008).

To enhance transparency, Gibbert et al. (2008) suggest putting together a case study protocol that is a report that specifies the procedures of the entire case study. An independent case study protocol was not produced for this study; however the method chapter of this thesis aims to serve as a substitute for this, so that subsequent researchers could repeat the study.

Replication can be accomplished through creating a case study database that includes case study notes, the case study documents, and the nar-
 CHAPTER 3. METHOD  31

ratives collected during the study (Gibbert et al. 2008). However, a case study database was not created for this study due to confidentiality restraints as the meetings and interviews contain sensitive data for the Company; something that affects the reliability of the study. Furthermore, as the data is highly qualitative, reliability in terms of replication of data can be deemed lacking. All the observation and interview guides are made available in the appendixes for the readers, but as the interviews were semi-structured and probing questions were asked the interview sheets do not cover all that was discussed during the interviews.

3.5.2 Validity

Validity refers to the extent that a study investigates what it claims to be investigating (Collis & Hussey 2009) (Gibbert et al. 2008). It is especially important to consider during the phase of data collection to ensure that the procedures lead to accurate descriptions of reality. To increase the validity of a study, Gibbert et al. (2008) suggest that the study should establish a clear chain of evidence and use data triangulation.

A clear chain of evidence is established through detailed descriptions how the researcher went from the research questions to final conclusions (Gibbert et al. 2008). This was achieved in this thesis by continuously motivate delimitations, methodological decisions, the chosen theory and how result and theory support the conclusions.

Data triangulation can be achieved through adopting different angles to look at the same phenomenon, i.e. by using different data collection strategies and different data sources (Gibbert et al. 2008). Two different data collection methods were used in this thesis to increase the triangulation; however, both observations and semi-structured interviews are considered qualitative methods with the interviews being the main source of data. The data triangulation and thus the validity could have been improved by collecting more non-interview data.
3.5.3 Generalizability

Collis & Hussey (2009) describe generalizability as to what extent the research findings can be extended to other cases and settings. As previously mentioned, a case study can never be statistically generalizable and thus the result can never claim to apply to all other cases (Blomkvist & Hallin 2015). However, a case study can be analytical generalizable, meaning that it can be discussed how this can apply to similar cases. To achieve analytical generalizability the study needs to be described with sufficient details so that the reader can assess whether the reasoning holds (Blomkvist & Hallin 2015). This thesis has strived to properly record and describe the case study, as can be read in this chapter.

3.5.4 Complementary evaluation criteria

Common critic against the above framework is that it is impossible to replicate social settings, thus making it impossible to achieve reliability and validity in qualitative research in the field of social science. Eriksson & Kovalainen (2015) suggest that if the research relies on relativist ontology (that there are multiple realities and that reality is subjective) or subjectivist epistemology (the emphasizing of that the researcher and the participants jointly create understanding), then these philosophical starting points should be considered when discussing the research quality. Instead of the traditional framework, Eriksson & Kovalainen (2015) suggest the use of the concept of trustworthiness created by Lincoln & Guba (1985). This concept consists of the following four aspects: Dependability, that the research has been logical, traceable and documented; Transferability, the degree of similarity between the research and previous results in other research contexts; Credibility, the degree of how suitable a researcher is to study a certain topic, if other researchers would reach the same conclusions with the same material; Conformability, relating the findings and interpretations to the data in ways easily understood by others.

This study relies on both relativist ontology and subjectivist epistemology, as the researcher believe that her background affects how she analyzes the results and together with the participants create an understanding of the studied phenomenon. Acknowledging how the re-
searcher’s own subjectivity, her background as a Master of Science in Computer Science student with a direction of Industrial Management has implications for the study. Her background in computer science could be seen as an opportunity in the research as she is able to understand the software developers and help her identify opportunities and challenges in the communication. Similarly, her background in industrial management could be seen as an opportunity as she also can understand the viewpoint of the product owners. Thus, making the researcher suitable to study this topic, giving high credibility to the research.

This thesis has aimed to properly record and present the research process in the Methods chapter to raise the dependability of the research. Furthermore, the thesis has aimed to properly show relations between this research and previous result through properly documenting the case and introducing previous research to ensure high transferability. Lastly, in chapter 4 Results & Analysis the aim has been to properly display the relationship between the data and how this was interpreted as findings to achieve conformability. Through applying these measures, the researcher has aimed to achieve high trustworthiness in the research in this alternative criteria for research quality.

### 3.6 Research Process

The research process for this study has been iterative, with the empirical data as a guide for the literature search and formulation of research questions. Blomkvist & Hallin (2015) explain this to be an inductive research process, where the researcher allow the empirics to generate theory in order to make the best fit of theory to the empirical data.

Firstly, an exploratory pre-study was conducted to understand how the development and communication processes are at the Company. This was done through observations and from this the important role of the product owner became obvious, and thus affected the direction of the research process. Thereafter, the theoretical framework was adjusted and the observations continued to prepare for the interviews. After the interviews, no major theoretical changes were made
but rather fine adjustment to ensure a better fit to the empirical material.
Chapter 4

Results & Analysis

This chapter presents empirical findings, gathered through interviews and observation, jointly analyzed with secondary data gathered through literature. Firstly, the challenges and opportunities of a product owner acting as a translator between the software developers and their stakeholders and effects of that will be analyzed. This is then followed by the challenges and opportunities of a product owner being the point of contact, and how the lack of a corresponding person in the stakeholders creates complications. Lastly, a recollection of the challenges and opportunities with prioritization of requests.

4.1 Product Owner as a Translator

Something that became obvious in the interviews and Discoveries attended was the translation work that the product owner do between the development team and the stakeholders. A part of it is because of the technical level of the software developers that the stakeholders can not match, with one of the product owner expressing that they sometimes need to remind the developers to take a step back from the technical solution as to not lose the room. This is an example of professional language (Geijer 2003) and might become a barrier for the communication in the meeting. One of the developer expressed it as follows:

"Explaining things so that they understand it especially if it is technical issues. They usually have a quite different view of the system than we have
because they only see some parts of it and then it’s sometimes really hard to explain it in a good way so they understand why things are as they are and why things happen as they happen.”

The developers might see it as a challenge for them to explain things properly, but it is however considered to be a small risk at the Company so they must be doing it quite well together with the product owner. As one of the product owners puts it:

“There exist a stereotypical idea that software developers and stakeholders don’t speak the same language. I feel it is a pretty small problem”

Another product owner approached it as follows:

“You should not worry if the developers can’t explain to you so that you understand, then it is a bad developer and not your fault.”

This was further elaborated as that a good software developers know how to explain technical decisions fitting for the listener, and that the product owner felt that they only had good developers in their teams. The ability to adjust to the technical level of the listeners was reflected in the Discoveries. In the Discoveries with stakeholders with lower technical background, the conversation was not technical, and in the Discoveries with stakeholders with more technical knowledge (e.g. the Analytics team), the developers used more technical language. The most technical language was observed in the Discoveries with only development teams present.

In the situations when either side of the communication finds it difficult to explain what they mean, the product owner would sometimes suggest that they should use the whiteboard in the meeting room to better explain something by drawing examples. This is something that they could and sometimes did do without encouragement from the product owner, but it again shows how the product owner helps decrease the barriers of communication. This might not be exactly the same as the collages that was used to lower the barriers of professional language in the study of Williams (2002), but it does remind of the artwork that proved useful in that study. Similarly, it relates to the images used to better communicate vision as a way to lower the barriers of ingroup languages (Kelly 2000).

However, the professional language also includes processes and ways
to describe them (Geijer 2003) and the communication around the software development process at the company is something that all interviewees identified as something lacking. Most of the stakeholders have problems grasping how the development teams work, as put by one of the software developers:

“They don’t understand that we start with building the minimum viable product (MVP) that covers all the basic needs and that we then continue to build on the product in iterations. If someone hasn’t worked like that before, then they tend to be disappointed when they don’t get everything from the beginning.”

The stakeholders unfamiliarity with the development process is a challenge that can result in disappointment for the stakeholders when their expectations do not get fulfilled, something that the product owners often have to deal with. One of the product owners expressed the difficulty with stakeholders always wanting to know the expected deadlines of a project or a feature:

“Almost all the stakeholders want to have exact time frames for when it should be done. It is reasonable as they need to communicate that to their stakeholders, I understand that. But I have learnt that if I promise a date and not keep it, then it is a lot worse than not promising something.”

The product owner said that in case a deadline had been communicated and later was not kept, and the product owner did not properly explain why then the stakeholders take it almost as a betrayal and get very upset. Furthermore, it is not only the stakeholders that get a bad experience out of it; when a product owner promises a date, both product owners and developers identify that it puts unnecessary pressure on the developer to deliver something that results in being worse than it would have otherwise. In the situations when the stakeholders want a deadline and the product owner can not say anything final, the product owner deals with in the following way:

“Some may think that I am pretty vague in my communication, but it is a conscious choice as the alternative is worse.”

In one of the observed stakeholder meetings, there was a situation where a stakeholder asked for a date when the request could be expected to be finished. Instead of giving a deadline, the product owner explained the software development process and said that it is difficult
to know that far ahead as they work in 1-week long sprints. However, it is not only the product owners feel the need to explain the process. In the interviews with the software developers some also expressed that they feel responsible to educate the stakeholders on the concept of first building the MVP.

“You need to teach people that you actually can ship things in steps, that you can build a MVP.”

When one software developer was talking about the greatest challenges in the communication with stakeholders, the challenge of making the stakeholders understand the process and not forget it was raised:

“I usually bring up the skateboard. First you build a skateboard, then a kick scooter, then a motorcycle and lastly you build the car. That example is usually appreciated, it makes it easy to refer back to it when they have forgotten it.

Although both software developers and product owners expressed that they feel like they educate the stakeholders, it was the product owners that could be observed doing it in the actually meetings and Discoveries. While not being complete translation work between the stakeholders and developers, the product owners definitively help clarify the process to the stakeholders. This education can be consider similar to what Åkesson & Henell (2014) suggested; by teaching stakeholders about the process they use knowledge-sharing to lower the barriers of professional language to improve collaboration. Similarly, the interviewees expressed that stakeholders who have worked with them a lot before was easier to communicate with as they understand the process, something that aligns with the research of Geijer (2003) which showed that through continuous interactions the professions would pick up each others professional language.

However, the largest translation work that the product owner does is actually translating from the stakeholder to the software developers. As one of the product owner said, referring to different departments in the Company:

“I have a better ability to understand the individual perspectives from Commercial, Operations and Medical. It might be because I have a different background, education and/or training than the developers to understand these driving forces. So I try to translate the commercial driving forces to
What mainly needs to be translated from the stakeholders is the context around a request. The product owners recognize that there is a stark difference between how a developer and business people think; the developer focus on creating a great technical product and beautiful code, while the stakeholders want to maximize profit. These different mindsets are what the product owner has to translate, by explaining the context of a request and connecting it to a product need. This could be referred to as communication barrier of perception as expressed by (Kelly 2000), where the product owners help lower the barrier by translating the intentions of the stakeholders to the developers. One of the product owners expressed it as follows:

"Rather than communicating the feature request is communicating the context around the feature request."

The difficulty to understand each others driving forces and thus requiring a product owner to translate can be connected to the professional cultures in these different groups (Hall 2005)(Schein 2016). Research suggest that understanding these driving forces can enable better communication between the group; in this case it is the responsibility of the product owner to understand the different sides to properly translate between the groups and thus ensuring better communication.

By understanding the context and reasons behind a feature request, the software developers and product owners see an improvement in development speed and quality. The right thing is built faster from the start, and the software developers build things that the stakeholders and users actually needs. One of the product owners even expressed this as being one of the main indicator of a job well done as a product owner.

"If I have done my job well, then I have helped specify so that the developers understand the users need and how something is meant to be used."

In the Discoveries with both stakeholders and developers present, it was clear that while the product owner do not control the conversations they do steer them. They are able to steer the direction of the conversation to ensure that both the stakeholders and the developers get what they need from the communication. This shows that the
product owner is capable of understanding the needs on both sides of the communication, but it also leaves a lot of room for mistakes and misunderstandings if something is forgotten. During the Discoveries at the Company, they sometimes (some teams more often than others) use a template to ensure that the meetings are structured in the right way. Despite that, it is not fully or always used as it is not something that fits all meetings they have. It is sometimes too specific in some areas that do not need that much attention and then the teams feel it is obnoxious and take too much time to use.

4.2 Point of Contact

In alignment with Scrum theory, the empirics confirm that the product owner is the one facet that the stakeholders (mostly) turn to. When they become the representative of the stakeholders towards the team (Sverrisdottir et al. 2014), then they also become the representative of the team towards the stakeholders. This makes it easier for the stakeholder to know whom to turn to, solving the communication barrier of absence of formal communication channels mentioned by Kelly (2000). It was also consider one of the main benefits of having a product owner facilitating the communication between stakeholders and developers, as expressed by one of the developers:

“The product owner contributes the most by being the point of contact. It is not a healthy work environment to have the stakeholders run and disturb the developers, knock them on their shoulders. It is not a healthy work environment for anyone.”

When the stakeholders turn directly to developers with requests or questions, they get disturbed and lose their focus. With the product owner as the main point of contact, the developers do not need to make context switches as often and can faster implement and deliver the features requested. It lessens the workload issues of the developers, and it is something that a lot of them feel strongly about. They express that they want to interact with the stakeholders as little as possible outside of understanding the scopes or requirements in features.

“I try to turn down as many meetings as possible. Even if it is only 5
minutes it is a context switch, and those 5 minutes might cost a whole hour of my productivity. I don’t enjoy that, it is a real pain, and that is why I have set up time slots in my calendar where people can’t book me.”

So with the product owner there to take care of a lot of the initial communication and go to the initial meetings, everybody expressed that it enables the developers to better focus on their job and develop software. In other words, the product owner being the point of contact enables everyone to work more efficiently.

“We would have to be involved in a lot more alignment about motivation and product shaping if we wouldn’t have someone doing this work for us. I think that would take a lot of time and could hurt efficiency.”

The interviewees also see it as beneficial to the relationship with the stakeholders, as it is easier to maintain and keep up a good relationship with one representative than a whole team. With only one person responsible for the communication, it help build trust between the departments and teams.

However, it was not only seen as a good thing that one person was responsible for having a good relationship with the stakeholders.

“If the product owner always is the conduit between stakeholders and developers then it becomes second hand information that needs to be translated and interpreted. Like a game of “Chinese Whispers”.

One of the product owners also said that if the developers have a good relationship with a lot of trust with the stakeholders, it also makes their job easier when translating needs and making privatizations.

Furthermore, while the developers appreciate not being involved too much too early in the process, they also see problems with not contributing to the scope of the problem with their expertise. Both product owners and software developers recognize that if the developers are not there early in the process and really understand why something is meant to be built then it will be costly to change.

“Sometimes you get into the process too late and then it’s really hard if you discover something. It is not gonna be great to change it then and it will be more costly to change.”

Similarly to the solution of the communication barrier of information
overload (Kelly 2000), it is important to get a lot of information but the information that you get needs to be the right information.

When the interviewees were asked to describe examples of good and bad communication, it often came back to point of contact. However, it was the lack of clear point of contact in the stakeholders that caused a lot of problem. In the situations where they described it as being bad communication they complained about not knowing who was responsible for the project and not having a clear structure.

“*There was no clear lead of the project I think which made the planning really hard.*”

Similarly, in the examples of good communication they mentioned clear structure and division of responsibilities. One of the developers suggested that to improve the communication in projects, the stakeholders should have one person responsible from their team to always handle the communication with the development team and their product owner.

“A common problem is that we have the development team with a product owner, and then we have a lot of different stakeholders in different groups. If we could have a corresponding person on their sides that collects all the communication flows and speaks with the product owners, then I feel like we could save a lot of time for the same reasons that we have a product owner. Maybe a person on their side that is a bit more technical, then that person would take more responsibility as well. It is nice to be able to demand something back, and that is difficult to do to a group of people.”

Again, this would help lower the barrier of communication that refers to the lack of formal channels (Kelly 2000). By setting up two clear points of contact, it will be easier for both sides to know who to contact and talk to regarding a specific request.

Another challenge in the communication with stakeholders is that the stakeholders often come up with a solution prior to talking to the development team. Sometimes, that solution is very wrong as they have a faulty perception of how the product works behind the scene, and something that the developers would identify in a second as a small fix becomes a big project.

“*People very quickly have solutions ready for problems and they think this is*
the solution for the problem. But often if you go back to the origin of the problem then you might have a different perspective as a developer and see other solutions than what the stakeholders have suggested.”

If the communication fails and the teams fail to understand the original problem, then the stakeholders usually have to go back to the drawing boards a couple of times before they get it right. This causes a lot of delays and often a lot of frustration for the developers as they feel like the end product is worse than it would have been otherwise. As they have expressed, the quality of the written software and product is very important to the developers so this is something they want to avoid.

“We developers are usually the ones that really feel the pain of misshaped poorly planned projects.”

What the product owners try to do for the development team to solve this situation is by always making sure that the stakeholders do not only present their thought-out solution, but first explain why this is a problem in the first place. This way the developers and product owners can make use of their expertise and if needed find a better solution than was originally suggested. This problem of stakeholder bringing an already finished solution to the table could be considered just another version of the problem of the developers being involved too late or the problem of the stakeholders having difficulties expressing the context of a request.

### 4.3 The Art of Saying “No”

One of the greatest challenges for Product Owners in managing the communication between software developers and stakeholders is to know what requests to say no to. There is a lot of pressure on the product owner to know what request that will make the product better and bring more value to the users, and compare different request against each others to make a prioritization for the development team. Requests that the product owner that feels is not a good direction for the product or is something that is a good idea but has low prioritization compared to other requests at the moment, will need to be rejected (Sverrisdottir et al. 2014).
“You can’t be a people pleaser as a product owner, it will not work. There is a lot more no than yes.”

It brings relief to the developers in their daily work that the responsibility lies with the product owner to do the prioritization and ultimately decide whether the development team should do the request from the stakeholder. Being bothered by stakeholders that come up to them and ask for different things is something that greatly affects their focus and ability to develop software; having a product owner as a kind of firewall for the request is something that the developers greatly appreciate.

“Before our team had a product owner a lot of people came directly to me and my colleagues in my team and peppered us with questions and feature requests. But when we got our first product owner everything changed and we got more focus.”

Moreover, having a product owner as the one responsible for making the prioritization not only decreases the amount of request that the developers directly receive, it also gives them a way to deal with those stakeholders that still turn directly to them. As one of the developers put it:

“It makes it easier for me when I say no to a stakeholder that comes and screams that their thing is the most important one. It is nice to be able to lean on the fact that it is not my decision.”

The product owners also see great advantages of this, as they notice the decrease in stress that it causes the developers. As they see it, a lot of the developers are too nice to say no and then the work pressure will build up as they accept more and more requests. This might also include requests that are bad for the product and not aligned with the plan for it. Sverrisdottir et al. (2014) agree with this and say that the product owner need unrestricted authority regarding prioritization, otherwise the danger of misunderstanding the requirements is prominent.

“Sometimes I feel there is a risk that someone would be too pushing and force their will on the developer as they can’t say no. Then I rather it go through me.”

For the product owners to be able to make the prioritization, it re-
quires that they have an overview of the product, all the requests and
the overall direction of both the product but also the whole company.
They need to understand the goals of the company and product, and
be able to see the value in requests to be able to rank them.

“It is all about making sure that the team is working on the right thing and
that you always prioritize the thing with the most business value. If you
think about it in those terms it should be pretty easy, but in reality it is very
difficult to know that brings the most value.”

By being the point of contact for stakeholders enables them to get the
overview of everything that they need to make prioritization. This is
beneficial since the product owner can make better and faster decisions
and better long going plans in contrast to what the Scrum Master can
do who focus more on short term and how to implement the requests
(Judy & Krumins-Beens 2008). By having the product owner focusing
on the long term, it allows the rest of the Scrum team to completely fo-
cus on what is important right now. As one product owner expressed
it:

“We can know we make decisions that we stand by. We don’t look at a piece
of code that we wrote six months ago and think “oh my god I regret
everything about this and we will feel the pain of that forever more”.

The developers also appreciate it, as they also like to know and feel
like there is a purpose to what they do. As De Ridder (2004) found,
the higher quality of task-related information, in this case understand-
ing the purpose and know the long term effect of it, the higher com-
mitment and satisfaction they feel towards the company. It also puts
more trust in management, or in this case the product owner and that
the product owner will lead them in the right direction.

Another consequence of the developers not being responsible for the
prioritization is the situations when the product owners can not at-
tend a Discovery. In the Discovery attended that lacked a product
owner, the software developers would often come back to the fact that
the product owner was missing and that they would have to table a
discussion until they could check something with the product owner,
simply because they were very reliant on the product owner knowing
everything. This puts a lot of pressure on the product owner if the
team can not move forward without their constant presence, making
it important for the product owner to properly communicate things to the developers.

Moreover, as the product owner becomes a representative for the stakeholders to be able to do the prioritization and keep the product in line with the company visions, the developer sometimes feel like the product owner do not let them do important development work. A large part of programming is code refactoring, meaning improving the already existing code by restructuring it without changing the external behaviour. It is something that is important to keep the program working smoothly, but also to prepare for the implementation of additional features. The interviewed developers felt like this was one major challenge in communicating with their product owners, as one said when asked about what is difficult to explain to the product owners:

“When it concerns tech things that basically don’t change anything visible, but for us developers it seems to be a very important issue and might benefit the company in the long run. Then it is really hard to make a case against for example implementing a new feature because that is visibly creating value. But if you refactor something in the background that might not be visible, it might reduce blocks and issues in the future”

It is a bigger problem to get time to refactor code if a product owner is not very technical, then there needs to be a lot of trust between the product owner and the development team. This situation demands that the product owner really can say no to the stakeholders as to give the developers time to do the non-visual but very important work. One of the developers mentioned that if they do not get the time to do so because the product owner does not trust that this really is important, then they have to go behind the product owners back while doing it. This worsen the relationship between them, making the product owner even less inclined to trust them when they in the future ask for time to make a big code refactoring. The product owner also identify the difficulty to give time to the developers to do this kind of work, and elaborated that it is easy to just do whatever the stakeholder that screams the loudest wants and that the product owner actively have to consider the developers side to prevent forgetting about that kind of work.

If the product owner was not there to push for new features instead of letting the software developers do all the pure technical things that
they want, then the developers acknowledge that they would probably focus too much on refactoring. Not enough attention would be put to satisfy the stakeholders and fulfill company goals, and they would focus too much on doing things without visible value. Thus, a product owner helps the developers to move the product forward.

One of the developers talked about previous experience at another company, where the product owner was not part of the team and how that negatively affected the quality of the product and the developers that had to implement different features. Those product owners would do whatever the loudest stakeholder wanted because they considered themselves closer to the stakeholders than the product and development team. A couple of the interviewed developers emphasized the importance of the product owner being a part of the product team because then they think more product than business and make sure that the product goes in the right direction.
Chapter 5

Discussion

In this chapter findings from the previous chapter 4. Results & Analysis are discussed and compared to theory. It will also include a reflection on sustainability and limitations of this study as well as suggested future research.

The purpose of this study is to describe the impact that a product owner has on the communication between software developers and stakeholders. By answering the research questions through thematic discussions on relevant findings, a greater understanding of how software developers and stakeholder communicate and the impact that the product owner has on it can be achieved; it gives insights into how empirics and different fields of theory correspond, as well as when conflict between theories and practice arise.

As a reminder, the research questions are as follows:

- What are challenges and opportunities in the communication between a software development team and stakeholders?
- What are challenges and opportunities with the product owner facilitating the communication?
- How does the above affect the software development process?

Something that became obvious in the observations and interviews was the huge role that the product owner plays in the communication; it became obvious how much both the stakeholders and developers rely on the product owner to handle the communication. While this is hinted by the theory, claiming that the product owner should be
the representative for the stakeholders (Judy & Krumins-Beens 2008) (Sverrisdottir et al. 2014), it fails to recognize how much more it is than just that.

**Translating processes, technical jargon and context**

One of the big impacts that the product owners have is from their work as translators where they help overcome hinders such as professional language, different cultures and perceptions. From the empiric it became clear that the stakeholders have problems understanding the software development process. It makes them disappointed when they first receive something that is less than they expected because the developers start by building the minimum viable product. It also upsets them when they are promised a deadline that the development team then can not keep as deadlines are hard to guarantee when working with Scrum. So it is a challenges in the communication, both for the developers and product owners to explain and teach the stakeholders about the process.

The stakeholders learn about the development process while working with the development team, but it would be advantageous if they understand it prior to working with them the first time. A possible solution to lessen the confusion about the process could be to explain it whenever a new employee joins. The Company have intro-days where all the departments come and introduce what they do to new employees, and it might be useful to also include an introduction of how they do what they do. By educating everyone about the communication and how the process works, the collaboration could improve as stated by Abrahamsson & Gredevik-Heinebrün (2009).

Surprisingly, professional language was less of a challenge than research implies it should be (Geijer 2003). The opinion of both developers and product owners was that they usually seem to understand, even if it sometimes requires the developers to have to work hard to explain. This could be due to the fact that they use symbols and visual tools such as drawing on the whiteboard to explain. By making even more use of visual tools, like whiteboard and digital presentations, it could help stakeholders and developers in their discussions even more (Williams 2002) (Kelly 2000).
However, this opinion of the relative lack of barrier of professional language could simply be due to the fact that this study did not interview stakeholders. It could be that the developers and product owners are so used to working together (Geijer 2003), that they simply do not recognize when they say something that might be incomprehensible to stakeholders. This could therefore be completely different if the stakeholders were asked. In those cases when misunderstandings happens, then the software development gets affected with worse scopes and more badly planned projects. In turn, that could worsen the relationship between them and make it even harder to communicate in future projects by rejecting each others competencies, as a risk expressed by Geijer (2003).

The largest impact of the product owner as a translator, is when it comes to determining the context of a request. Apparently the software developers sometimes find it challenging to understand the original problem behind a request and this is something that the product owner can help with. The challenge to understand the underlying reason can be related to professional cultures and the differences between the culture of the software developers and the stakeholders (Schein 2016)(Hall 2005), as well as the communication barrier of perception (Kelly 2000) as the software developers struggle to understand the intentions and motives of the stakeholders. In these cases the product owners expressed that they are better at understanding the culture of stakeholders and thus the underlying reasons for a request, because they have a different background than the software developers.

Furthermore, at the Company they would sometimes use a meeting template as to ensure that everything that needs to be covered is covered. This one is however not very regularly used, at least not fully used. By using the template more or by creating and using a checklist that covers what all the participants in a meeting need to do before, during and after the meeting, they could reduce the amount of misunderstandings. In the operating room health care personnel have applied the WHO Surgical Safety Checklist, which has proved to cut cost and improve the safety of the patients (Semel et al. 2010). The checklist used in the research of Lingard et al. (2005) was also proven to be a successful way to promote information exchange and team cohesion in interprofessional collaborations. There exists similar checklists as the WHOs one for software development. One example is the Essence
Alpha Cards that consists of checklists for the whole software development process (Jacobson et al. 2019). However, there is a lack of research on how these checklists affect the communication between stakeholders and software developers, although it has been proven to improve the overall software development process (Kemell et al. 2018).

The effect of having a product owner translating the communication between software developers and stakeholders is that it enable the developers to more properly understand the context of a request. This reduces the development time and the risk of developing something unnecessary and bad for the product. Also the developers enjoy work more and thus feel more committed to the organization as the quality of task-related information improves (De Ridder 2004).

**A Point of Contact to Clarify Formal Communication Channels**

Software developers find it both annoying and distracting when stakeholders contact them directly with requests. It is something that have direct negative effect on the software development process as it takes longer time for the developers to finish a request as they get interrupted all the time. By having the product owner as a point of contact makes it easier for the stakeholders to know who to talk to and thus bother the developers less, lessens the barrier of lack of formal channels (Kelly 2000). Furthermore it also gives the developers an excuse when an astray stakeholder turns to them with requests.

A great challenge was identified in that the stakeholders commonly do not have a single representative that act as their point of contact. No corresponding person as their “product owner”. The lack of a clear leader often results in misunderstandings when shaping the scope of the product request, resulting in a longer development time and more unnecessarily developed software. By assigning one of the stakeholders from each of their group as their point of contact could greatly improve the communication. This would not only improve by reducing the barrier of lack of formal channels (Kelly 2000), but also by allowing that person to better understand the software development process and the professional language through continuous interactions (Geijer 2003).

However, having the product owner as a point of contact was not only
seen as being advantageous, and this is due to the fact that it tends to remove the positive opportunities of the communication between software developers and stakeholders. As the product owner deals with the initial shaping of the feature requests and that the product owner go to meetings by themselves, the software developers lose the opportunity to early in collaboration with the stakeholders identify and come up with advantageous solutions for the original problem using their expertise. In the cases where it goes wrong, it results in a worse end-product or a higher development cost as it cost more to go back and change whatever was wrong later in the process. This poor communication is identified as one of the most common factors for why software fails (Charette 2005).

Another problem that the developers experience is that the stakeholders often come to them with a prepared solution, just a finished feature request. This could be consider being a combined effect of that the developers get involved in the process too late and an effect of that the stakeholders are bad at explaining the context behind a request. That way the stakeholders have come too far in their scoping of the request, and it becomes more expensive to go back and change something when the developers discover something faulty. The inability to handle a project’s complexity is also one of the most common factors for failed software (Charette 2005), and something that might happen if the ones with the technical understanding, i.e. the developers, get involved too late to grasp the complexity of the stakeholder’s request.

However, with the ambition of giving the developers as much of time as possible to do their development work, without too many context switches, it might be important to evaluate how much information is too much information. When does information-overflow become a barrier for communication for the developers, and what can be done about it? According to Kelly (2000) too much information is in fact not a problem, but it is important to only take in actually necessary information. Then the next question becomes how to determine what is important information. This relates back to the template for the meetings (Semel et al. 2010), and the usefulness of a checklist to confirm what needs to be known and communicated (Lingard et al. 2005). Again, the Essence framework with their Alpha state cards could be used to simplify and make the communication, and thus the information, more

**Saying “No”, Prioritization and Non-Visual Values**

An important result of the communication with stakeholders is the requests that the developers need to implement. It might however be difficult to determine what is most important and has the highest priority; this is one of the largest challenges and responsibilities of the product owner (Judy & Krumins-Beens 2008)(Sverrisdottir et al. 2014). The difficulty lies in that they have to tell stakeholders no and explain that their request either is too bad for the product or of so little value compared to other requests that it will not be of high prioritization in a long time. However, the support of handling this by the product owner offers a lot of relief for the developers as previously discussed.

To be able to make this prioritization, the product owner needs to have an overview of everything that goes on related to the product. There is a lot of pressure on the product owner to do this well because if they are not successful it will lead the product and the development team in the wrong direction. In the context of the product team, the product owner has almost become an irreplaceable asset as they are the only ones with all the information and the power to decide over the product backlog. This by itself might become a communication barrier when the product owner can not attend a meeting, as the stakeholders and software developers becomes hindered and limited in what decisions they can take. If it is only once in a while, then this does not have too much of a negative effect on the overall development process except that it gets delayed a bit. However, the question then becomes what would happen if the product owner for some reason can not perform their work? Will the work quickly pile up for the developers and prioritization derail as they do not have as much information as the product owner? Perhaps the solution would be to have collective product ownership as in the study of Judy & Krumins-Beens (2008), as the whole team including the product owner then is responsible and knowledgeable about the prioritization. However, as this has yet to be proven high-performing, the repercussions on the software development process by implementing it might worse than the odd-case of the product owner disappearing.
As the product owners always have to prioritize and deal with the requests from the stakeholders, the developers seem to feel that the product owner sometimes fail to allow them to do other types of important development work. These non-visual improvements of software are very important to continue to have a high quality software product (Fowler 2018) with a good architecture that allows for future modifications. By having trust between the team and the product owner, the developers identified that they were able to get more time to do this type of refactoring of code as the product owner trust their judgment in that it will bring a lot of value to the product.

### 5.1 Reflections on Sustainability

Throughout the study the three pillars of sustainability were considered; environmental, economic and social sustainability. The aspect of environmental sustainability is not considered to be directly affected by the outcome of this study. Since neither the processes, the people or the communication affect the environment in the context of this study, this pillar becomes irrelevant for this case.

The economic pillar of sustainability is more relevant in this study. As stated by Hansen & Press (2009), good collaboration between teams is a factor that can mean the difference between a fail or a success. The outcome of this study can help improve the communication and the collaboration between software developers and their stakeholders, thus creating better software with higher success rate in shorter development time. This results in lower development costs for the companies and in an extension better economical sustainability for companies with software or software-reliant products.

However, the main contribution to sustainability lies in the social sustainability. As the phenomenon of this study is a very social one, it has more potential to greatly affect this pillar. Through using a product owner to enable software developers to do their development work in peace and quiet with a reasonable amount of pressure on them, while knowing the vision of the product and the importance and impact of their work, it greatly improves the work environment for the developers. It also keeps them more engaged and satisfied with the orga-
nization they work for. Furthermore, having a product owner facilitating the communication and being a clear and easily accessible point of contact for the stakeholders, makes it easier for stakeholders to do their work. The contact with the development team will not only be an unpleasant necessity, but rather a smooth process that makes their work more enjoyable. These are all contributors to social sustainability at organizations, and the outcome of this thesis can help guide product owners, stakeholders and developers alike in the communication between them. The joy of being part of a successful delivery of a product that is important for the company, is hard to beat!

5.2 Limitations & Future work

There are a couple of limitations with this study, an important one being the limited number of interviews with software development and product owners. To have more interviews, especially at other companies, would have given a wider perspective of opinions. It might also have given insight into how the result might differ in a much large organization, a bigger enterprise. One suggestion for future research is therefore to repeat this study at different companies.

Furthermore, the results might have been different and more extensive if stakeholders had been interviewed. Different challenges and different opportunities might have been identified, and also different suggestions for ways to improve the communication. Another suggestion for future research is thus to do a study on this subject completely in the perspective of a stakeholder.

Lastly, it would be interesting to complement this qualitative study with a research that uses quantitative methods to collect data. It would make it possible to better generalize the result and make it applicable on a greater scale.
Chapter 6

Conclusion

The final chapter presents the final part of the thesis and concludes it. The conclusions are presented with a primary focus on delivering concise answers to the research questions and some implications of this study.

The purpose of this study is to describe the communication between software developers and their stakeholders as well as the impact that the product owners have on the communication and how the communication affects the software development process. This was done through observations and interviews with product owners and software developers at the case company Kry.

One of the main themes identified was that the product owner becomes a kind of translator between the stakeholders and the software developers. In a lot of cases the stakeholders do not understand the software development process; this results in disappointment, wrongful expectations and unnecessary pressure on the developers which negatively affects the software development process as it might be more difficult to fulfill the requirements. It also worsens the relationship between stakeholders and the team. The product owner and software developers tries to educate them whenever this occur, but the stakeholders commonly forget it again until the next time. By educating them more structurally about the process, e.g. when new employees start working at the company, these types of misunderstandings might decrease. The professional language and technical jargon of the developers were deemed to be less of a challenge than anticipated; which might be because of the use of whiteboards and presentations....
to visually aid the communication when the professional language becomes a barrier. However it might also be because the stakeholders were not interviewed in this study and that the product owners and software developers work very closely together and thus are used to the in-group language. The biggest translation work for the product owner is the translation of context behind a request from the stakeholders. This becomes the responsibility of the product owners as they commonly are closer in work culture to the stakeholders than the developers are. The positive effects on the software development process of having the product owner translate the context is that it enables the developers to understand what is important to the request and thus build better and more fitting software already from the start. One way to improve and guarantee that everyone gets what they need from a meeting could be to rely on checklists such as the ones by Essence (Jacobson et al. 2019).

The second main theme was the importance of having the product owner as a point of contact for the stakeholders. It makes it clear for them who they can turn to for answers, and it relieves the developers from being constantly bothered by the stakeholders by questions and requests. This allows the developers to focus on the development work, which cuts development time and costs. However, it might also result in the developers being involved too late in the process, so that they discover possible faults in solutions that the product owner and stakeholders have thought out too late. Thus, the software development process could become very costly, as the earlier something is discovered the easier it is to fix. Another challenge identified was that stakeholders rarely have their own point of contact, something that often results in misunderstandings when scoping the product requests. By implementing a point of contact on the stakeholders side this could be improved.

The last main theme discovered was the challenge for product owners to prioritize the requests and how this makes the product owners almost irreplaceable. There is a lot of responsibilities on the product owners to have a complete overview of everything related to the product as this is the way for them to make the best prioritization. If the product owner is good, this will lead to a successful product but if the product owner fails it will lead to a worse product. The developers also see it as a challenge to convince the product owner to let them do
important refactoring work, but with good trust between them the product owner is more inclined to allow them time to do so.

A lot of the main challenges that the software developers experience in their communication with stakeholders are improved or completely overcome with the involvement of product owners. This clearly shows the importance of product owners in the success of product development. Good communication can mean the difference of success and failure (Hansen & Press 2009), and through presenting the challenges and opportunities of this communication, the results of this study hope to contribute to the theoretical basis so that future research can improve the communication even further. Moreover, the results are not limited to the field of software development and could also be fitted to other fields that work in similar team constellations.

As a conclusion, the author is convinced that communication between stakeholders and developers powered by product owners will greatly enhance the possibility to build good, maintainable software and lead to a more sustainable work environment for all involved.
Bibliography


Babylon (n.d.). URL: https://www.babylonhealth.com


Blomkvist, P. & Hallin, A. (2015), Method for engineering students: Degree projects using the 4-phase Model, Studentlitteratur AB.


Czarniawska-Joerges, B. (2007), Shadowing: and other techniques for doing fieldwork in modern societies, Copenhagen Business School Press DK.


Fowler, M. (2018), Refactoring: improving the design of existing code, Addison-Wesley Professional.


God forskningsseed (2017), Stockholm.


Holcombe, W. M. L. (2008), *Running an agile software development project*, Wiley Online Library.


Jungbeck, A. (2018), ‘How can dual-track agile contribute to a better design process and workflow in larger teams?’.


Kry (n.d.). URL: https://www.kry.se/en/


Appendix A

Observation Template

This structure was written with the intent to ensure that nothing would be overlooked during the attended meetings and Discoveries.

Link to documentation available before the meeting:

Length:
Team:
Stakeholder:
Product Owner:
Participants:
About:
Social Environment:
Quotes:
My impact:
Observations:
Questions?:
Expected future observations:
Appendix B

Interview - Product Owner

These questions were written with the intent to assist during the interview. Although most were used, some were altered or skipped to the progress of the interview.

Introduction:
  - Explain the purpose of the interview and what the results will be used for.
  - Is it OK to record this?

About the interviewee:
  - Who are you?
  - What background do you have?
  - How did you end up in the position of a Product Owner?

Challenges:
  - What are the greatest challenges for you as a Product Owner?
    - What are challenges in communication?
    - Tell me about a situation
  - How do you transfer information between the developers and stakeholders?
    - To stakeholders?
Opportunities

- In the communication between the developers and the stakeholders, where do you as a Product Owner contribute the most?
- How would the communication between them be if there were no Product Owner to handle some of the communication?

Effects

- What role do you strive to take in meetings and or discoveries with both stakeholders and members of the development team present?
  - How do you facilitate their communication?
  - What is your impact?
  - How would it be if you were not there?
- What are the effects of Product Owner facilitating the communication?
  - Positive:
  - Negative:
- How does this affect the development process?
  - Challenges:
  - Opportunities:

Final

- If you could honestly say something to the stakeholders and your software developers, what would it be?
  - Stakeholder:
  - Developers:
- Is there something that you think I should have asked you?
Appendix C

Interview - Developer

These questions were written with the intent to assist during the interview. Although most were used, some were altered or skipped to the progress of the interview.

Introduction:

- Explain the purpose of the interview and what the results will be used for.
- Is it OK to record this?

About the interviewee:

- Who are you?
- What do you do?

Context:

- Describe different situations in your day-to-day work where you talk to stakeholders
  - What do you think that you get out of these meetings?
- What parts of the communication with the Stakeholders do the Product Owner handle for you and the rest of the development team?

Challenges:

- What are the greatest communication challenges for you?
– With Stakeholders?
– With Product Owner?
– Tell me about a situation

Opportunities:

• Where do you think that the Product Owner contributes the most in the communication between you and your stakeholders?

• How would the communication between you and your stakeholders be if there were no Product Owner to handle some parts of the communication?
  – Are there some parts of the communication today where you think the involvement of Product Owner is unnecessary?

Effects:

• What role do you strive to take in meetings/discoveries with stakeholders, Product Owner and other members of the development team present?
  – What impact do you have on the communication?
  – What impact do the Product Owner have?
  – How would it be if no developer was there?

• What are the effects on the communication with a PO facilitating parts of the communication?
  – Positive:
  – Negative:

• How does this affect the development process?
  – Challenges:
  – Opportunities:

Final:

• If you could honestly say something to the stakeholders and your Product Owner, what would it be?
– Stakeholder:
  – Product Owner:

  • Examples of good/bad communication?
  • Is there something that you think I should have asked you?