Agile Software Development:

Exploring the Values and Principles, Collaboration and Boundary Objects

A thesis submitted in partial fulfilment of requirements for the degree of

Doctor of Philosophy

Anna Laura Kristiina Zaitsev

Faculties of Business Information Systems and Work

The University of Sydney Business School

2018

Supervised by

Associate Professor Uri Gal, Business Information Systems

Associate Professor Barney Tan, Business Information Systems

Publications related to this thesis:

Chapter 4 draws on material published as "Zaitsev, A., and Tan, B. 2014. "Agile Project Done Right: Lessons from Project Betterproduct", in the Pacific Asia Conference on Information Systems

- I designed the study, analysed the data and co-wrote the paper. Authorship is based on contribution.

Chapter 7 draws on material published as "Zaitsev, A., Gal, U., and Tan, B. 2014. 'Boundary Objects and Change in Agile Projects' in proceedings of 25th Australasian Conference on Information Systems, Auckland, New Zealand and as "Zaitsev, A., Tan, B., and Gal, U. 2016. 'Collaboration Amidst Volatilty: The Evolving Nature of Boundary Objects in Agile Software Development', in proceedings of 24th European Conference on Information Systems, Istanbul, Turkey

- I designed the study, analysed the data and co-wrote the paper. Authorship is based on contribution.

Student Name: Anna Zaitsev

Signature: Date: 12.3.2018

As supervisor for the candidature upon which this thesis is based, I can confirm that the authorship attribution statements above are correct.

Supervisor Name: Uri Gal

Signature: Date: 12.3.2018

This is to certify that to the best of my knowledge, the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes. I certify that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged.

Student name: Anna Zaitsev

Signature: Date: 12.3.2018

In memory of Natalie Hardwicke

Acknowledgements

I originally began the project because I felt that I was lacking intellectual challenges. And truly did this thesis project deliver! Writing this thesis has been quite the journey but none of this would have been possible without my supervisory duo, Uri Gal and Barney Tan. I believe I have been very fortunate; not every student gets to work with two passionate and intelligent supervisors who perfectly complete each other. You successfully brought the best out of my work and pushed me to do better. We all got grey hair during this project but I believe it was worth it!

I would also like to thank the scholars and practitioners who have encouraged pursuing my topic and given me valuable pearls of wisdom. I am grateful for the support I have received from BIS faculty and other members of the Business School, especially from Dirk Hovorka, Deborah Bunker, Kai Reimer, Sebastian Boell, Catherine Hardy, Catherine Welch, Robert Johnston, Sharon Coyle and Allen Lee.

The Agile community has been very supportive towards my sometimes very theoretical and non-practical pursuits, willing to engage and participate in this study with little reservation. I would extend thanks to everyone in the Agile community who participated in the thesis as interviewees and gave me a bit of their time as well as the Agile gurus I got the privileged to have a discussion with, Pragmatic Dave and Uncle Bob. You are the MVP's of this thesis.

Thank you to Geoff and all my other BUSS3500 colleagues, thank you for the Monday morning meetings and all your hard work that also kept me employed during my time at Sydney University.

I am also thankful to the support from my co-sufferers, the CUBIST's. I will always be gratefu to have made a group of fiends who understand the daily grind of thesis writing and who get my bad TAM jokes. Helena, Miffy, Aaron, Wazza, GoBe Rosie, Nami, Siri and HoDo, you and other lovely people of the SRDL, have always provide me the needed distractions, hits to the head and beers at the pub that were truly required to in order to keep myself sane during this project. Salla, Henrik, Siobhan and Mattias, thank you for the all the fun times we have spent together when I have not been busy staring at the laptop screen.

Thank you to my family, supporting me from afar. Äiti, thank you for providing a bit of healthy thesis completion competition. Aarno and Julia, thank you for the academic insights and Paapa, thank you for always trusting that I know what I am doing. Sisters and brothers, thank you taking part in making me the person I am today.

Finally, I would like to thank my beloved husband and best friend Klaus. None of this would be possible without your eternal patience and support, insights and occasionally offered nuggets of wisdom.

Abstract

Agile software development, both a movement and a set of software development methods, guided by the values and principles state in the Manifesto for Agile development (Beck et al. 2001), should not be seen only as a collection of development methods. This thesis proposes that in order to understand what constitutes differences or similarities between Agile organisations, one should apply a holistic view of Agile development. This view consists of three elements that form the Agile development environment: the perspective the organisations have towards Agile, the Agile process the organisations follow and the Agile structures that form the basis for the process. By proposing this framework of Agile engagement, the thesis answers following research questions:

1. **How does collaboration in organisations differ under the various approaches towards Agile?**
2. **How do organisations structure collaboration between Agile teams and business stakeholders?**
3. **How are boundary objects used in collaboration between these groups?**

As research method, I followed the eight steps of the SPS approach and conducted a qualitative, interpretive case study. The study consists of three case studies. First, the Extreme Inc. case is an example of an organisation where the members hold an Avid perspective towards Agile methods. The organisation had arranged the collaboration to follow a tightly integrated model where boundary objects are applied to support the pair programming method and foster face-to-face collaboration.

The case of Escapade and Carmine presents an example of an Inclusive Agile perspective, where organisations strive to focus on collaboration and boundary mitigation. The organisations had set up a collaboration configuration, where boundary spanners and all boundary objects were designed to create a sense of presence and ease of collaboration.

The last case, the PrecautionCorp program, is a study of an organisation where the members of the organisation have chosen to observe Agile methods in a Pragmatic way. All collaboration between the stakeholders was organised via selected boundary spanners who mitigated the boundaries but also maintained a level of control over the chaos by applying a variety of boundary objects.

By analysing the three case studies, I have identified three main elements that impact Agile software development: the perspective the organisations have towards the Agile philosophy which impacts the Agile process, that is, how the Agile methods are applied in practice; the configuration of the business stakeholder collaboration; and the application of the boundary objects. Based on this analysis, I have presented the framework for Agile engagement, a holistic theory that tied together the elements of Agile development.

The practical contributions of this thesis are several: practitioners can apply the framework for Agile engagement when analysing their own positions, can benefit from better understanding of the relations between Agile process, Agile perspective and Agile structures, and can enhance their understanding of the best possible application of boundary objects.

Table of Contents

Figures

Tables

1. Introduction

> *Many, many, years ago, in 1999, when the 'Extreme Programming' book first came out, my Team Lead at that time lent me a copy of the book and I read it and thought: 'this is amazing – we have to do this!'*
>
> *– Head of Engineering, Extreme Inc., 2017*

Robots and AIs, social media products and complex algorithms fascinate people and entice artists to tell stories of our struggles to understand modern software technology. But the methods, tools and complex teams behind the creation of these complex systems are less familiar to the general public. The deep ethical and philosophical discussions of empowerment and craftsmanship of software development can be as intriguing as the end products, but this side of development is often discussed only by the people who are already in the know, initiated into the world of software development. However, a specific stream of software development methods, Agile software development methods (Beck et al. 2001), is breaking into new areas of application outside the software development world, into design, marketing and strategic planning (Rigby et al. 2016). The umbrella term 'Agile software development methods' refers to a set of software development methods that follow the values and principles of the Manifesto for Agile Software Development (Beck et al. 2001).

The Manifesto for Agile Software Development (Beck et al. 2001) was originally penned by members of a grassroots movement, in order to influence the wider software development community. The Manifesto, along with the ensuing rise of the Agile development community, is one of the most substantially influential events impacting organisational structures and cultures in the last few decades. In recent years, Agile development has become prevalent not only in the software development industry, but also in information technology departments across multitudes of organisations. The early adopters of Agile, such as finance and telecom industry departments, are currently being followed by the public sector, retail and industrial arts industries, as well as product development and even radio programming (e.g. Wang et al. 2012; Scrum Alliance 2015; Seymour & Coyle 2016; Rigby et al. 2016; VersionOne 2017).

Yet, despite the popularity of the methods, common issues in software development projects, such as late deliveries, inadequate product quality or customer dissatisfaction, persist. According to a survey by VersionOne (2016), only 53% of software projects using Agile software development methods were seen as successful. The most cited reasons for Agile project failures are related to company culture, lack of management support, and other internal communication and collaboration problems of the organisation (VersionOne 2016). To ensure success and collaboration, organisations should overcome organisational boundaries between the Agile development teams and the other stakeholders (Boehm & Turner 2005).

Software development organisations that aspire to trial and adopt Agile development methods, or who wish to extend their Agile practices outside their IT departments, need to be aware of the organisational changes that can be

prerequisites or consequences of adopting the methods (Conboy et al. 2010; Rigby et al. 2016). There is a risk that the organisations that lack an understanding of the core principles of the Agile Manifesto, will struggle with the implementation of Agile and fail to produce the results they expected (Rigby et al. 2016).

The Manifesto presents a radically different worldview of project management and challenges many entrenched ideas of how projects should be managed. The Agile Manifesto contests the values of traditional development methods and strives to be at the 'edge of chaos', rather than requiring order (Scrum Alliance 2014). For example, in traditional project management, change has been seen as an adverse, disruptive force or a risk to project progress (Boehm 1988). The traditional perspective of change is that it should be avoided, if possible, especially in the latter stages of the project. The reason for change aversion in a plan-driven project environment is the potential domino effect that can start from smaller changes and create extensive rework and replanning (Highsmith & Cockburn 2001). The unpredictability of changes in software development makes it hard to build adequate contingencies into project plans and contracts (Larman & Basili 2003).

The Agile Manifesto takes a very different approach to change. The Manifesto proclaims that projects should prefer changes rather than planning activities (Beck et al. 2001). Agile practitioners recommend that change is 'embraced' and incorporated into the project as a source of potential improvement (Beck 1999). Change can be used to encourage creative solutions because the development team is not hindered by flawed plans. Such creativity can facilitate a faster creation of systems that provide business value for customers (Boehm & Turner 2005).

Understanding Agile software development stems from understanding the goals of the Manifesto for Agile Software Development. The values of the Manifesto shift the perspective of software development from managerial control towards collaboration, and from development tools to humans utilising the tools (Cohn & Ford 2003). The collaboration between customers and developers in the requirements prioritisation process is a way of assuring that the customers get a working system with the most important requirements. Any long-term plans are left deliberately vague to allow changes to occur (Schwaber 2004).

The promise of Agile development with its focus on collaboration and faster delivery of results, can be enticing for any organisation struggling with deadlines and siloed departments. Collaboration across organisational boundaries is challenging and even if Agile development methods are applied, organisations still struggle with speed of development and customer engagement (Martini et al. 2016). Solutions to these challenges are often related to strategic application of artefacts that can be used to bridge the boundaries. For example, Agile task walls can be applied to synchronise tasks performed by the various members of the organisation (Strode et al. 2012), and burndown charts and user stories can convey information in virtual collaboration (Bass 2016). Even though the Agile Manifesto downplays the role of artefacts and emphasises the human aspects of the development, research shows that artefacts, which support the development efforts, have a significant role in enabling collaboration (Strode et al. 2012, Martini et al. 2016), especially when it comes to virtually organised teams (Marheineke et al. 2016).

Boundary objects are often used as a theoretical lens by researchers who study collaboration that crosses organisational or team boundaries. Boundary objects are artefacts that are applied to convey meaning and enable collaboration between different parties (Star & Griesemer 1989). The parties can employ boundary objects to communicate status information, discussions and agreements (Barrett & Oborn 2010; Yakura 2002). Objects may

vary in form and information capabilities, with different configurations that are appropriate for conveying distinct types of information (Levina & Vaast 2005). The definition of boundary objects states that boundary objects have properties that allow them to be both abstract and concrete, depending on how they are applied. The objects are malleable and flexible when applied to fulfilling the needs of different local parties, but when they are applied in individual use, they are robust and strongly structured (Star & Griesemer 1989). This polymorphic nature of boundary objects makes them especially suitable for supporting the diverse collaboration requirements of Agile software development projects. For instance, boundary objects can enable project stakeholders to visualise and clarify software designs (e.g. Gal et al. 2008; Winkler et al. 2014). Despite their apparent usefulness in facilitating collaboration (Dybå & Dingsøyr 2008), the role and nature of boundary objects in the Agile software development setting has not been studied to a significant degree. There is only a handful of articles to date discuss the impact of boundary objects in the context of Agile development environment (e.g. Strode et al. 2012; Winkler et al. 2014; Martini et al. 2016) and their focus is not on the challenges posed by the unique nature of said environment.

The unique nature of Agile development itself is another topic which lacks theoretical discussion (Hummel 2014). There are very few papers that seek to create unifying theories of Agile development which would reveal the 'defining constructs' of Agile (Conboy 2009; Drechsler & Ahlemann 2015). I wish to critique the approach taken in such efforts; I believe that authors who strive to create a unifying theory view Agile development from a reductionist perspective. The common denominator of Agile methods is unlikely to be found by investigating only the best practices of Agile methods. These methods vary greatly in their extent and complexity. Some methods prescribe very minor changes, such as different visualisation techniques, but some suggest that organisations should implement more intricate and profound changes, such as new roles of Product Owner or Scrum Master (Schwaber 2004) that can, when applied, inflict substantial organisational alterations (Cohn & Ford 2003). As Conboy remarks (2009, pp. 330):

Some represent prescriptive operational instructions for developers (i.e., XP), others bear closer resemblance to project management methods rather than ISD methods per se (i.e., Scrum), and yet more can best be described as a set of philosophical principles (i.e., Poppendieck's Lean Software Development). In extreme cases these methods are even contradictory.

Instead of analysis of the methods, I suggest that holistic, unifying theory is better distilled from investigation of the understanding of Agile and its core values by the organisations which apply the Agile methods.

To date, there is very little academic literature that analyses the values and principles of the Agile Manifesto and the worldview they represent. Nerur et al. (2005), Nerur and Balijepally (2007), and Conboy (2009) are among the few authors who discuss Agile methods as a philosophical rather than only practical approach to software development. However, neither study provides unifying, theoretical insights into the subject matter; rather, the majority of academic research into Agile development circumvents the subject of Agile as set of values with philosophical underpinnings and instead, highlights different paths to Agile adoption or focuses on the development teams and methods (Zaitsev et al. 2016). The values and principles of the Agile Manifesto are rarely examined, mainly referenced when discussing the origin of the methods. I believe that the collaboration between the Agile teams and their internal business stakeholders becomes an even more pertinent subject now that Agile methods are becoming

mature in the software development industry (West et al. 2010) and are increasingly being adopted by non-software development businesses (Rigby et al. 2016). Without holistic understanding of the underpinnings of Agile development, the new adopters of the methods might have to unnecessarily repeat the struggles of the early Agile adopters (Dybå & Dingsøyr 2008). In addition, understanding of the material aspects of work is another element that has been increasingly studied in the field of Information Systems (Hafermaltz & Riemer 2015); however, this enthusiasm has not yet spread to the research of Agile software development methods.

It seems that every few years an Agile literature review, journal paper or editorial calls for more studies of what 'agility' really means (Abrahamsson et al. 2003; Conboy 2009; Goh et al. 2013; Hummel 2014). Next, I will introduce the research questions that I chose to use as my navigational tools in search for the answer to 'what is agility'.

1.1. Research Questions

This thesis focuses on creating a holistic perspective of Agile development environment, a view that encompasses both the defining values and principles of the Manifesto and the multitude of practical method descriptions. The Agile software development environment is not only a collection of methods, but also encompasses Agile values and principles, Agile methods, artefacts that support Agile activities, and the variety of stakeholders who participate in the software development process. Analysing the different pieces of the Agile puzzle separately would not allow a comprehensive understanding of the phenomenon. By separating the Agile Manifesto from Agile projects, the original rationale for Agile development becomes less clear. Agile values and principles have to be included as part of the holistic view, along with the more concrete elements such as application of a selected set of Agile methods or artefacts that support said methods.

The aim of this thesis is to shed more light on collaboration and on the different views on Agile that are held by the members of the organisations that apply different Agile methods and subscribe to different views on Agile development. The extant academic literature on Agile development indicates that there are different types of adherences to Agile values and principles. Literature presents a multitude of case study examples where the case organisations subscribe to different sets of Agile methods and apply the methods in their own ways. Moreover, one can also analyse the way the authors of the papers view Agile development; how they describe what Agile is and what constitutes application of Agile methods. Some authors have researched Agile from a practical toolkit perspective, whereas others have proclaimed that there are deep-rooted philosophical elements entangled within the Agile software development environment (see 2.4). I was interested to see if my case organisations would follow similar patterns and view Agile from differing perspectives, and if these differences would have an impact on the application of Agile methods.

Furthermore, I wished to investigate how the members of the Agile teams collaborate with other Agile and non-Agile teams within their own organisations, and how the Agile team members collaborate with stakeholders across organisational boundaries. The Agile Manifesto and the Agile methods advocate strong collaboration with customers, both internal and external (Beck et al. 2001; Beck 1999; Schwaber 2004); yet, depending on the organisational structure and the history the organisation has with application of Agile, the customers have varying

prior experiences of Agile methods and collaboration with Agile teams. Customers, internal or external, come from different departments and often represent the business needs. Their software development experience can be limited, but their commitment to the collaboration with the development team is invaluable (Martin et al. 2010; Hoda et al. 2011). Cross-departmental and cross-organisational collaboration, collaboration across boundaries, has been studied in the context of more traditional projects (e.g. Levina & Vaast 2005; Gal et al. 2008). I have applied these proven theoretical lenses, boundary spanners and boundary objects, in the Agile development context. I believe that these lenses can expand the understanding of boundary spanning and collaboration. Collaboration between the Agile software development teams and their stakeholders, the other parties involved in the daily work, is one of the key elements that organisations need to address in order to succeed with their Agile initiatives (Rigby et al. 2016). Supporting objects, that aid collaboration, are part of the holistic view on the development work.

These elements of Agile development environments – the underlying Agile perspective, the configuration of stakeholder collaboration and the application of collaboration artefacts – address different aspects of Agile development. In order to form a holistic view of the phenomenon, each aspect needs thorough investigation; thus, the research questions are as follows:

1. **How does collaboration in organisations differ under the various approaches to Agile?**
2. **How do organisations structure collaboration between Agile teams and business stakeholders?**
3. **How are boundary objects used in collaboration between these stakeholders?**

The first question addresses the different perspectives the team members hold towards Agile development, whereas the other two questions form a connected inquiry into the ways in which the organisations need to structure their collaboration between different parties in order to succeed in collaboration and the range of artefacts which are often needed to support the configuration.

The next section will discuss the research focus of my thesis and outline the research questions that have guided me through the design and implementation of my case study research.

1.2. Thesis Structure

This thesis is structured into eight chapters. In this first chapter, the introduction chapter, I have outlined the context of my research, Agile software development, and collaboration between the Agile development teams and their business stakeholder counterparts. In addition, I have briefly discussed the factors that make Agile software development a unique phenomenon and an area of research that lacks theoretical contributions. In order to achieve said contributions, I have incorporated a theoretical lens of boundary objects and discussed why this lens is appropriate. The discussion on Agile development methods and boundary objects is tied together by outlining the focus of my research, a holistic view of Agile development, and the research questions which will lead me to the answer of what a holistic view of Agile development will look like.

The second chapter of this thesis begins by outlining in more detail the origins and popular methods of Agile software development. This first section of the second chapter provides context for the activities described later in the case study chapter. The contextual discussion is followed by a review and analysis of Agile literature. The focus

of this review is to explore the different perspectives towards Agile development methods. The outcome of my literature analysis is that perspective towards Agile is not uniform but rather a spectrum. The authors engaging in discussion on Agile methods see the role of the methods and the Agile Manifesto in different lights and use different terminology when discussing the topic. I will provide examples that fall between the extreme ends of the spectrum and in the middle. This analysis is followed by a discussion on boundary objects, my theoretical lens. I will first outline the definition and origins of the concept in order to familiarise the reader with the theory. The second section expands on what the authors who have applied this theory mean by boundaries, boundary spanning and how boundary objects are defined in the light of these other concepts. In order to describe the applicability of boundary objects as a theoretical lens, the third section outlines the literature that has discussed the utilisation of boundary objects in a variety of contexts, from software development to factory work. This section is followed by a more specific analysis of the literature that has focused specifically on application of boundary objects in the context of Agile software development projects.

The third chapter describes the research methodology and philosophical underpinnings of my study. In this chapter, I describe how I have utilised an interpretive research view and iterative research method, framed by a structured-pragmatic-situational research approach (SPS) (Tan & Pan 2011). I briefly explain why I have based my interpretive research on the works of Walsham (1995; 2006), why I have applied the concept of 'casing' (Ragin 1992), rather than a more positivist way of case definition, and why I have applied the SPS approach when it comes to the data collection, analysis and thesis writing. The methods chapter will then outline the steps I took to conduct my three case studies.

The methodology section is followed by chapters four to six, which describe the case studies. Each case study description focuses on three elements: which Agile perspective the organisation subscribed to; how the technical teams collaborated with the business teams; and how artefacts were used to support the Agile methods. For clarity, the case studies are all formatted similarly. The first section is always an introduction to the case, followed by the description of the organisation(s) involved in the study. The third section details the research methods applied to each case. This is followed by a section that describes the Agile perspective of the organisation(s), then a section discussing the Agile practices and objects applied. Each case study chapter is concluded with a discussion that ties the Agile perspective to the corresponding literature and reviews the ways the business stakeholders and the Agile development teams were engaged and how the boundary objects were applied by the collaborating parties. The discussion section after each case study integrates the cases into the literature on Agile development and boundary object but also facilitates clarity. Each case remains separately analysed in its respective chapter.

In the seventh chapter of this thesis, I present the comparative analysis of the case studies. I discuss the differences between the perspectives held towards Agile, how the organisations had structured their stakeholder collaboration and the different ways the organisation applied boundary objects. The object application is analysed via a theoretical mapping of the three different categories of boundary objects: Infrastructural, Projective and Process objects. This mapping helps to bring abstraction into the different motives the organisations had when applying boundary objects as supporting tools for stakeholder collaboration.

This comparative analysis leads to the final model, the model of Agile engagement. By presenting my three research questions – how the Agile development environment and the process of Agile collaboration is influenced by the organisations' perspective, how the organisations structure their collaboration. and how the teams apply boundary objects – I have come to the conclusion that these three elements of Agile development (the perspective, collaboration configuration and boundary objects) form the underlying structures that impact the Agile development process and the outcomes of the process. In other words, what is done during the development activities is dependent on how the organisations perceive Agile development, how the organisations collaborate across team and organisation boundaries, and how boundary object application supports the collaboration and thus enables the project outcomes.

Finally, the thesis is concluded by a discussion on the theoretical and practical contributions this work has made and what the limitations of this research are, as well as where I plan to take this research in the future.

1.3. Conclusion for the Introduction Chapter

In this introductory chapter, I have outlined the background and motivation to study Agile development through the theoretical lens of boundary objects. The first chapter presented the research questions that guided my inquiry into stakeholder collaboration, boundary object application and different perspectives on Agile development. This introduction has also provided a brief summary of the research approach, introduced each of the three case studies, and outlined the results from the analysis of the cases.

The next chapter provides an overview of Agile software development methods, discusses the Agile literature and the three views on Agile philosophy, and explores the concept of boundary objects, boundaries, boundary spanners and the application of the concept in the context of Agile development.

2. Context and Literature

"Agile is dead. Long live Agility!"

– Dave Thomas (2015)

2.1. Introduction

In this chapter, I will introduce the concept of Agile software development and discuss its philosophical underpinnings. I will critique the literature on Agile software development and the perspective many authors have taken on the subject. I will base this critique on a discussion of the Agile perspective, that is, how the members of the organisations, and organisations as aggregates of their members, view and understand what Agile development is. I see this as a crucial element that is not thoroughly discussed in current literature. I will introduce a selection of Agile methods that the reader will later encounter in the description and discussion of the case studies. I will end this chapter with a literature review on the theoretical lens I will apply in this research – the concept of boundary objects.

Before proceeding further, a few clarifications are required. The language describing Agile development is not without controversy. I have chosen to use 'Agile' with a capital 'A' to signify the Agile practice, whereas 'agile' with lower case 'a' is reserved for the adjective. In place of 'methodology', I will use the term 'method' to describe the best practices of Agile development. I have also deliberately chosen to use 'Agile software development' or 'Agile development' instead of the often used 'Agile project management'. This is an ideological decision, based on the original source – the Manifesto for Agile Software Development, which was written by developers for developers (Beck et al. 2001). The concept of 'management' is not compatible with the values and principles of the Manifesto.

Finally, one more clarification is needed: 'practice' as a term in both academic and practitioner literature can have multiple interpretations, as is explored in the section discussing practice theory. Agile methods are often called Agile practices, but for the sake of clarity of this proposal, I will use the term 'Agile method' when I'm referring to guidelines for work in an Agile environment. The term 'method' will be applied both for the higher-level concept of Agile method as well as the individual Agile methods, such as 'Scrum method', or 'Extreme Programming method'. 'Agile practitioners' are the people who apply Agile methods and 'Agile practice' is what the practitioners do. For example, the members of the organisations can be either Agile practitioners or practitioners of other methods, such as the waterfall method for software development or both, depending on the work. Similarly, 'Scrum meeting' is an Agile practice, described by the Scrum methods.

2.2. What is Agile Software Development?

Agile methods are rooted in the frustration of a group of software development practitioners who critiqued the ways that software development projects were conducted in the corporate world and who had a desire to improve the work and its outcomes (Beck et al. 2001). It is not accidental that the springboard for the Agile movement was named Manifesto for Agile Software Development instead of something less politically charged. Today, in addition to the

original seventeen signees, the Manifesto has been signed by thousands of people who subscribe to the values and principles stated on the website where the Manifesto and the signatures can be viewed.

There are two parts to the development of the Manifesto for Agile Software Development information systems: Agile values and Agile principles. The Manifesto declares the following (Beck et al. 2001, emphasis in the original):

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

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

That is, while there is value in the items on the right, we value the items on the left more.

The second part of the Manifesto consists of twelve principles, which describe in more detail the ideas of the authors. The role of the developers as capable individuals, who should be trusted and not managed, is clearly and repeatedly stated. The twelve principles are summarised in the ***Table 1*** below:

Principles of Agile Software Development (Beck et al. 2001)		
1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.	2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.	5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.	6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.	8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.	9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity – the art of maximizing the amount of work not done – is essential.	11. The best architectures, requirements, and designs emerge from self-organizing teams.	12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Table 1. Principles of Agile Software Development (Beck et al. 2001)

The Agile principles further stress the importance of working software and collaboration between all the project stakeholders. Harnessing change as an advantageous act, rather than seeing it as an annoyance for the work, is further emphasised in the principles. The authors of the principles proclaim that projects should be conducted at a

sustainable pace that the team can maintain indefinitely. This means that overtime and weekend work, activities that projects often revert to when deadlines are pressing, are not acceptable.

The Agile values and principles targeted the software developer community with an emancipatory message. The authors of the Manifesto all have a background in software development and their experiences in the plan-driven environment of the software industry led them to test and design better ways to conduct software development. Highsmith (Beck et al. 2001) writes in regard to Agile practices that "practices define a developer community freed from the baggage of Dilbertesque corporations". He continues:

We want to restore a balance. We embrace modeling, but not in order to file some diagram in a dusty corporate repository. We embrace documentation, but not hundreds of pages of never-maintained and rarely-used tomes. We plan, but recognize the limits of planning in a turbulent environment. Those who would brand proponents of XP or SCRUM or any of the other Agile Methodologies as 'hackers' are ignorant of both the methodologies and the original definition of the term hacker.

Software developers, according to Weber (2004), see their craft not only as a way of solving engineering problems, but also as self-expression. The code they create is a manifestation of aesthetics, style and elegance. Software developers form an occupational subculture that shares norms, values and perspectives (Van Maanen and Barley 1982). The technical knowledge of the individual is highly valued and community members praise the feats of such individuals (Koch 2004), but the community also acknowledges the complexity of software development and thus also values collective efforts when solving larger, more intricate problems (DeMarco & Lister 1987).

For a developer who is influenced by the hacker culture, management and even large corporations that create proprietary software, such as Microsoft, are seen as the enemy (Raymond 2001; Weber 2004). The community sees management – especially micro-management – as detrimental to software development, sometimes even as an insult to the software developer, the artist or the artisan (Weber 2004). The Manifesto and collection of Agile software methods, modelled on the values of the Manifesto, propose a solution to the issue of management. In their world, there are no project managers, but different roles that either facilitate the practices or give direction to the products (Schwaber 2004).

Initial studies of organisations that were early adopters of Agile reported that adopting Agile is a game changer, a significant modification of the daily work, that requires rethinking of the organisational culture (Strode et al. 2012). Agile methods and their use have ramifications which extend beyond development teams (Martini et al. 2016). For example, Ceschi et al. (2005) contrast organisations practicing traditional project management and Agile methods in terms of project success. They conclude that the primary issues with adopting Agile development methods were cultural and involved a reluctance to accept the methods. The need for support from management or other stakeholders is a continuing issue perceived by Agile practitioners (Scrum Alliance 2015); however, Agile methods have become the norm and most organisations have adopted some form of Agile development (VersionOne 2016).

In the next section I will discuss the history of the methods and introduce more thoroughly some of the most widely used Agile methods which were also applied in the case study organisations: Scrum, Kanban and Extreme programming. This will provide context that will aid the reader with understanding the case study descriptions and provide background information that explains the importance of these concepts.

2.3. Popular Agile methods and Key Concepts of Agile Development

The value statements of the Agile Manifesto are not methods or practices; they describe the focus and emphasis the Agile development team should have. The methods are concrete development guidelines that interpret the values and principles of the Agile Manifesto and provide clear instructions on how development work should be conducted. Some of these methods, such as Extreme Programming (Beck 1999) and Scrum (first discussed in 1999 but popularised by Schwaber 2004) Lean development (adapted from Toyota's car manufacturing process (Poppendieck & Poppendieck 2003)) or the Chrystal Family of methods, developed already in 1992 (according to Cockburn 2018) predate the Manifesto. Others, such as Kanban (popularised by Anderson 2010, amongst others), are more recent additions into the Agile fold. In 2002, Abrahamsson et al. collected and evaluated ten different Agile methods. These numbers have grown with the introduction of newer concepts, which are extensions of older methods of Agile development, such as Enterprise Agile (Schwaber 2007) and other frameworks or methods associated with the Agile development, for example Scaled Agile or SaFe (Leffingwell 2016), Design Thinking (Rowe 1991), or Lean Startup (Ries 2011). The practices range from very concrete, such as 'conduct short, daily meetings' (Scrum) to more abstract ones such as 'shared ownership of the work' (Extreme Programming). The values and principles of the Manifesto are the glue that ties all the suggested practices under the same family of methods, but they also give freedom to practitioners to select the most suitable methods that bring value to their projects and organisations.

The flexible premises and the rebellious origins of the Agile practice encourage creation of method concoctions, tailored to the needs of the organisations. Thus, organisations that practice Agile software development are not uniform in their methods or perspectives towards Agile development; however, there are a few key concepts that were applied in the case organisations described in this thesis. I will now briefly introduce these concepts and their origins.

2.3.1. Scrum, Kanban and Extreme Programming

Scrum, Kanban and Extreme Programming are amongst the most popular Agile methods (Scrum Alliance 2015; VersionOne 2017). (Scrum and Kanban are commonly used together as a complementary method, Scrumban (VersionOne 2016)). Scrum is the most well-known Agile practice and was originally created by Jeff Sutherland (Scrum Alliance n.d.) and made popular by Schwaber (2004). The rugby term 'scrum' is borrowed from a paper, "The new new product development game" by Takeuchi and Nonaka (1986), which has been identified as one of the original inspirations for the Agile movement (Beck 1999).

2.3.1.1. The Scrum Method

The Scrum framework focuses on the project management side of software development and addresses the organisational roles and communication practices that ensure collaboration between the development teams and the business stakeholders (Schwaber 2004). The focus on the communication between and responsibilities of the stakeholders such as developers or customers and the lack of programming tips makes Scrum easy to understand but not necessarily easy to implement, as it might require organisations to modify their existing structures and practices (e.g. Marchenko & Abrahamsson 2008).

The Scrum process (***Figure 1***) begins by creating a product backlog that consists of requirements in the form of user stories, that is, what the user would do with a specific piece of functionality. From the backlog, a set of user stories is selected into a sprint, also called iteration. The length of sprints or iterations is defined by the organisation but commonly sprints are from two to four weeks. Tasks that are allocated to a sprint are assigned to a sprint backlog, which defines the tasks for that sprint. The Scrum team meets daily, in a Daily Scrum or Stand-up meeting, to discuss finished tasks, upcoming work for that day and issues the team might face during the day. After each sprint, the team should have a product that could be potentially delivered to end users (Schwaber 2004; Scrum Alliance 2017).

Figure 1. The Scrum Process as Presented by Scrum Alliance (Scrum Alliance 2017)

2.3.1.2. Kanban Technique

The Scrum method focuses on the roles and responsibilities but lacks guidelines on visual representation techniques. Thus, organisations have supplemented the Scrum method with other methods that present tasks visually, especially with a method called Kanban (Anderson 2010; VersionOne 2016). Kanban is a workstream management technique that focuses on timely and non-wasteful delivery. It has its roots in the visual representations of product status as conventionally indicated in Japanese product manufacturing. Each project task is written on a card (physical or virtual) and allocated into a Kanban board, which contains swim lanes that describe the statuses of the different tasks. A simple board could have the following swim lanes: ‘not allocated’, ‘in progress’ or ‘done’. The tasks move from lane to lane according to their status. Figure 2 illustrates an example Kanban task board.

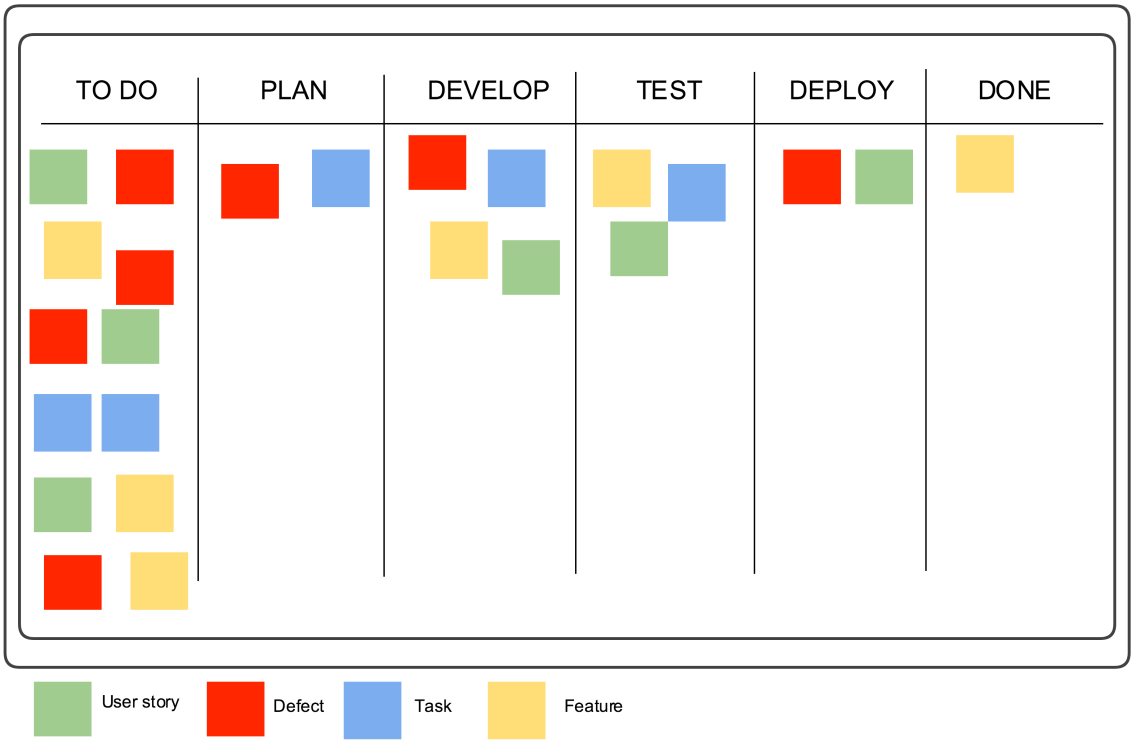

Figure 2. An Example of a Kanban Board

2.3.1.3. Product Owner or Product Managers

To enhance customer engagement, Scrum uses the role of Product Owner, who is a customer representative working with the development team on daily basis. The Product Owner is responsible for the return on investment of the project and for prioritising project tasks and deliverables, and is able to readjust the priorities as needed (Schwaber 2004; Beck 2000).

According to McHugh et al. (2012), the role of Product Owner is challenging for many reasons: there can be tension between the development team and the Product Owner who might not feel that they are part of the team; the Product Owner may have to juggle multiple loyalties to the development team and the business; the communication between the Product Owner and the team should be built on mutual trust; and the relationship needs to be nurtured and maintained throughout the engagement (McHugh et al. 2012).

2.3.1.4. Self-organising team

In addition to sprints or iterations and Scrum, Takeuchi and Nonaka (1986) discuss another key concept of Agile software development, the self-organising team. They describe three conditions for the self-organising team: 1. autonomy, meaning that management does not intervene in the work and the team sets its own directions; 2. self-transcendence, meaning that the team pushes its goals further and further and breaks the status quo if needed; and 3. cross-fertilisation, the team consists of members from different specialisations, thought processes, with a wide variety of personalities encouraging and balancing each other (Takeuchi & Nonaka 1986).

Agile teams are self-organising (Maruping et al. 2009), sometimes also called autonomous workgroups, self-managing teams, self-regulating teams or empowered teams. The team members are given authority and responsibility over many aspects of their work and they can make decisions which will have economic consequences (Guzzo & Dickson 1996). There is greater emphasis on control inside the group than from outside the organisation and the team members can plan and schedule their own work and assign tasks to each other (Manz & Sims 1987).

The concept of self-organising teams has been introduced as an outcome from a socio-technical systems theory (Cummings 1978). Studies show that empowered teams are more productive and more proactive (Kirkman & Rosen 1999) and they are used to improve productivity and quality, and/or to reduce conflict (Manz & Sims 1987). There is also research evidence that lack of autonomy has a negative influence on the performance of new product development teams, and an empirically found negative correlation between lack of autonomy and team performance (Gerwin & Moffat 1997).

2.3.1.5. Scrum Master

Scrum Master, or Coach as the role is called in XP, is a role for someone who is partially a team member but also has independent perspective. The Scrum Master or Coach should not be immersed in the day-to-day work of the team but should retain distance, so that they can notice bottlenecks in communication and encourage the team to work independently (Beck 2000). Scrum practice defines that the Scrum Master is used to shield the development team from distractions and to make sure that the Scrum process is understood and followed. This role is usually more technical and has less people management responsibilities than a traditional project manager (Sutherland 2008; Drurya et al. 2012).

Scrum Master is not a project manager and there is no completely equivalent role for project manager in Scrum. DeMarco and Lister (2013) offer a distinction between a project manager and Scrum Master: in their view, the project manager's role is not to *make* the teamwork but to *enable* them to work without interruptions from outside. The Scrum Master is the enabler, whereas the traditional project manager role has been one of an authoritative overseer. In Scrum, the three Scrum roles – the Team, Scrum Master and Product Owner – all share parts of the responsibilities of the old project manager role. Schwaber (2004, pp. 7) describes, "... in Scrum, the world is divided into pigs and chickens. Pigs are those who are committed to the project, those who have 'skin in the game.' Chickens are the spectators". This analogy is used to explain the three roles of Scrum: all three are 'pig' roles – everyone with a managerial role should share the responsibility and commit to the project. This is a significant change, since the traditional project manager role is separated from the team, watching the team from the outside. In addition, the team, now also part of the management as the self-managing entity, cannot transfer responsibility to outside management but has to be equally committed, along with the Scrum Master and the Product Owner. The application of Agile methods reduces the hierarchy of the project organisation by removing the project manager from the pedestal and distributing the responsibilities amongst the team members (Sutherland et al. 2008).

The Agile method is challenging all traditional project management literature, where the project manager is managing the scope, costs and schedule, since now there is no predefined scope and the costs and schedule are defined by the customer, directly or via Product Owners, prioritising and choosing the next deliverable.

This does not mean that the there is no oversight at all, but that management, distributed amongst different team members, is shifted from assigning tasks and goals to two primary roles: protector and coach (Boehm & Turner 2005).

2.3.1.6. Extreme Programming and Pair programming

Extreme Programming (aka eXtreme Programming or XP) is one of the earliest and most highly influential Agile methods. Unlike the other Agile methods, which focus more on the managerial side and roles of Agile development (for example Scrum (Schwaber 2004)), Extreme Programming focuses on the programming activities. The methods are directed at the developers, in order to enhance their work (Beck 1999; 2001). These best practices (***Table 2***) follow and overlap with the twelve principles of the Agile Manifesto (Beck et al. 2001).

The Extreme Programming practices (Beck 2001)		
The Planning Game—Quickly determine the scope of the next release by combining business priorities and technical estimates. As reality overtakes the plan, update the plan.	Small releases—Put a simple system into production quickly, then release new versions on a very short cycle.	Metaphor—Guide all development with a simple shared story of how the whole system works.
Simple design—The system should be designed as simply as possible at any given moment. Extra complexity is removed as soon as it is discovered.	Testing—Programmers continually write unit tests, which must run flawlessly for development to continue. Customers write tests demonstrating that features are finished.	Refactoring—Programmers restructure the system without changing its behavior to remove duplication, improve communication, simplify, or add flexibility.
Pair programming—All production code is written with two people at one machine.	Collective ownership—Anyone can change code anywhere in the system at any time.	Continuous integration—Integrate and build the system many times a day, every time a task is completed.
40-hour week—Work no more than 40 hours a week as a rule. Never work overtime a second week in a row.	. On-site customer—Include a real, live user on the team, available full-time to answer questions.	Coding standards—Programmers write all code in accordance with rules emphasizing communication through code

Table 2. Extreme Programming Practices (Beck 2001)

Unlike Scrum, which has been successfully applied in diverse development environments, from simple and collocated to complex and distributed (Scrum Alliance 2015), Extreme Programming has sparked more debate on the benefits of Agile methods. This scrutiny has been especially directed towards application of the most 'radical' of the methods, pair programming (Dybå et al. 2007; Salge et al. 2016). Pair programming is a method of code development where two programmers work on the same piece of code. The pair shares one computer with two screens and two keyboards. The shared space and shared work is aimed at enabling the effective distribution of knowledge via instant, face-to-face communication between the members of the organisation. Consequently, less

documentation is generally required (Beck 1999). Research has shown that pair programming produces better-quality code and the developers are more satisfied in their work (Balijepally et al. 2009). Although pair programming is widely researched, the interest in the practice has lately been declining (Dingsøyr & Lassenius 2016). The reason for the decline in interest is related to the environment where Agile development is conducted in organisations nowadays. The practice of pair programming is demanding and requires collocation, whereas a significant portion of development is conducted in globally distributed teams (VersionOne 2016).

Pair programming alone does not guarantee enough communication, especially with non-developer project stakeholders who are not paired up. Meetings and Agile walls are required to ensure the distribution of information.

2.3.1.7. Agile walls and meeting practices

Agile walls or Agile task boards are physical walls or virtual reproductions of walls or boards that are used to convey status information of the tasks. On the physical walls, the tasks are written on Post-it notes, which are then organised on the wall according to a set of swim lanes, which represent the state of each task (Cohn 2010). The Kanban wall discussed earlier is one example of Agile walls (example Kanban wall in ***Figure 2***). The virtual walls are created with tools specifically designed to imitate the appearance of the physical wall, with small virtual cards placed on swim lanes (Cockburn 2006; Sharp et al. 2009).

Several Agile practices are commonly linked with Agile walls. Daily Stand-ups or daily Scrum meetings (Schwaber 2004), as discussed earlier, are short meetings conducted every day. The development team discusses the work done yesterday, the upcoming work and issues that they might have (Agile Alliance 2016). During these meetings, the tasks on the wall are discussed and moved along the swim lanes according to their current status. The planning meeting, where the tasks for the next iteration are selected, is similarly held with the tasks in sight (Cockburn 2006). Finally, retrospective meetings, which are reflections of the successes and issues of the past iteration (Derby et al. 2006), might refer back to the task boards.

2.3.2. Continuous Delivery, Minimum Viable Product and DevOps

The practice of continuous delivery, or frequent delivery as it is described in one of the principles of the Manifesto, is a practice where new features, fixes and changes are delivered to the production environment, for example to the end user, frequently (Humble & Farley 2010). Practitioners of continuous delivery are merging the new pieces of code with the existing code base and striving for code that is always ready to be deployed to users (Cockburn 2004). Continuous delivery is tied together with the Crystal methods (Cockburn 2004) and Lean practices of automation of workflows, minimising system duplication and improving the development process (Ries 2011). The aim is to deliver better quality products with lower costs, faster to market (Humble & Farley 2010).

Minimum Viable Product or MVP is a term that describes a version of a software product that is "that version of a new product which allows a team to collect the maximum amount of validated learning about customers with the least effort" (Ries 2011). The MVP is the first version of the system that could be delivered to end users to test whether a business concept is actually viabled. It has become a tool for start-ups and more established businesses to quickly get their ideas to market (Ries 2011). Continuous improvements that are applied to the MVPs will

eventually transform them into fully-fledged products that continue to be developed until the product itself becomes obsolete.

One of the more recent organisational changes that came about with the advent of Agile methods is the creation of DevOps teams. DevOps is a combination word that is derived from software **dev**elopment and information technology **op**erations. Traditionally, the technology operations team is the team that oversees the hardware side of the information systems, whereas the developers are focused on the software development. By combining the responsibilities of these two teams, a DevOps team is trying to break down the silos between these two parties (Huttermann 2012). Continuous delivery practices and the advent of cloud computing, Software-as-a-Service and Platform-as-a-Service reduce the importance of traditional hardware infrastructure, but emphasise the importance of people who understand the modern infrastructure models and are able to assist software developers (Smith 2011; Huttermann 2012).

2.3.3. Summary: Agile Values and Methods

The previous sections have discussed a variety of practices which all subscribe to the four Agile values and twelve Agile principles of the Manifesto for Agile Software Development. The ***Table 3*** below links example practices and their implementation guidelines to the four values of the Agile Manifesto.

Scrum and Extreme Programming are the two most popular methods applied by Agile practitioners (VersionOne 2016) and also the two most researched Agile practices (Hummel 2014). These two methods are often complemented by other methods, such as the principles of the Crystal Family (Cockburn 2004), Lean (Poppendieck & Poppendieck 2003) or Kanban (Anderson 2010), to create a holistic development framework.

Agile methods are grounded in the values and principles of the Agile Manifesto. An understanding of the Agile origins provides the foundations for deeper analysis on the phenomenon. The next section will discuss the three different viewpoints that one can identify in the literature.

Value	Example Practices	Implementation Guidelines
1. Individuals and interactions over processes and tools	Scrum meetings (Schwaber 2004) including: Daily or weekly meetings Product demonstrations to the extended team (Highsmith 2002)	Meetings should preferably be face-to-face (Beck 1999). Members of the extended team (i.e. stakeholders who do not necessarily participate in the day-to-day activities) should be involved as they may have valuable information or inputs regarding the project (Schwaber 2004). Osmotic communication between team members (Cockburn 2004).
2. Working software over comprehensive documentation	Creation of prototypes to simulate the end products (Highsmith 2002; Poppendieck & Poppendieck 2003; Cohn 2010) Frequent delivery of products for assessment and feedback (Cockburn 2004)	Rapid prototyping can ensure early and constant feedback before further development (Schwaber 2004). Prototype development should be performed simultaneously with systems development (Highsmith 2002), in order to accommodate changes even in the late stages of development. Products are to be released frequently and often for end user feedback (Cockburn 2004).
3. Customer collaboration over contract negotiation	Engaging customers in requirements gathering and prioritisation process (Highsmith 2002). Creating stories describing requirements in lay terms (Paetsch et al. 2003; Schwaber 2004; Cohn 2004).	Specifying requirements as stories enhances communication between the development team and the customers unfamiliar with technical jargons and conventions (Beck 1999). Interpretation of the stories should be verified with the customers to ensure common understanding between all parties (Schwaber 2004).
4. Responding to change over following a plan	Embracing change as a source of potential improvement (Beck,1999).	Maintain receptivity to change, which stimulates creativity that can facilitate the quicker development of the systems that are valuable for customers (Boehm 2002).

Table 3. Summary: Agile Values and Practices

2.4. Agile Software Development in the Academic Literature

The strong practitioner focus of the early writings on Agile and the 'from developers to developers' ideology behind the Manifesto can pose a challenge for a reviewer of the Agile literature. The literature discussing Agile development is fragmented and multi-disciplinary (Dingsøyr et al. 2012) and there is an abundance of practitioner-oriented guides that detail practical application of the multitude of Agile methods (e.g. Agile series).

However, in order to address my first research question, on how different views on Agile development might influence collaboration, I decided to review a selection of both practitioner-oriented and academic papers that could provide an overview of the theoretical views of Agile development methods and offer more insights on how Agile development methods are described in literature. A more thorough analysis on how the papers see Agile and what is

perceived as ‘proper’ application of Agile methods was relevant to my interests, since there was a distinct lack of theoretical understanding of Agile methods and unifying theory (Conboy 2009; Dingsøyr et al. 2012; Hummel 2014) and I could not apply an existing framework in my later data analysis on the organisational Agile perspectives. Nevertheless, there were three extensive literature reviews by Dybå and Dingsøyr (2008), Dingsøyr et al. (2012) and Hummel (2014), which summarised the current body of knowledge on Agile development and guided me with my review and analysis. These literature reviews have distilled the most common research methods, most discussed topics and theories applied, and provide statistics on the state of the research. Based on these reviews, I identified the key papers discussing each theoretical perspective and provided references to historical context as well.

By applying a hermeneutical approach, as described by Boell and Cecez-Kecmanovic (2014), I first summarised and categorised papers according to their topics; then in the second iteration, I delved deeper into the descriptions of Agile. My goal was to analyse if there were different perspectives towards the Agile Manifesto and whether the authors were critical of the concept or the application of the methods, or if ‘agility’ of the projects was taken at a face value.

Based on my review, an Agile perspective continuum began to emerge. The early guidebooks describing the popular methods, such as Scrum (Schwaber 2004), Extreme Programming (Beck 2000) or the Agile Ecosystem (Highsmith 2002), provide a window into the practical application and into the worldview of the prominent authors of the Agile Manifesto movement. These guidebooks and early papers, published in ‘Computer’ (e.g. Beck 1999; Cockburn & Highsmith 2001; Highsmith & Cockburn 2001; Boehm 2002; Cohn & Ford 2003), sparked the discussion on Agile in the development community and fostered the early Agile adoption in organisations. Agile implementation guidelines and explanatory pieces describe the intentions and motivation of the proponents of Agile development. These works provide a starting point for analysis of literature on the Agile development methods as a developer-driven organisational change movement or philosophy, rather than a project management methodology.

These early papers, describing what Agile was and how it should be applied, were aimed at the general public, developers and managers alike. Few papers included short examples of successful Agile implementation (Beck 1999); others relied on explaining the underlying philosophy of the values (Highsmith & Cockburn 2001). With a strong commitment to the newly minted Agile cause, these papers had an agenda and a mission to evangelise Agile into all organisations where software was being developed.

Once the methods became more widely applied, case studies of organisations utilising Agile began to emerge. A stream of research papers and case studies focused on the complexity of human interactions of the Agile teams in both collocated and globally distributed projects (e.g. Holström et al. 2006; Sarker & Sarker 2009). A selection of papers focused on virtual or distributed teams, and their application and occasional struggle with the Agile methods. Another set of papers investigated application of certain types of Agile approaches, for example Scrum or Extreme Programming, often backing their research with case data describing successful Agile projects (e.g. Sutherland et al. 2008). In their conclusions and recommendations, these papers often address the businesses applying Agile methods, and detail what practical implications their papers have that could benefit the teams as well as their managers. Finally, in recent years, the Agile research has also focused on the business benefits of applying Agile methods, based on case studies in large organisations that have applied large-scale Agile projects (e.g. Martini et al. 2016).

The conclusions of such research papers support the narrative of the benefits of Agile, even if the application of Agile is not fully faithful to the instructions of the original Agile guidebooks.

My review of the literature across the body of knowledge of Agile development, presents a spectrum of views and perspectives on the values and principles of the Manifesto, the methods themselves and their application. This spectrum of views ranges from very strict adherence to the Manifesto to advocacy of Agile methods as a flexible business tool. Authors and the organisations they have studied fall into a continuums of different views: the Manifesto and the guideline of Agile methods as either the non-negotiable baseline, as a loose set of ideas that can help organisations to succeed or as a set of best practices that can help the organisations to become more efficient and effective. The following sections will review a selection of papers that present this spectrum of perspectives in literature in more detail.

2.4.1. Agile Manifesto as Philosophy

The rapid growth of the popularity of the Agile methods and large organisations adopting Agile development (VersionOne 2016) has not been welcomed by everyone. The members of the Agile community, even the originators of the Agile Manifesto, have raised concerns about the ideology being thwarted by the 'corporate versions' of Agile – Agile courses and certificates provided by consultancies – which have become a large business. There is a movement that advocates reclamation of the Manifesto and claims "Agile is dead!" (Hunt 2015; Thomas 2014). This backlash against the 'corporate Agile' is an indicator of a more strident and purist camp of the Agile practice, that aligns its ideas of Agile strictly with the Manifesto.

Unsurprisingly, the original signees of the Manifesto have been the most vocal advocates of application of Agile values and principles in their original form. Agile methods have a clear emphasis on the social aspect of the software development, but also philosophical and ethical underpinnings. For example, Beck (1999, pp. 71) has named one of his methods as 'metaphor' and describes it as "the shape of the system is defined by a metaphor or set of metaphors shared between the customer and programmers". In the same paper, he writes (citing Lakoff & Johnson (1999) and Coyne (1996, pp. 73)): "XP's use of metaphors comes from George Lakoff and Mark Johnson's books, the latest of which is Philosophy in the Flesh. It also comes from Richard Coyne, who links metaphor with software development from the perspective of postmodern philosophy". In a similar vein, one of the original authors of the Manifesto, Robert 'Uncle Bob' Martin (personal communication, 7.12.2016), described Agile to be about the ethics and the principles and the craftsmanship. I enquired from Martin what was his opinion on the 'corporate Agile' but unlike Thomas, he was more optimistic about the survival of the ideas and the ethical aspects of Agile, even if the terminology would change in the future.

Ethics and philosophy are also addressed by Beck (1999), who writes that one of the influential works for Extreme Programming was a book named "Peopleware", by DeMarco and Lister (2013, first edition 1987), which drew attention to the human aspects of project development and the sociological rationale of issues in projects. Highsmith (Beck et al. 1999) describes in the History page of the Agile Manifesto:

At the core, I believe Agile Methodologists are really about 'mushy' stuff... about delivering good products to customers by operating in an environment that does more than talk about 'people as our most important asset' but

actually 'acts' as if people were the most important, and lose the word 'asset'. So in the final analysis, the meteoric rise of interest in and sometimes tremendous criticism of Agile Methodologies is about the mushy stuff of values and culture.

Highsmith and Cockburn (2001) and Cockburn and Highsmith (2001) describe the two sides of the Agile development, what they call the people factor and the innovation factor. In their two papers, Highsmith and Cockburn unpack the values and disciplines of the Manifesto from both a people and a business perspective. They reiterate: "Agile organizations and agile managers understand that demanding certainty in the face of uncertainty is dysfunctional. Agile companies practice leadership-collaboration rather than command-control management" and that "they understand that agility depends on trusting the individuals to apply their competency in effective ways". These statements reflect the origins of the Manifesto. Extensive planning has never proven to be compatible with highly elaborate projects such as software projects. Software developers are highly skilled individuals who need leaders, not managers. The onus of the project is on the developers, whom the managers should trust to do their work. Innovation is almost seen as a byproduct of the rapid development cycle.

When it comes to perspective towards Agile methods and the rigour they should be applied with, Highsmith and Cockburn (2001) do not compromise. To them, Agile methods are the baseline that should not be cherrypicked but built upon:

Agile methods offer generative rules – a minimum set of things you must do under all situations to generate appropriate practices for special situations…a team that follows generative rules depends on individuals and their creativity to find ways to solve problems as they arise. Creativity, not voluminous written rules, is the only way to manage complex software development problems and diverse situations.

Similarly, a holistic approach is recommended by Cohn and Ford (2003), who add that adoption of Agile methods is a transition that requires the attention of the whole organisation. Partial adoption, siloed use or resistance from individual employees can be detrimental to a successful adoption. They point out the fact that adoption of Agile might make the developers think that there is *more management* rather than less, due to the communication practices. Patience, collaboration and clarity are their key suggestions for successful transition. For organisations considering application of Agile, Cockburn (2001) has advised, "Consider 'agile' as an attitude, not a formula. In that frame of mind, look at your current project and ask, 'How can we, in this situation, work in an agile way?'"

Even though Agile methods were first implemented by the early adopter practitioners, it did not take long for researchers to embrace the Agile concepts too. Abrahamsson et al. (2002) summarised the Manifesto and early papers (e.g. Highsmith & Cockburn (2001) and Boehm (2002)) into their definition of Agile development methods. According to them, a development method is Agile when "software development is **incremental** (small software releases, with rapid cycles), **cooperative** (customer and developer working constantly together with close communication), **straightforward** (the method itself is easy to learn and to modify, well documented) and **adaptive** (able to make last moment changes)" (emphasis in original).

In a later paper, Abrahamsson et al. (2003) refer to Agile methods with this definition, but also allude to 'philosophical metalevels', which are unfortunately not further detailed in their paper. Instead, a more critical perspective towards Agile emerges; the paper analyses which traditional project management practices are supported

by the Agile methods of the time. Their analysis concludes with the notion of 'incompleteness of the Agile methods', that none of the methods cover all activities they consider necessary for project, especially neglecting project management. The authors called for method specialisation and clearer definitions of method applicability.

Even though Abrahamsson et al. (2003) rely on the Manifesto as their guidance of what Agile is, their paper implies that a mixture of methods and tailoring might be a reasonable approach to take; however, not everyone agrees with tailoring the methods. A more recent paper that particularly discusses the perils of 'cherrypicking' practices without proper understanding of Agile comes from O'hEocha et al. (2010). The authors warn against 'agile assessments' that are conducted to measure compliance to a method, rather than the context of the organisation and the issues. The paper stresses that the feedback loop is a fundamental concept in Agile, a concept that is often forgotten. The authors also note that in their research cases, only the development team was converted into Agile; the holistic approach encompassing the whole organisation was not the focus of their paper. Only more recently has there been more discussion on implementing Agile across different organisational functions; albeit the proponents of such extensions are academics who have been advocating Agile methods since the beginning (e.g. Sutherland & Takeuchi in Rigby et al. 2016).

Nevertheless, the organisational aspect has been the subject of analysis since the very inception of Agile. The common viewpoint has been the comparison between the more traditional and the Agile methods. Boehm's (2002) comparison between the two opposite ends – plan-driven methods and Agile methods – describes what he calls a 'sweet spot', a perfect balance between traditional and Agile. Boehm sees that these methods are not incompatible and that neither represents a fringe but rather a 'responsible centre' for the development. According to Boehm, Agile methods can be applied to entice people away from tempting fringes, the method-free and control-free 'cowboy coding' environment. On the contrary, other authors think that there are more fundamental differences between plan-driven and Agile methods. For example, Nerur et al. (2005) address the differences between the 'traditionalists' and the 'agilists', declaring that the methods are grounded in different concepts. They see that "a rationalised, engineering-based approach, grounded in principles of hard systems thinking" has been the dominant way. This perspective assumes that problems can be specified and there are always solutions, whereas Agile is the antithesis to this way of thinking. To Nerur et al. (ibid), shifting the organisation from traditional methods to Agile requires attention to management, people, tools and the process, all of which should align with the Agile ideology, in order for the methods to work in an organisation. Furthermore, Nerur and Balijepally (2007) speculate on how organisations could better solve so-called wicked problems with Agile methods, problems which traditional methods fail to address sufficiently. They proclaim that "emerging agile philosophy heralds a new epistemology of software development", compare Agile to Action Learning Theory, and declare that Agile facilitates double loop learning and offers "an expansive metaphor of design and the theory of holographic organization" as theoretical bases for Agile development. Unfortunately, even though the 2007 paper is well cited, the suggestions of the authors did not catch the attention of other researchers to an extent that would have produced a theoretical explanation based on these ideas (Hummel 2014). The follow-up discussion by Nerur et al. (2010) focuses more on the historical underpinnings of the phenomenon, rather than new theory building.

However, few other theoretical approaches have been applied. Vidgen and Wang (2009) have applied Complex Adaptive Systems (CAS) theory in the context of Agile development. They cite that a CAS viewpoint is appropriate

as Highsmith and Cockburn (2001) refer to Agile as complex adaptive systems: "One aspect of agile development is often missed or glossed over: a worldview that organisations are complex, adaptive systems". The authors bring together Agile development and CAS concepts such as the 'edge of chaos' or 'region of emergent complexity'; self-organisation, which is already addressed in one of the Agile principles; and the balanced innovation and knowledge creation and process and product improvements. Unsurprisingly, their two case study organisations have elements of Agile organisations but also elements that are unsupportive of the CAS view. Although Vidgen and Wang are not redefining Agile, they do provide a list of Agile team capabilities and organisational traits that act as Agile enablers and Agile inhibitors. The paper does mention that their list could be used to measure 'true agility' of teams, which doesn't quite correspond with the idea of Agile as values or attitude. On the other hand, the authors write that the capabilities are emergent properties of the teams. To me this statement hints that the authors see Agile as an innate process and thus placed this paper as an advocate of Avid Agile.

Theory creation has been a subject also for Conboy (2009), who provides one of the more often cited definitions of Agile (pp. 338) "the continual readiness of an ISD method to rapidly or inherently, create change, proactively or reactively embrace change, and learn from change, through its internal components and relationships with its environment". Conboy bemoans the lack of clarity, theoretical glue, cumulative tradition and parsimony when it comes to Agile research. This often-cited definition is formatted into a taxonomy of Agile development practices (pp. 341):

1. To be agile, an IS development method component (An ISD method component refers to any distinct part of an ISD method) must contribute to one or more of the following: (a) creation of change, (b) proaction in advance of change, (c) reaction to change, and (d) learning from change

2. To be agile, an IS development method component must contribute to one or more of the following (and must not perform poorly in any one of them): (a) perceived economy, (b) perceived quality, and (c) perceived simplicity

3. To be agile, an IS development method component must be continually ready (i.e., requiring minimal time and cost to prepare the components for use)

Another example of a paper that clearly identifies its position on Agile and, in addition, elaborates on the Agile practices of its case study organisations comes from Strode et al. (2012), who discuss coordination of Agile teams in collocated premises. They begin by acknowledging the revolutionary nature of the Agile methods and proclaim that Agile development is a new paradigm in information systems development. They discuss the ramifications of Agile adoption; for example, changes in the organisation culture, roles, customer involvement and team interactions. They also discuss the Manifesto as a unifying philosophy that ties together Agile methods.

Introduction of new methods such as Kanban or Lean has driven researchers to re-examine the definitions of Agile. Wang et al. (2012) have analysed Lean (Poppendieck & Poppendieck 2003) against the values of Agile. They state that some see no difference between Lean and Agile but others claim that Lean is a strategic way of thinking rather than a tactical one. Lean principles, which recommend that customer value should dictate all parts of the development, echoes the Agile principles of the Manifesto. In their analysis, the authors claim that Lean approaches can be especially helpful when collaborating between different stakeholders from the business and from the development organisations. Wang et al. cite Poppendieck and write: "lean concepts and principles have often been

used as thinking or sense making tools to guide the practice of agile software development" (pp. 1297). They conclude the paper by reflecting on Agile and Lean as mindsets and encourage further examination of these two concepts on a more detailed and operational level. The authors of the "Leagile" paper see both Agile and Lean as compatible ways of thinking, similar and intertwined. The mindset aspect, understanding Agile as a philosophy with the willingness to allow compromises, makes this paper an excellent bridge between different schools of thought and perspectives towards Agile.

Topic	Agile Is…	Authors
Agile is about the self-organising team, change and quick responses in a changing business environment	Worldview of organisations as complex adaptive system, a set of generative rules and an ecosystem, celebrating the craftsmanship of individuals	**Highsmith and Cockburn (2001) and Cockburn and Highsmith (2001)**
Introduction to different Agile methods and literature review of origins of Agile Comparison to plan-driven methods and between Agile methods	Agile development is a risk avoiding, incremental, cooperative, straightforward and adaptive. Agile development has philosophical metalevels and complex origins; some branches of the origins are discussed by Abrahamsson et al. (2003)	**Boehm (2002); Abrahamsson (2002) et al. and Abrahamsson et al. (2003)**
What is required from the organisation that wishes to utilise Agile methods and challenges organisations might have when adopting Agile methods	Agile is a group effort. Whole organisation is required to partake in the transition; contrarian actions of individuals or stakeholder groups, especially in management, can sabotage the use of the methods. An agile philosophy, a concept competing with the traditional values, social inquiry in which extensive collaboration and communication provide the basis for collective action	**Cohn and Ford (2003), Nerur et al. (2005)**
Examination of the conceptual foundation and philosophical shift	An expansive metaphor of design and the theory of holographic organisation, an emergent paradigm, leaning towards the paradigm of 'social relativism' (Hirschheim and Klein 1989)	**Nerur and Balijepally (2007), Nerur et al. (2010)**
Definition of Agile software development, often cited and influential	The continual readiness to rapidly or inherently create change, proactively or reactively embrace change, and learn from change	**Conboy (2009)**
Enablers and inhibitors of Agile methods, application of Lean practices in Agile project, application of coordination practices and discussion on how the methods should be applied	Replacing detailed upfront plans, precise predictions and rigid control strategies with Agile, used to "to bound, direct, nudge, or confine, but not to control". It is a mindset, a new paradigm, designed to deal with change and uncertainty. Agility is founded on creativity, pro-action, reaction, learning, cost, quality and simplicity.	**Vidgen and Wang (2009), O'hEocha et al. (2010), Wang et al. (2012), Strode et al. (2012)**

Table 4. Agile as a Philosophy

Table 4 collects the example papers discussed throughout this section and summarises the topics of the papers and their view on Agile development. In the next section, I will discuss examples of research where the empirical cases and their implications to practice take precedence over philosophical discussions.

2.4.2. Application of Agile in Constrained Environment

Whether Agile methods are seen as a philosophical paradigm shift (Nerur et al. 2005) or as an example of complex adaptive systems (Vidgen & Wang 2009), the intention of the authors of the Agile Manifesto was to create a better way of working and better software development project outcomes (Cockburn & Highsmith 2002). The application of the methods in the 'real world' and the outcomes of said projects are as relevant as the paradigm discussions. The short history of Agile method application shows that the methods were not only an academic experiment, but rather a significant driving force of organisational change (Dybå & Dingsøyr 2008; Hummel 2014; VersionOne 2016). The widespread adoption of Agile methods across industries and geographical regions has been motivated by the observable success of the organisations who have applied Agile (Scrum Alliance 2015). But as with any idea which begins to spread across the world, different perspectives begin to emerge.

Where some organisations and authors hold to the ideal Agile application and emphasise the philosophical nature of Agile (for example Strode et al. 2012), other organisations and authors see Agile development methods as a means to an end, as a way to enable efficient collaboration and communication rather than a strict rules that has to be observed. Literature that discusses case studies based on empirical findings often indicates that the Agile values and principles are not forgotten in the 'real world', but that when it comes to implementation of Agile methods, compromises are acceptable. The ideas of the Manifesto are still accounted for, but rather than transforming the organisations to fit the method, the methods are selected and applied however the organisations deem beneficial, in order to achieve better communication and collaboration. Agile methods are seen as facilitators of collaboration rather than a philosophy or mindset.

Such a perspective is especially prevalent in papers that discuss partially or fully virtually conducted and globally distributed software development (Sarker & Sarker 2009; Modi et al. 2013) or larger-scale projects (Sutherland et al. 2008). While these works detail the benefits of the Agile methodology, the focus is not on how best to implement the methods, but on how Agile can be used to make the development better in various circumstances. The practical constraints of the environment take precedence over the Agile principles. By embracing this approach, the proponents of Agile methods have been able to overcome the original stigma of Agile: a view that Agile is only applicable for simple projects and smaller teams (Dybå & Dingsøyr 2009). For example, Manifesto signatory Jeff Sutherland describes how Scrum can be implemented in large organisations. In this description, Sutherland does not explore the values or the principles but matter-of-factly lists methods that work in the large-scale environment. He advocates applying the Scrum method in distributed teams as well as large teams, by scaling the method and combining it with the CMMI (Capability Maturity Model Integrated) model (Sutherland et al. 2008). Researchers have shown that the approach advocated by Sutherland, that is, application of Scrum in large-scale, distributed projects, can been successfully applied with positive effects (Paasivaara et al. 2008). However, later work of Paasivaara et al. (2012) suggests that some organisations struggle with the application of the methods and fail to reap all the benefits that other organisations have achieved with their application of Agile. These two studies imply

that Agile development methods have been applied outside the originally intended scope and must be tailored to fit the context of globally distributed development. The authors proceed to discuss Agile methods through the application of the suggested activities, rather than as a philosophy.

Similarly, Holmström et al. (2006) only briefly refer back to the values of Manifesto, but swiftly proceed to the application of methods to bridge different elements of distance in globally distributed teams. They conclude that even a very rudimentary use of few Agile methods helps to reduce distances between team members and they advocate further research into the application of the methods in such teams. Korkala and Abrahamsson (2007) present another perspective on a similar topic – communication in an Agile, distributed project – with even less emphasis on the Agile aspects of the project. The authors do not question if their case study could be considered Agile by other Agile practitioners, nor specify what they mean by Agile. In this paper, Agile is seen as a collection of methods: a way of doing, not a way of thinking.

In addition to global software development and control issues, the more flexible perspective of framing Agile as a matter-of-fact way of working is prevalent in case studies involving the customers and their role in projects. Where the virtual and distributed nature of projects present geographical and temporal constraints, the customer involvement can present a different set of constraints. Customer involvement, or the lack of it, poses challenges for the Agile development teams (Hoda et al. 2011). These constraints are discussed from different perspectives in papers ranging from ideal customers (Martin et al. 2010) to customer involvement (Kautz 2009) and the role of onsite customers (Koskela and Abrahamsson 2004). A common thread across these papers is that sufficient customer involvements can be achieved, but not necessarily without some compromises in Agile method application. When discussing customer involvement, the authors source definitions of Agile methods from academic literature, such as Conboy (2009) or the Manifesto (Beck et al. 2001), and the guidebooks of the Agile methods, such as Scrum (Schwaber 2004) or Extreme Programming (Beck 2000; 2004).

The authors of literature discussing distributed virtual collaboration and customer involvement acknowledge to the readers that they are familiar with the origins of the methods, but refrain from critique of the modifications applied by the case study organisations and the perceived level of ‘agility’ of the organisations. The application of Agile methods in the case organisations studied in these papers is not contested. By adopting this ‘liberal’ perspective on Agile methods, the researchers who have conducted these studies can aid other organisations in their journey of Agile adoption and adaption. The empirical findings have highlighted issues and benefits of Agile application in their case organisations and the results and suggestions are actionable by other organisations who wish to follow the advice.

However, one might think that these papers could have benefitted from a more philosophical perspective and a discussion on whether the members of the organisations themselves subscribed to the values and principles of the Manifesto. The analysis in the papers lacks discussion on whether the results were due to proper understanding of Agile values and principles or despite it. Granted, there have been no clear guidelines on how to evaluate ‘agility’ (O’hEocha et al. (2010)) and single case studies lack comparison aspects that would reveal differences in the perspectives. Nevertheless, even though an Agile purist might contest the ‘agility’ of the research premises, the flexible approach towards application of Agile methods does extend the reach of Agile beyond the ‘ideal’, small and

collocated organisations. If one believes in the premises of the Agile environment, creation of a better work environment for the developers themselves is already a noble goal (Cockburn & Highsmith 2002) and elitism and exclusivity should be suspended, and adoption of any type of Agile environment should be lauded.

Topic	Agile Is…	Authors
Large scale Scrum development case study discussion and examples of achieving shared understanding and communication	Designed to add energy, focus, clarity and transparency but might pose issues for communication due to volatile environment "At the bottom of these methods lies the assumption that software development is an empirical rather than a defined process and needs a different way of working than described in the waterfall model" (Paasivaara et al. 2008, pp. 87)	**Sutherland et al. (2008), Korkala and Abrahamsson (2007), Paasivaara et al (2008; 2012), Modi et al. (2013)**
Development of agility in globally distributed, offshored and virtual environment for bridging different distances (temporal, geographical and sociocultural) between globally distributed teams with Agile.	Multi-faceted phenomenon, requires resource agility, process agility and linkage agility and an attempt to satisfy the industry quest for faster development processes	**Holmström et al. (2006), Sarker and Sarker (2008)**
Studies of customer involvement, participation and empowerment and reasons for the lack of involvement in the XP and other Agile project	As defined in Manifesto (Beck et at. 2001), XP method guidelines by Beck (2000) or by Conboy (2009)	**Koskela and Abrahamsson (2004), Martin et al. (2010), Kautz (2010), Hoda et al. (2011)**

Table 5. Application of Agile in Constrained Environments

Table 5 summarises the topics and the perspectives towards Agile development in the literature that has investigated the role of Agile methods in various empirical settings. These papers have analysed the benefits organisations can gain from even partial application of Agile methods and how these organisations have managed to apply Agile methods despite geographical, temporal or contractual constraints that are not addressed in the original Agile guidelines. When it comes to the Agile perspective spectrum, the papers discussed in this section fall somewhere in the middle. They are not strictly 'Agile purists' but the authors often address that Agile methods are not only tools but have more depth.

2.4.3. The Business Benefit View of Agile

While the philosophical view of Agile is at one end of the spectrum of Agile views, the opposite end of the spectrum is formed from a perspective that sees Agile methods as a means to provide organisations with the readiness to

change according to market conditions. This perspective does not disregard the perspective of Agile which emphasises the philosophy or craftsmanship and team empowerment, but these factors are not seen as the primary drivers for Agile adoption.

An example of this perspective is provided in a paper discussing the adaptation of Agile by Cao et al. (2009), who assert: "we define adaptation as the process of changing agile methods to align them with the needs of different projects and organizational environments. This process involves adding, dropping, or modifying specific practices prescribed by agile methods". While the authors who view Agile methods as a philosophy would rather change the organisation to fit the philosophy, the more business-benefit-oriented authors would rather change the methods; however, Cao et al. do point out that if the adaptation of practices goes against the 'spirit of Agile', the adapted method might be less effective.

Chan and Thong (2009) only briefly mention philosophy, when referencing Nerur et al.'s (2005) paper. They have created a framework of Agile acceptance where, based on their literature review on Agile, they break agility into three factors: ability-related factors, motivation-related factors and opportunity-related factors. According to Chan and Thong's framework, these three factors, along with the characteristics of Agile methods and knowledge management, have effect on Agile acceptance in organisations. By knowledge management they mean the creation, retention and transfer of both tacit and explicit knowledge. In their view, the more the organisation knows about Agile, the better the acceptance. Paradoxically, this perspective manages to simultaneously capture and lose the value-driven essence of Agile; for example, the factors of Agile highlight individual characteristics and motivation, but omit striving for self-management and craftsmanship. They discuss self-efficacy in ability-related factors, but link it to use of 'an agile methodology', rather than accomplishments in the work. Training is seen as an encouraging factor for acceptance, but the authors also seem to believe that there should be pre-existing factors, such as good communication or shared understanding with the customers, in order for Agile methods to be appropriate. This seems to me to be a backwards approach: trying to fit Agile into organisations that are already fairly Agile, rather than educating organisations who struggle, to become better at Agile.

Lee and Xia (2010) approach Agile development from the perspective of 'software development agility': effective and efficient response towards change. They combine concepts of project team autonomy and diversity with response effectiveness and response extensiveness in an attempt to distil the building blocks of an Agile team. They focus heavily on the project outcomes, rather than the Agile journey, and see Agile development as a way to improve the change response. Software development teams are encouraged to prioritise between costs, schedule and scope; a prioritisation that will define the need for each agility dimension introduced in the paper, which should in turn define how much autonomy and diversity the teams require. To an author from the Agile-as-a-philosophy camp, this idea might be unsavoury, as it implies that the teams should conduct their work in the exact opposite way of what was described in the Manifesto. The Manifesto declares that the team should always be autonomous and diverse when it comes to technical competences.

Similarly, a research by Goh et al. (2013) discusses how trust and control can be balanced in a large-scale Agile implementation, but the authors seem to misunderstand the main Agile principles. Goh et al. are not satisfied by current empirical studies, which adopt the practitioner-driven Agile practices as their foundations, since these

practices are interpreted differently. Instead, they rely on the Conboy (2009) definition of Agile: the taxonomy discussed in a previous section. Goh et al. emphasise control over collaboration and mention rigorous change management processes that make the Agility of their case project questionable. The paper cites the Agile taxonomy coined by Conboy (2009) and discusses internal control mechanisms which are relevant to self-organising teams (Manz & Sims 1987); yet it manages to bypass the values of the Agile Manifesto. By applying only Conboy's taxonomy and definition and ignoring the values and the disciplines of the Manifesto, one can avoid applying practitioner-driven Agile, but I find this approach misguided. I agree with Wang et al. (2012, pp. 436), who point out: "Agile method use is often superficially judged as used or not used whereas the actual implementation can be subtle, partial and inconsistent and so categorising a method as used or not used may be overly simplistic".

The last example of a business-oriented perspective comes from Martini et al. (2016), who have researched boundaries between the Agile teams and other business stakeholders. Their study focuses on software development teams who are applying Agile methods, but who reside embedded inside larger organisations whose core capability is not software but manufacturing. Martini et al.'s paper is oriented around the speed of the development and challenges in interactions between the Agile teams, the larger organisation and the non-software stakeholders.

Topic	Agile Is …	Authors
Model of acceptance and adaptation of Agile methods	Consists of ability-related factors, motivation-related factors and opportunity-related factors, A method collection that provides advantage to business Has a 'spirit' of which adaptations might be consistent and if inconsistent, should call for scrutiny of the organisations	**Chan and Thong (2009), Cao et al. (2009)**
Relationship between response efficiency and response extensiveness and the impact of the team autonomy on both concepts, Evaluate trust and controls in Agile environment. Effects of different controls such as self-controls or outside control	"Software development agility, which is defined in this research as a software team's ability to efficiently and effectively respond to user requirement changes" and "As software development agility is a central concept and a core value of agile development" (Lee and Xia, 2010, pp. 88) Allows adapting to changing environment and provides better collaboration between developers and business.	**Lee and Xia (2010), Goh et al. (2013)**
A comparative study of Agile development in three large Agile companies	Stressing the importance of responsiveness and interactions between involved actors	**Martini et al. (2016)**

Table 6. Business Benefits oriented Agile Literature

A summary of the literature discussed in this section is provided in ***Table 6***. The previous three sections have provided examples of literature that covers the spectrum of perspectives towards Agile development from the

philosophical end to the business end. In the next section, the analysis shifts from the discussion of Agile perspectives to a brief review on the Agile literature, incorporating artefacts as part of the studies.

2.4.4. Stakeholder Collaboration and Artefacts in Agile Literature

The previous three sections detailed the perspectives on Agile development methods, pertinent to the literature. These papers differ not only by the views presented, but also by the distinct topics that are discussed. The philosophy-oriented papers focus on analysing Agile methods from a theoretical perspective. Many of the early papers are guidelines of how Agile should be implemented (Boehm 2002; Highsmith & Cockburn 2001; Cockburn & Highsmith 2001; Cohn & Ford 2003), whereas some of the papers are seeking to define Agile (Nerur & Balijepally 2007; Conboy 2009) and discuss Agile adoption (Vidgen & Wang 2009; O'hEocha et al. 2010; Wang et al. 2012).

The more flexible views of implementation of Agile methods are often seen in papers that address the management of Agile project teams, factors that promote effectiveness to stakeholder inclusion, and the role of the customers. These authors have investigated the human elements of Agile, such as levels of team autonomy (e.g. Sharp & Robinson 2004) and team member diversity (e.g. MacCormack et al. 2001). More recent studies of this category have also looked at the means of enhancing communication within globally distributed teams (e.g. Korkala & Abrahamsson 2007; Ramesh et al. 2006; Modi et al. 2013) and the extension of the concept of development teams to include external stakeholders, such as customers or product owners (e.g. Hoda et al. 2011).

Finally, the papers that focus on business benefits emphasise factors such as team efficiency (e.g. Lee & Xia 2010) and effective collaboration (e.g. Chow & Cao 2008; Martini et al. 2016). These factors are applicable for creating greater business benefits for the organisations; however, amidst the papers that represent these three views, there are knowledge gaps that require more attention. My second research question, the two-part question on collaboration between the Agile teams and other parties and the use of artefacts as part of the collaboration, has not been sufficiently addressed to date.

The first part of the research question, the stakeholder collaboration, has attracted some attention. Researchers have been studying the relationship between the Agile developers and their customers (Hoda et al. 2011; Kautz 2009; Schlauderer & Overhage 2013); however, customers are not the only stakeholders that the development team has to collaborate with. Internal collaboration between the development teams and business stakeholders, such as other organisational functions, is an under-researched area. The study of Martini et al. (2016) (discussed in more detail in section 2.5.4) is one of the rarities that provide accounts of intra-organisational collaboration.

The second part of the research question, the application of boundary objects, is even less studied. Currently, only ten per cent of literature focuses on tangible or virtual artefacts that could be fostered by the human behaviours and aid adoption (Hummel 2014). Based on the literature reviews, I have found a handful of papers that analyse the artefacts commonly used to support the Agile development environment.

There are some common elements to these studies: the papers that address artefacts are currently less theoretical and aimed at providing practitioner-oriented contributions in the form of prescriptions that organisations can apply (e.g. Azizyan et al. 2011; Dowling & McGrath 2015). A perpetual topic for these papers is the concern around clashes

between the use of the tools and the Agile methods. For example, Azizyan et al. (2011), who conducted a survey of Agile tools used by organisations, state that the choice of Agile tools is highly relevant both for managers and developers. They report that common issues that their respondents had with the tools were related to the tools that were not flexible enough to accommodate the process changes of Agile development. These inflexible tools were also seen as inhibitors of creativity in Agile projects. Azizyan et al. conclude that the need for tools was context specific. Different stakeholder configurations required different approaches. In a more recent paper, Dowling and McGrath (2015) list a set of open-source tools, which can be applied in Agile development. Their paper focuses purely on listing the tools and their concerns are targeted towards specific tools and technical details, not the overall Agile development landscape. However, they do assert that there can be a mismatch between tools and the desired view on what Agile development is.

Researchers who investigate how organisations can use objects across the project lifecycle have taken a slightly more theoretical perspective. Kurhmann et al. (2013) propose an artefact model that matches artefacts with different Agile methods and with the stages of the project they are used in. Wagenaar et al. (2015) present a Scrum artefact model, which references the Kurhmann model but creates a Scrum-method-specific version. Interestingly, the findings from the Wagenaar et al. study indicate that their research did not identify artefacts that would store knowledge and carry it across the iterations. Their paper states that knowledge transferability over to other teams might become an issue in Agile software development.

Where Wagenaar et al. point out the issues of knowledge transfer, Ghobadi and Mathiassen (2016) attempt to address the issue by applying causal mapping. They define seven knowledge-sharing barriers: team diversity, team perceptions, team capabilities, project communication, project organisation, project settings and project technology barriers. The authors discovered that the user representatives in their case study were blaming the development teams for lack of understanding of the users. Unfortunately, this paper does not present definitive solutions to these barriers. It does, however, mention that managers have to be careful when implementing solutions, for example Agile meetings or communication tools, as they might have inhibiting impact.

Sharp et al. (2009) take a different perspective toward the artefacts. Instead of mapping the artefacts against Agile methods, they focus on the Agile walls and story cards. Their guiding theory, cognitive dimensions framework, shows that the walls and the story cards are important tools for several reasons: they increase visibility; they encourage consistency and progressive evaluation amongst the developers; and they reveal dependencies. They conclude that the walls and the story cards are tightly linked and transferring them from physical format into digital format must be done with great care. By creating a digital version of the wall, an organisation might lose some of the social context and behaviours, which are linked to physical walls. ***Table 7*** summarises the authors and papers that address artefacts or issues that the artefacts could be used to mitigate.

The importance of the right set of tools or artefacts is the common thread amongst these papers. Organisations should be careful when selecting artefacts, in order to remain innovative, flexible and retain the social elements of Agile development. As discussed in the introduction, a subset of these artefacts can be identified as boundary objects. In one of the few papers that combine Agile development with the concept of boundary objects, Bass (2016) describes a set of artefacts used in a large-scale Agile development, applying the pragmatic perspective of Agile.

The paper recognises the tensions between the documentation and other artefacts mandated by the scale of the projects and the values of the Agile Manifesto, which de-emphasises the use of such artefacts. The concept, boundary objects, provides "a complicated sensing device to register a complicated set of events" (Weick 2007, p. 16), which I applied as a theoretical lens for the case studies when I analysed the collaboration activities and stakeholder engagement. The next section will further expand on the concept of boundary objects.

Topics	Agile Is…	Authors
Use of story cards and Agile walls from the perspective of cognitive dimensions, surveys of which Agile or open source tools are currently used in the industry	Agile is seen as a philosophy or the views on Agile development are not stated	**Sharp et al. (2009), Azizayan et al. (2011), Dowling and McGrath (2015)**
Theoretical model propositions of artefacts in Agile development	Agile can be made to work in constrained environment with some modifications	**Kurhmann et al. (2013), Wagenaar et al. (2015)**
Investigation of knowledge barriers in Agile teams resulting in a framework of seven types of barriers	Case organisations are defined as Agile practitioners and a few practices are mentioned but their Agile practices are not further explored.	**Ghobadi and Mathiassen (2016)**
Artefacts used in tailored large-scale Agile projects	Pragmatic perspective, Agile is a way to avoid project failures, development using short iterations, feature-driven development and the close interaction with customers	**Bass (2016)**

Table 7. Objects in Agile Literature

2.5. Boundary Objects

In this section, I will first discuss the literature on boundary objects. This section is divided into four parts. First, I will define what boundary objects mean and provide examples of literature that has applied the concept as a theoretical lens. In addition, a brief historical overview positions the concept of boundary objects as a derivative of the Actor-Network Theory and mentions similar theoretical concepts. The second part presents examples of literature that has introduced concepts that have been helpful when I have conducted my analysis of boundary objects: definitions of boundaries, boundary spanners and extensions of the concept itself.

In the third part, I will introduce in more detail a selection of papers that have applied the concept and derived different results on how the objects are applied and what forces can impact the application. The different views presented in these papers have provided me with a sounding board with which I can compare my own findings. Fourth, I will review the handful of research papers that I found during my literature search that address the intersection of research of boundary objects with Agile software development environments.

2.5.1. Definition and Origins of Boundary Objects

Boundary objects are artefacts that are used to convey information across groups or organisational boundaries (Star & Griesemer, 1989). According to the originators of the concept, Star and Griesemer (1980, pp. 393), boundary objects are:

…both plastic enough to adapt to local needs and constraints of the several parties employing them, yet robust enough to maintain a common identity across sites. They are weakly structured in common use, and become strongly structured in individual-site use. They may be abstract or concrete. They have different meanings in different social worlds but their structure is common enough to more than one world to make them recognizable, a means of translation. The creation and management of boundary objects is key in developing and maintaining coherence across intersecting social worlds.

Boundary objects allow different groups to work together without consensus and act as 'organic infrastructures' that arise organically from the needs of the collaborating parties, responding to information and work requirements that these parties pose (Star 2010). Features such as adaptability to local needs and simultaneous plasticity and robustness allow boundary objects to play a vital role in enhancing common understanding and collaboration among a diverse set of participants (Vlaar et al. 2008; Barrett & Oborn 2010).

In a later paper, Star (2010) further clarified the intentions of the boundary objects concept. In this paper, she places the emphasis on the actions of the different parties towards the objects. The clarification states that boundary objects are artefacts that are acted 'towards and with'. According to Star, the action of the parties attributes to the objects their materiality, not their actual physical or virtual form. The objects are rather sets of work arrangements, both material and processual, which are then worked on by local groups who are cooperating without consensus (Star 2010, pp. 604). Star proclaims that there is no need for consensus at the beginning of the interaction, nor there is need for consensus at the end of the interactions; the boundary objects support the actions nevertheless.

The literature provides a diverse set of examples of boundary objects. Star and Griesemer (1989) specify four types of objects: repositories, such as libraries or museums; ideal types, such as diagrams and atlases; coincident boundaries, such as maps; and standardised forms that require each party applying the forms to fill in the same information. Other authors have researched boundary objects such as sketches, CAD models and prototypes (Henderson 1991; Subrahmanian et al. 2003; Levina & Vaast 2005; Gal et al. 2008), product engineering or product designs (Carlile 2002; Subrahmanian et al. 2003), other conceptual drawings (Beckhy 2003), timelines and Gantt charts (Yakura 2002; Boell & Hoof 2015), machinery designs (Karsten et al. 2001), and other computer-aided design tools such as requirement specifications, project management tools or intranets (Boujut & Blanco 2003; Levina & Vaast 2005). This list demonstrates that even in the context of organisations and software development projects, boundary objects encompass a wide variety of artefacts.

Boundary objects range from being concrete and tangible, such as paper maps (Star & Griesemer 1989) to representations of abstract concepts, such as time in the form of timelines (Yakura 2002). Star (2010) points out the paradoxical nature of boundary objects: "how can [boundary objects] be simultaneously concrete and abstract? The same and yet different?" (pp. 614). Winkler et al. (2014) address this dilemma by stating that "boundary objects can

simultaneously be abstract and concrete, or robust and plastic because the object itself consists of different dimensions – with one dimension being abstract (e.g. the building prototype) and the other being concrete (e.g. the working prototype)" and that "one boundary object can have polarized properties since one object exhibits those properties at different points in time" (pp. 17)

The multidimensional, abstract and concrete, material and immaterial nature of boundary objects stems from the origins of the concept. Star and Griesemer coined the concept based on research and theories of Law, Callon and Latour (Callon & Latour 1981; Law 1992; Latour 2005) and more specifically, their work on Actor-Network Theory (ANT). Actor-Network Theory was suggested as a method that allows the analysis of the networks that form the world, from a non-human-centric perspective, rather than 'a theory' in the usual sense of the word (Latour 2005). ANT has been influential in the field of information systems (Walsham 1997; 2006) and has influenced other popular information systems concepts, such as sociomateriality (Orlikowski 2007), which has in turn influenced numerous other research papers (Parmiggiani & Mikalsen 2013).

The concept of boundary objects has resonated with researchers interested in solving organisational collaboration challenges. Few theories have been proposed to accompany the concept. The earlier studies that derived their theoretical foundations from Actor-Network Theory (Latour 2005) are followed by studies that apply practice theory (Carlile 2002; Levina & Vaast 2008). The next sections will review in more detail the literature and the theories that discuss the concept of boundaries and boundary spanning.

2.5.2. Defining Boundaries and Extending Boundary Objects

By definition, boundary objects allow different parties to collaborate across boundaries; however, what defines said boundaries is a complex subject. In their case studies, Levina and Vaast (2008) have explored collaboration across boundaries in globally distributed projects, listing the boundaries created by physical distance or institutionalised social boundaries, such as boundaries created by culture, organisations or the functions of the collaborating parties, as well as the boundaries created by the situated practices of the parties.

Situated practice refers to knowledge embedded in the organisation; knowledge that is tacitly shared and ways of working that are familiar amongst the members of the organisations. In Levina and Vaast's study, situated practices are related to the different backgrounds of the developers and their managers, ranging from the practices created by the computer science discipline to the financial services industry. The authors explain that the different boundaries, based on educational background, knowledge of the industry and access to resources, should be taken into account when discussing cross-boundary collaboration.

When analysing the boundaries, Levina and Vaast source their definition of boundaries from practice theory (Bourdieu 1977). They state that fields or practices are separated by the boundaries that arise from differences in the practices across the fields. These fields are dynamic and the boundaries are renegotiable. New 'fields of practice' and new boundaries can arise when new information systems are created or new practices are applied. They state, "the fields and the boundaries produce both sharedness and differences" (pp. 309). The 'sharedness' is required for collaboration, whereas the differences can impede it. New, joint fields of practice emerge from boundary-spanning competencies of the boundary spanners (Levina & Vaast 2005). Levina and Vaast (2005) state that development of

boundary-spanning competency "means producing a specific type or organizational capital by using and relating capital produced in other fields." By 'capital', they mean knowledge, experience or other capital embedded in the field.

Project conflicts and conflict resolution are tied to the boundaries and to the structures that create boundaries. Carlile (1997; 2002) applies Bourdieu's theory as a tool that investigates the connections between individuals and structural forces. Carlile explains that Bourdieu's practice theory omits duality between these forces and asks 'where' the individuals and structures are connected instead of 'what' connects them. The lack of dualism allows the investigation of objects as constraints and structures of knowledge. Knowledge-in-practice can be embedded in objects and can be transferred across organisational boundaries.

Carlile (2002) defines three types of boundaries: syntactic, semantic and pragmatic. In his research of boundary objects, such as drawings and prototypes used in design, production and manufacturing, he notes that the practical role of boundary objects is to establish "a shared syntax or language for individuals to represent their knowledge" (Carlile 2002, pp. 451) and that the object "provides a concrete means for individuals to specify and learn about their differences and dependencies" (Carlile 2002, pp. 452). Syntactic boundaries are boundaries created by different syntax conventions: a lack of shared syntax can inhibit the quality of the communication. Semantic boundaries arise when, even in case of shared syntax, the intended meaning is not carried across and interpretations differ. By pragmatic perspective towards boundaries, Carlile means that the individuals should understand the differences and the consequences of dependencies. When applying boundary objects, the pragmatic boundary is addressed by facilitating a process where the collaborating parties can jointly transform their knowledge and apply changes into the objects, if the collaboration results in negative consequences. According to Carlile, boundary objects also have a political role, which comes from the need to facilitate a transformation of knowledge to create new knowledge. Carlile writes that sharing knowledge across boundaries comes at cost; transforming knowledge is creating new knowledge, altering existing knowledge and validating it. What happens to the objects during this knowledge transformation is not discussed in the 2002 article, but in his earlier work, Carlile (1997) points out that effective boundary objects are accessible and timely, up to date. This can be interpreted to signify that carelessly made changes, not concerned with the timeliness or accessibility of the object, can transform the objects into less effective versions, collaboration-wise.

The concept of boundary objects has been developed, extended or contested in a multitude of studies. According to Henderson (1991), Lee (2007), Levina and Vaast (2005), and Nicolini et al. (2012), there are object categories that share similarities with boundary objects but fall outside the original definition of Star and Griesemer. First, Henderson (1991) proposes one of the extensions and defines the concept of 'a conscription device'. Conscription devices are objects that "enlists group participation, are receptacles of created knowledge and are adjusted through group interactions" (Henderson 1991, pp. 456). The study argues that boundary objects become too inflexible when a multitude of stakeholders are trying to apply the objects, whereas conscription devices can enlist more participation. The failure of boundary objects happens when the organisations applying objects try to encompass too much information into their objects. To remain relevant, the objects should maintain flexibility, or the phenomena they are trying to capture should restructure according to the object depiction. However, the conscription devices remain a subcategory of boundary objects and the differences between these two categories are not further explored.

The theoretical underpinnings of Henderson's concepts stem from Latour and Woolgar (1986), in the form of 'inscriptions', a concept that Henderson borrows to explain conscription.

Lee (2007) critiques that research has often black-boxed boundary objects. To address this issue, her paper expands on the concept of a conscription device and claims that such devices, adjusted through group interactions, are not boundary objects. According to Lee, the process of method standardisation has been detached from the objects and according to Lee's paper, boundary objects as originally defined do not sufficiently describe the materiality inside communities of practice. Lee proposes that the objects applied in the process of non-routine and novel collaboration are 'boundary negotiating artefacts'. Boundary negotiating artefacts are fluid and change status over time and contexts. Lee examines artefacts that were used to negotiate or cross boundaries, or to push boundaries further in an environment where the work was non-routinised and lacked established practices. Lee's rationale for distinguishing boundary negotiating artefacts from boundary objects into a separate category of potential boundary object predecessors relies on this difference in the nature of the work. Lee delegates boundary objects into routine work and boundary negotiating artefacts into non-routine work.

Levina and Vaast (2005) investigate the nature of the work and boundaries in their second major publication on boundary objects. In this paper, the authors discuss how boundary objects and boundary spanners can fail and succeed in their intended purpose. They show how objects that were designated as boundary object only became boundary objects, applied as intended by the members of the organisation, after actions of boundary spanners. Boundary spanners are people who were designated to this role and who had to actively perform their role to change the way the objects were applied. Levina and Vaast argue that designating an object or a person with the state of boundary object or boundary spanner is not enough: the objects and boundary spanners become 'true' boundary objects or boundary spanners only through action of the boundary spanners.

Nicolini et al. (2012) paraphrase Levina and Vaast (2005) and state that boundary-spanning activities, such as meetings and personal visits, support the role of the boundary objects and form around the object, but are also mediated by the boundary objects. In their study, Nicolini et al. (2012) attempt to draw from several theories that analyse collaboration and the material aspects of collaboration. Their clarifications of a few aspects of boundary objects are notable. They highlight the fact that not all parties have to share and understand all aspects of the objects in the same way. The article declares (pp. 617):

This example of the bioreactor also illustrates that boundary objects delimit the need to learn across the boundary of practice. This is because they carry details that can be understood by both parties, but neither party is required to understand the full context of use by the other because the object itself takes care of performing such mediation.

Nicolini et al. (2012) strive to investigate objects through different theoretical senses, such as boundary objects and Activity Theory, amongst others, in order to create a pluralistic framework of three tiers of objects that are used to facilitate cross-disciplinary work by enabling common understanding and providing basic infrastructures. Their study highlights the contextuality and multiplicity of the objects and the actions directed towards them by the different parties.

Finally, this multifaceted investigation of the objects concludes with an application of another concept by Star: the concept of material infrastructures (Star & Ruhleder 1996). The infrastructures, another aspect of Star's work on the

material, are linked to her later work on boundary objects. In the Star (2010) paper, she especially highlights the infrastructural nature of boundary objects. Star is not the only author that has aligned infrastructure with boundary objects. In their theoretical lens investigation paper, Nicolini et al. (2012) differentiate between two types of infrastructure. First, they discuss work-oriented infrastructure: infrastructure that establishes communication channels, such as project proposals or a project repositories. Next, they analyse service infrastructure: infrastructure that exists in the locations where the collaboration is taking place – in their example, chairs and meeting rooms. These infrastructures become visible only when they are not functioning up to the standards or expectations, or as Star and Ruhleder explain (pp. 113), infrastructure is "embodiment of standards. Modified by scope and often by conflicting conventions, infrastructure takes on transparency by plugging into other infrastructures and tools in a standardized fashion" and the infrastructure "Becomes visible upon breakdown. The normally invisible quality of working infrastructure becomes visible when it breaks; the server is down, the bridge washes out, there is a power blackout. Even when there are back-up mechanisms or procedures, their existence further highlights the now-visible infrastructure". Even if the environments are not meant as boundary objects or used as such, a breakdown can potentially transform an infrastructure object into a boundary object. *Table 8* summarises the literature of this section.

Topic	Focus Area	Authors
Original definition of boundary and discussion of the concepts. Extended by analysis of what is considered infrastructure and what is the function the infrastructure serves in team collaboration	Boundary object concept introduction, categorisation of different boundary objects. Star's 2010 paper provided more clarification of the concept and discussion on how it has been applied since its inception. Star and Ruhleder (1996) paper provides an analysis of different aspects of infrastructure in a multidisciplinary team.	**Star and Griesemer (1989), Star (2010), Star and Ruhleder (1996)**
Discussion of effective use of boundary objects, different types of boundaries	Analysis of different boundaries and uses for boundary objects to mitigate them	**Carlile (1997), Carlile (2002)**
Boundary-spanning activities and actors and analysis of how agents who act as boundary spanners created boundary objects that enabled new practices to develop	Levina and Vaast have done both analysis of the practice-situated boundaries and their causes and boundary spanners and boundary objects that become boundary spanners-in-practice and boundary objects-in-use.	**Levina and Vaast (2005), Levina and Vaast (2008)**
New boundary object categories discussed via use of sketches and drawings as tools for visual representations	Definitions of boundary objects subcategories such as conscription devices that enlist group participation and boundary negotiating artefacts designed to push the boundaries in complex, non-routine projects that lack standardised objects for collaboration	**Henderson (1991), Lee (2007),**
Boundary objects, objects that can precede or follow use of boundary objects and objects that can transform into boundary objects	Analysis of different theoretical explanations discussing the material elements of organisations and specifically infrastructures. Includes discussion through multiple theoretical lenses, such as Activity Theory, boundary objects and practice studies.	**Nicolini et al. (2012)**

Table 8. Literature on Boundary Objects and Boundaries

2.5.3. Application of Boundary Objects

One of the common applications of boundary objects has been to investigate team and project collaboration, especially collaboration in virtually organised projects, through the lens of this concept. A recent literature review of boundary objects as the tools for virtual team communication has identified several roles for boundary objects (Marheineke et al. 2016). According to the review, boundary objects can be used to negotiate meanings in order to fulfil tasks, can be applied for information processing, can affect people in collaboration processes and can bring structure to the collaboration. The review applies Carlile's (2002) three boundaries – syntactic, semantic and pragmatic – with the addition of emerging boundaries, in the classification of boundary objects literature. The review concludes by addressing the need for research of boundary objects as part of the work of the boundary spanners, the people who mediate the collaboration process and create new competencies.

The interconnected role of boundary objects and boundary spanners can create both positive and negative effects on their environment. For example, Di Marco et al. (2012) state that boundary objects can aid negotiations between virtual team counterparts. They highlight that especially visual boundary objects are helpful when negotiating over complex design artefacts. In a similar vein, Iorio and Taylor (2013) conclude that boundary objects are useful when conflicts arise in virtual project environments, especially if they are referred to early. Repeated applications of the objects during the conflict shorten the conflict and more strategic application further increases their efficacy.

While the accounts from Di Marco et al. and Iorio and Taylor highlight the positive aspects of boundary object application, the research by Barrett and Oborn (2010), on boundary objects in distributed teams, discusses the negative effects linked to use of boundary objects. In this study, shifting power dynamics during the projects created a situation when the control over the use of the objects was not equally distributed amongst all stakeholders. The objects were used to enforce the boundaries between the members of the organisations and the project ran into conflict situations (Barrett & Oborn 2010).

Remote and distributed work has been studied in a similar vein by Jonsson et al. (2009), who focused on the boundary-spanning capabilities of the boundary objects in maintenance work using remote diagnostic systems. The study states that the information system is not only a mediator of the information, but also has an active role in spanning the boundaries, creating new boundaries and reinforcing existing ones.

Building on Jonsson et al.'s (2009) research, Barrett et al. (2012) describe boundary relations created by the objects. The object, a pharmacy robot in their case, facilitated cooperation between one set of practitioners, but simultaneously there was neglect and strain between another set of practitioners. Introduction of the robot into the work environment elevated the status of the robot technicians, a group of boundary spanners, and the pharmacist working with the technicians. However, the previously mutually collaborative relationship between the pharmacists and pharmacy assistants deteriorated due to boundary neglect, or a lack of boundary spanners that would mitigate the boundary between these two parties, albeit that boundary spanning is not particularly discussed in this paper. Similarly, the relationship between the pharmacy assistants and the robot technicians was strained, due to increased fragmentation of the work around the object and again, if interpreted with the lens of the boundary spanning

concept, due to the lack of a boundary spanners taking action when it came to the technicians. In their study, Barrett et al. highlight the fluid and hybrid nature of the objects and how the adoption of the technological innovation was used to span and reconfigure boundaries. Their case study highlights both the positive and negative effects of the boundary objects and discusses briefly how the power structures shift in the organisation due to introduction of the objects.

The object in the Barrett el al. (2012) case does not undergo drastic changes, but other case studies have investigated the results of the changes in objects themselves. Subrahmanian et al. (2003) study the application of prototypes as well as other boundary objects in a distributed environment. Their case study shows that uncoordinated information flow can erode the usability of the prototypes. Another source of erosion was due to changes in the requirements and organisational changes. Reorganisation of their case project led to creation of new boundary objects, which were unclear to collaborating parties.

But if the erosion of the object is contained, prototypes as boundary objects can be used to bridge multiple boundaries between the organisations and benefit collaboration. For example, Winkler et al. (2014), address how prototypes can be used to bridge the syntactic, semantic and pragmatic boundaries defined by Carlile. The uses of objects include contrasting more complete prototypes with more abstract ones, visually and verbally exemplifying how the prototypes work and what functionalities are still missing, emphasising the openness of the prototype and facilitating feedback, and preventing misunderstandings.

In addition to research that investigates boundary objects' use in collaboration, there are more perspectives that have been addressed and other object categories that are similar but not quite the same. Gal et al. (2008) take a broader perspective to examine boundary objects in the context of the identities and practices of the organisations that use them. They investigate the interdependencies among organisational identities, inter-organisational work, and the boundary objects that support this work. Boundary objects can even impact the identities of other organisations working for the same project, the organisations 'across the boundary'. From the Agile perspective, this finding is significant. Gal et al. imply that boundary objects can not only be seen as tools for collaboration in Agile projects, but they can also help to create, maintain and distribute the Agile identity across organisational borders. Thus, the objects selected for the project should be aligned with the Agile values, especially flexibility and change, so that the project team using the object will not miss the opportunity to further grow its Agile identity.

In summary, boundary objects are powerful tools for collaboration (Di Marco et al. 2012) and can even be applied to extend organisational identities (Gal et al. 2008), but the changes of the environment (Subrahmanian et al. 2003), or more specifically in the power structures of the environment (Barrett & Oborn 2010), can erode the objects and hinder their abilities. ***Table 9*** summarises the articles discussed in this section. The next section will briefly discuss the Agile research and boundary objects and conclude the context and literature review section.

Topic	Boundary Objects Are Used For…	Authors
Boundary objects as tools in virtual collaboration	Boundary objects can be applied for conflict negotiation in a globally distributed, virtual project environmental but changes can lead to erosion of objects as useful means for collaboration. Uneven distribution of control can inhibit the use of boundary objects as tools of collaboration. They can be applied to mediate complex design knowledge across team boundaries	**Subrahmanian et al. (2003), Barrett and Oborn (2010), Di Marco et al. (2012), Iorio and Taylor (2013)**
Boundary objects as facilitators of cross-cultural communication and cultural identities and change catalysts for the organisational identities	Boundary objects are embedded in information infrastructures and help to form organisational identities. They can be applied to extend and change identities across the boundaries.	**Gal et al. (2008)**
Prototypes, both building prototypes and working prototypes as boundary objects in a software innovation project	Prototypes are transformed into boundary objects by 1. Contrasting building prototypes with working prototypes, 2. Visually exemplifying and pinpointing, 3. Verbally relating concepts to prototypes, 4. Verbally emphasising openness for adaptation and 5. Verbally narrowing down the scope for changes	**Winkler et al. (2014)**

Table 9. Literature that has Applied Boundary Objects

2.5.4. Boundary Objects and Agile Software Development

A handful of papers address both boundary objects and Agile software development settings. Boundary objects and Agile software development is a research area that has slowly gained more traction in recent years. In the earlier papers, the concept has been somewhat vaguely tied to the development methods. For example, in a paper by Baskerville et al. (2011), the boundary objects are briefly mentioned in the context of Agile development, but the focus of their paper is on existing ways Agile organisations operate and on speculating what would be the next 'big thing' in software development, in a post-Agile work environment. In their paper, Baskerville et al. (2011) argue that the Scrum method provides a number of different boundary objects, such as user stories, burndown charts and project plans; however, the paper does not describe boundary objects extensively and provides no further insights.

A more recent paper that addresses boundary objects in distributed projects presents boundary objects as means to create common grounds by promoting awareness (Modi et al. 2013). The study names user stories, shared code and test cases as objects that are integral to Agile development and can be seen as objects-in-use, borrowing the terminology from Levina and Vaast (2005). In this paper, Modi et al. (2013) see boundary objects as tools that aid grounding processes within the distributed teams and provide referential materials for the collaborating parties. The focus of the paper is an issue-tracker system that the teams use to manage workflows. The system itself is not seen

as a boundary object; rather, the objects are the user stories, the code that is being developed and the build platform, that is, the environment where the code resides.

Bass' (2016) paper discusses a similar environment setup to Modi et al.'s paper, that is, an Agile project in a globally distributed environment. The focus of the paper is on very large-scale projects and how such project organisations tailor their Agile methods and artefacts to better suit the needs of their environment. The study identifies artefacts in five different categories: feature, sprint, release, product and corporate governance. The study lists twenty-five artefacts that fit the five categories. Such artefacts include programme architecture standards, test plans, contracts, reference architectures, product backlog and different types of plans. Bass states that all the objects mentioned in the paper act as boundary objects, expanding the list of boundary objects to cover the majority of the artefacts, which are commonly applied in software development.

Topic	Boundary objects are used for…	Authors
Speculation on next evolution steps of software development	Boundary objects are important but not thoroughly discussed in literature	**Baskerville et al. (2011)**
Application of boundary objects in coordination and negotiating common ground	Boundary objects act as important vehicles for coordination in projects that ensure common ground and goals and provide common focus of goals. They act as mediators for communication, coordination and cooperation processes. They can aid in raising awareness, support the grounding process within the team and provide referential anchoring.	**Strode et al. (2012), Modi et al. (2012)**
Large-scale Agile project and the artefacts applied as boundary objects	Most objects that are applied in Agile development support boundary spanning and mitigate issues	**Bass (2016), Martini et al. (2016)**

Table 10. Boundary Objects in Agile Literature

A more succinct view on boundary artefacts and boundary spanning comes from Strode et al. (2012), who discuss coordination and boundary objects in collocated, Agile projects. The case organisations in their study apply mainly Scrum methods, but there is a dash of Extreme Programming in one of the cases as well. Their focus is on how boundary objects and other coordination mechanisms, can be applied to improve the effectiveness of the projects in cases where there are external parties, such as customers. Strode et al. propose a framework that splits Agile project activities into different categories of boundary spanning. According to their paper, the case organisation's boundary spanning consists of the boundary-spanning artefacts, boundary-spanning activities and coordination role (e.g. boundary spanner role). The coordination role, in Strode et al.'s study, is "a role taken by a project team member specifically to support interaction with people who are not part of the project team but who provide resources or information to the project" (pp. 1231).

Finally, Martini et al. (2016) present their findings from a study that focuses on collaboration of embedded software development teams: teams that exist inside organisations that focus on non-software products. The study stresses the need for boundary objects and boundary-spanning activities in Agile software development. It states "groups differing from the agile team appear to have different views and mindsets, which do not necessarily comply with ASD [agile software development]. This is hindering the development of boundary spanning activities and objects" (pp. 22). The study concludes by stating that organisations should apply more boundary-spanning activities, and thus boundary objects, to mitigate challenges that arise from project coordination issues, project speed and complexity.

Table 10 summarises the articles discussed in this section. Next section will present the conclusion for the context and literature review chapter of this thesis.

2.6. Conclusions for the Context and Literature Review

The first part of this chapter introduced the readers to the Manifesto for Agile Software Development, the values and disciplines listed in the Manifesto, and the most popular methods that adhere to the creed of the Manifesto (e.g. Beck 1999, Beck et al. 2001, Schwaber 2004). The second part of the chapter reviewed and analysed the differences in the ways the literature views Agile development. It presented a spectrum of perspectives that ranges from the very strict interpretation of the Manifesto and presented Agile as a fundamentally different paradigm from more traditional, plan-driven development (e.g. Nerur et al. 2007). At the other end of the spectrum are the authors who see Agile as a source of business benefits and a malleable tool for organisations to apply (e.g. Goh et al. 2013). This review of literature provides a basic understanding of extant perspectives organisations have towards Agile development and supports the search for the answer to my first research question, 'how does collaboration in organisations differ under the various approaches towards Agile?' The literature on Agile methods and Agile application in organisations also discusses a few ways of stakeholder configuration and ties to my second research question, 'how do organisations structure collaboration between Agile teams and business stakeholders?'

The second part of the literature review concludes with a discussion of the application of artefacts in Agile development, which leads to the more detailed discussion on boundary objects. The review of the boundary objects literature indicates that organisations that wish to ensure efficiency and efficacy of object application should plan and monitor how the boundary objects are utilised (Carlile 1997). Boundary objects have a significant role in Agile development, as the objects permeate many aspects of projects (Nicolini et al. 2012). Agile development projects, unique when it comes to pace and volatility of the work environment, can pose challenges for project organisations. The organisations have to ensure that artefacts are compatible with the chosen methods, as well as serve their purpose as creator of common understanding across the project borders.

Organisations who apply Agile software development methods are open to constant changes in the project requirements; however, the change in requirements can erode the ability of project participants to facilitate collaboration via boundary objects (Subrahmanian et al. 2003). Research on boundary object application in the Agile development environment will uncover strategies that organisations have taken to address issues stemming from the Agile environment. This part presents the groundwork for the analysis of my final research question, 'how are boundary objects used in collaboration between these stakeholders?'

With all these potential issues and caveats, how can organisations best apply boundary objects? The next chapter describes the steps I have taken to study the phenomenon of Agile software development with the theoretical lens of boundary objects.

Summary of Topics Discussed in the Literature Review			
Agile Perspective	**Agile as a philosophy**	**Agile in constrained environment**	**Agile as business benefit driver**
Definitions of Agile	A philosophy and an expansive metaphor of design, a worldview with philosophical metalevels, aligned with 'social relativism', readiness to change, creative and proactive	Designed to add energy, focus, clarity and transparency. An attempt to satisfy the industry quest for faster development processes. Can be beneficially applied in large-scale, globally distributed projects	Provides advantage to business but has a 'spirit', which needs to be maintained. Allows adapting to changing environment and provides better collaboration between developers and business.
Application of Artefacts in Agile	Studies on open source tools use, use of story cards and Agile walls, use of tools in large scale Agile projects		
Boundary Objects Definitions	Boundary objects are capable of bridging different kinds of boundaries but in order to be useful and effectively applied, they need to become boundary-objects-in-use. Objects, which are not designed to cross boundaries, can transform into boundary objects. They can be applied to enlist group participation or to push the boundaries.		
Boundary Objects Application	Boundary objects can be used as tools for virtual and globally distributed projects. They are applicable for facilitation cross-cultural collaboration, mediate complex design ideas, act as prototypes and facilitate common understanding, span organisational boundaries and mitigate issues		

Table 11. Summary of the Topics Discussed in the Literature Review

3. Research methods

"Clause nine of the Scribe' Oath: I will never stop learning and improving my craft!"

– Robert "Uncle Bob" Martin (2016)

3.1. Introduction

This thesis is a qualitative, interpretive, comparative case study that analyses three Agile development cases. The thesis is following well-established design of research on Agile development environments; almost half of the publications on Agile development environments have been qualitative (Hummel 2014). Qualitative case studies are an established perspective when it comes to qualitative research on Agile development (Dybå & Dingsøyr 2008; Hummel 2014). Qualitative case studies are especially well suited when the research is looking for answers to 'how' questions (Walsham 2006).

As stated in the introduction, the research described in this thesis is seeking to address the following research questions:

1. **How does collaboration in organisations differ under the various approaches to Agile?**
2. **How do organisations structure collaboration between Agile teams and business stakeholders?**
3. **How are boundary objects used in collaboration between these stakeholders?**

Next, I will explain why an interpretive approach was chosen to address these questions. This is followed by a description of how I defined my research cases. Then, I will discuss why I chose to follow the structured-pragmatic-situational research (SPS) perspective by Tan and Pan (2011), an iterative method for qualitative case study research. After the introduction of the method and a brief discussion on why the case organisations were chosen, the rest of this section follows the SPS structure. I begin with explaining my access negotiation process and the conceptualisation of the phenomenon, Agile software development. Next, I will describe the data collection and explain why the first two case studies were based on interview data, whereas the third case includes additional data obtained by observations. Finally, I will describe the data analysis and theorising process and provide examples of how these two steps were iteratively conducted throughout the whole process.

3.1.1. Interpretive Perspective and Casing of the Data

The theoretical lens of my research, boundary objects (Star & Griesemer 1986), is a concept that originated from Latour's Actor-Network Theory; however, the origin of the concept provides no direct anchoring to either positivist or interpretive world-view. This research seeks to understand how different aspects of Agile development have influenced the development projects and how organisations have approached the development methods when it comes to their collaboration and object use. All three areas are asking 'how' people interpret the Agile Manifesto and the development guidelines outlined by the proponents of the methods. Thus, the research question format and goals of the study point to an interpretive direction.

Interpretive research takes a position that an understanding of reality and human action is socially constructed, albeit there is an objective reality as well (Walsham 2006). However, the objective reality is always seen through a lens of an interpretation of humans who construct their worlds from the fragments of the observable reality. Walsham is one of the early advocates of interpretive perspective in IS research. Walsham draws on the work of anthropologists and organizational theorists, such as Geertz (1973) and Van Maanen (1979). Walsham does not provide the reader with clear instructions on how to analyse data or how many cases should be selected; instead, he gives advice on theory selection, stating that the researcher should choose a theory that feels insightful, and provides recommendations of how to conduct fieldwork, stressing good social skills (2005).

Walsham's perspective of case study research reflects how I see my case studies. In order to understand how collaboration and boundary object application are seen in organisations, especially when framed by a subjective interpretation of the Agile values and principles, I have to rely on the interpretation of others. Members of organisations have their own views on what Agile is and how it should best be applied. Subjective views lend themselves poorly to quantitative research. Surveys or other quantitative methods would only provide superficial understanding on the subject matter and theorising from such data without thick descriptions would not provide insights. An interpretative approach is also an appropriate way to analyse boundary objects, which are an interpretive concept as well. The concept depends on the context and perspective. Where one draws a line between organisations or teams is intersubjective and determined in philosophy rather than practice by, for example, the 'field of practice' (Levina & Vaast 2008).

3.1.1.1. Unit of analysis

The interpretive approach and the difficulty of the boundary definition introduces complexity when it comes to the definition of units of analysis. Commonly in case studies, the organisations that are studied are seen as the basic units of analysis. Most case studies of boundary objects and Agile projects, discussed in the literature review, treat single organisations as units of analysis. However, in my three research cases, this definition would only work for the first case study. The other two cases are illustrative of collaboration between two organisations or between a multitude of customers, partners and vendors. In other cases, there were multiple organisations involved in the work. A project or a program could frame another unit of study but in this case, the first case study would not fit the mould as the organisation was based on product development.

Defining that single organisation that forms the outlines of cases is not applicable in a study where multiple organisations form partnerships and collaborate across boundaries; instead, I turn to the concept of 'casing' by Ragin (1992). Ragin describes casing as (pp. 218):

…consider cases not as empirical units or theoretical categories, but as the products of basic research operation. Specifically, making something into a case or 'casing' it can bring operational closure to some problematic relationship between ideas and evidence, between theory and data. Casing, viewed as a methodological step, can occur at any phase of the research process, but occurs especially at the beginning of a project and at the end. Usually a problematic relation between theory and data is involved when a case is declared.

As Ragin suggests, I have limited my focus to the phenomenon rather than an organisation or a project, the phenomenon in this case being the application of Agile methods. I have iteratively updated my understanding of

what the cases consisted of when new information was acquired. My original casing was shifted and modified as the research progressed, as Ragin (1992) states it should.

The casing of the first case was clarified to me during data collection, when I understood that Agile method application was uniform across the organisation, prompting me to investigate the whole organisation as one case. In the second case, the limitations of the case were easy to define, as the Agile activities were cased in the project conducted by the two organisations involved. Conversely, in the third case, the organisation was applying Agile methods across all different departments, but the programme I was introduced to was established as a separate endeavour inside the organisation. Originally, I had very limited understanding of the program stakeholders, but once I understood the complexity of the program and the multitude of stakeholders, I had to expand the casing and include members across the different organisations when conducting the data collection.

3.2. Iterative Research Perspective

My research has been influenced by both the context of the research, Agile development, and the selected case study approach perspective. First, studying the Agile process has greatly impacted the way I have approached my own research. The flexible and open Agile mindset that invites change, improvement, self-reflection and feedback lends itself perfectly to knowledge work such as research projects. In addition, I discovered that the tools used by Agile projects, such as Agile task walls and the practice of continuous releases were also applicable for writing and theorizing. I tested these methods by using Post-it notes on a wall to clarify the sections of this thesis and tracked my progress against deadlines in online Agile tools. I have also included photos and illustrations in the case description that can act as boundary objects between the reader and me, to convey information as only pictures can. I believe that writing for conferences and journals during this the PhD thesis process could be considered as akin to continuous delivery. A curious reader could track the evolution of the boundary objects framework from my first paper, Zaitsev et al. (2014), to another paper published two years later (Zaitsev et al. 2016). Constant feedback on the framework has shaped my thinking and clarified the concepts, improving the framework step by step. Application of Agile methods to the research process further convinced me that Agile methods are more than a set of best practices: rather a mindset, a perspective towards work. This exercise, I believe, has helped me to understand the employees of the case organisations better.

Secondly, I followed the eight cyclical steps of structured-pragmatic-situational research (SPS) (Tan & Pan 2011), which describes qualitative case research, with interviews as the main source of data. The SPS approach is iterative. The steps guide researchers through the process of obtaining access to organisations, framing the research question(s) and theorising from the data until the write-up of the report. I found the SPS approach useful, as it gives a comprehensive representation of the research process. Research is an iterative process, which sometimes requires reframing, or refinement of concepts. The SPS approach does not mask the messiness of case study research, nor does it take a strong stance in the continuum between positivist or interpretive research. It describes the research process and coding strategies neutrally, drawing on both positivist traditions of Eisenheardt (1989) and Yin (2013) and interpretive methodologists Walsham (1995; 2006). The neutral outlook of the SPS approach is thus not misaligned with selecting the interpretive perspective on the study, but provides a neutral approach from which one can further refine the research perspective. In addition, the selection of this approach was further encouraged by

other research where the SPS approach has been applied in a similar setting, in previous research of Agile methods (Goh et al. 2013).

The eight steps include 1. access negotiation, 2. conceptualising the phenomenon, 3. collecting and organising the initial data, 4. constructing and extending the theoretical lens, 5. confirming and validating data, 6. selective coding, 7. ensuring theory-data-model alignment, and 8. writing the case report. The steps, along with the iteration cycles, are illustrated in ***Figure 3***.

I will provide examples to clarify how I applied each step, but first, I will detail the reasoning behind the case selection. This process diverges slightly from what Tan and Pan suggest in their paper, as it discusses additional research effort that was later excluded from the final thesis.

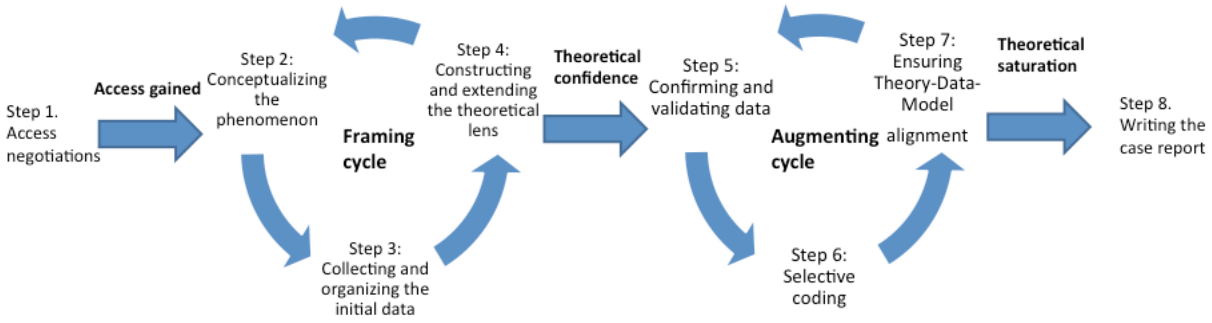

Figure 3. The Structured-Pragmatic-Situational (SPS) Approach (Tan and Pan 2011)

3.3. Case Study Design / Sampling

The first step in my study diverges from the SPS framework when it comes to the question of why I engaged the three organisations and what their selection criteria were. Pan and Tan suggest that cases should be selected based on how interesting the cases are. 'Interesting' is a subjective term: one might not be interested in Agile development at all. However, the fact that Agile software development has gained such prominence and significance amongst software development communities (VersionOne 2017), but is yet to be properly researched and understood (Conboy 2009; Hummel 2014), makes it an interesting topic.

The different ways of implementing Agile were equally interesting. Even more interesting were the differences in the methods and their implementation. Why were organisations so different when it came to Agile practices, when they were all based on the same sources and methods? Comparison between a variety of organisations and their projects was a reasonable approach, so I decided to engage with organisations which all had Agile experience and a good Agile reputation amongst the Agile community, but had different approaches when it comes to method implementation.

I set out to find organisations that would represent different approaches towards Agile, as was discussed in the literature section. Based on my analysis of the literature and research into the Agile community via blog posts,

conferences and Agile meeting discussions, I discovered that there was an ongoing debate of what was considered 'true Agile'. I wanted to find examples of the different types of organisations that would correspond to different Agile approaches and research whether the different paths to Agile development would manifest in different methods, collaboration and use of artefacts; however, the process was not straightforward. In addition to the three case organisations further detailed in this study, I negotiated access to three further organisations. From the six potential research organisation candidates, I conducted a few preliminary interviews with members from two of these organisations, but was eventually unable to secure enough interviews. In addition, these two organisations were similar to my existing cases, so I decided not to pursue these two leads further. With the third organisation, I conducted several lengthy interviews, but eventually it became clear that too much time had passed since their major Agile projects. However, this case provided me with invaluable information about the Agile community and the overall state of Agile projects in Australia.

The remaining three case studies described in detail in this thesis were thus not conducted 'accidentally' to fit into the three categories discussed in the literature review, but were specifically selected from all the leads I had to present different Agile approaches. The interviews conducted with the members of the other three organisations are not included as part of the data in this thesis, but these discussions provided me with an invaluable understanding of the Agile community and the roles of the methods in different industries, improving my contextual understanding.

3.4. Step One: Access Negotiation

The three case organisations, inhabiting different industries and different privacy policies, required different access negotiation processes. The common theme with all of the access negotiations was a proactive approach. I immersed myself into the Agile community of Sydney, a community which was only partially familiar to me before my research project. I participated in community meetings, volunteered at technical conferences and made my project known to anyone who was willing to engage. After a while, my efforts were rewarded with multiple leads and from these leads, three organisations were willing to let me study their projects and practices. Interpersonal skills, as described by Walsham (2006) were the key component in the access negotiations. As a doctoral student, I could not promise the organisations many benefits; instead, I described the study as an opportunity for self-reflection for the employees and potentially as a safe, confidential discussion environment where they could express their frustrations. Presenting myself as a former peer and a curious outsider worked well: interviewees went into details when describing practices and delivered candid accounts on the challenges in the organisations – more candid that I would have expected, considering that I was a complete stranger to most of them.

The access negotiations with the first case study organisations were straightforward. The organisation conducting internal development allowed me interview access and opportunities to snowball sample within their organisation. The majority of the people who were suggested by their colleagues agreed to the interviews without scrutiny.

The second case study required access to two organisations in a customer-vendor relationship. My initial contact was with the customer organisation and its management had no objections when I asked if I could also engage with their vendors to get a more complete overview of the case. The vendor organisation members were equally happy to take part in the study and they were open when discussing their relationship with the customer organisation.

With the third case study, I had an agreement that allowed me to both interview the members of the organisations involved with the work and observe meetings. I had a contact person who would let me onto the premises and I was allowed to wander around, see the Agile artefacts and observe a variety of meetings taking place in the premises.

3.5. Step Two: Conceptualisation

The next step in the research project was the conceptualisation of the phenomenon. Tan and Pan suggest that the researcher should read the 'non-technical literature' in order to familiarise themselves with the organisations and the phenomenon, in this case, Agile development (pp. 116). They also suggest that the researcher should read different theories, develop a basic understanding of the potential explanations there already are or find suitable theoretical lenses. Although I was already familiar with the concept of Agile development, I reread the 'classic' Agile guidebooks and papers, discussed in the literature review section.

From a theoretical perspective, I familiarised myself with the pertinent theories used in the information systems discipline. From the very beginning of the project, I had been fascinated by the theories discussing the materiality of software systems or the effect the material or immaterial actors had on development. Reading extensively about approaches such as Actor-Network Theory (Latour 2005) led me to the concept of boundary objects (Star & Griesemer 1989), which eventually became the main theoretical lens for the project. Armed with this theoretical lens, I set out to collect data from the organisations.

3.6. Step Three: Data Collection and initial Data Organising

The third step of the SPS is the data collection and data organising. This step went through several iterations. The data was collected in two ways: for the first two cases, the primary data collection method was semi-structured interviews; the third case study data is a combination of interview data and observations. The following section discusses these two data collection methods in details.

3.6.1. Data Gathering with Interviews

Semi-structured interviews were my main source of data collection. Snowball sampling (Biernacki & Waldorf 1981) – where interviewees directed me to the next person – was applied in all cases. For the actual interviewing process, I followed interviewing guidelines as described by Myers and Newman (2007). I briefly introduced my personal background, without going into too many details. I usually half-jokingly said that I was conducting the research in order to find good examples of Agile to counter my own experiences in waterfall projects, in order to set the scene to the interviewees. They could now use the technical jargon with ease and describe the events in their own terminology, as I shared the understanding of the language but described myself as an eager researcher, not an Agile authority of any kind. My attire was chosen to reflect the informal clothing of the organisations, usually with some quirky detail that the interviewees would expect from a researcher. Intentional use of 'nerdy' clothes or notebooks was sometimes a great icebreaker and conversation starter. Myer and Newman refer to this strategy as situating the researcher as an actor and minimising social dissonance, which was not difficult when I was already a member of the community.

The interviews themselves encompassed a wide variety of different people in different roles and each interview unfolded uniquely, but I did have a list of prewritten topics and questions, as summarised in ***Table 12***. The purpose of the list was to ensure that each interview covered similar topic areas, but the list was often unnecessary as the interviewees discussed the majority of the topics without any probing.

Interview topic	Purpose
Background information Name and role, experience in organisation and industry	Relaxing the interviewee, building trust
Software project methods Management and development methods, experience in methods	How the interviewees see Agile methods
Software projects/products What is the project/product, goals and expectations, challenges	How the interviewees see the purpose of their work
The team and stakeholders Daily, weekly and rare communication, rules and processes	How the interviewees communicate
Tools and documentation Communication tools and documents, when are they used, likes and dislikes, method support	How the interviewees apply tools

Table 12. Interview Topics and Their Purpose

For this research I drew from Walsham's (2006) suggestions on how to capture interview and other data. I did record all the interviews and the tapes were later transcribed, but I did also take detailed notes during the interviews, where I also collected signs of unusual body language or other things that might be lost with only recordings. As Walsham suggests, my interviews were complemented with background research on secondary data. Every time I would interview people from new organisations, I would first conduct research on the organisations and familiarise myself with what was publicly available. The organisations, their pseudonyms, interview participants and the fieldwork details are summarised in ***Table 13***.

Chapt.	Organisation pseudonym	Industry	Interview participants	Fieldwork	A case of…
4.	Extreme Inc.	Banking and finance	Technology lead interviewed twice 6 team/delivery leads 4 developers/designers 3 business stakeholder/product manager Department lead Total: 16 interviews	Office visits	Agile methods in collocated product development
5.	Escapade	eCommerce	Project manager, interviewed three times CFO interviewed two times 3 project Sponsors or other managers 4 testers or users **Total:** 12 interviews	Office visits	Agile methods in virtual environment between a customer and a vendor
	Carmine	IT consulting	Manager, UX designer, Developer **Total:** 3 interviews		
6.	PrecautionCorp.	Banking and finance	8 tech lead/iteration manager/business analysts 2 Agile coaches, both interviewed twice Lead Agile coach 4 program manager/business stakeholders, 3 other members of the organisation **Total:** 18 interviews	Observations of several meetings, other office visits	Agile methods in large offshore and local product development and process change program

Table 13. Summary of the Research Case Organisations

3.6.2. Data Gathering with Observations

The third case study was exceptional in two ways: I was allowed to follow the program for almost a year and I was also invited to observe the meetings of the program. The details of the arrangement are discussed in the case study description.

Observation of development projects and ethnographies does have a small but interesting part in the research of Agile development. Hummel (2014) claims that there have been sixteen ethnographic studies in the field of Agile

research. Even if the numbers are not large, ethnographies such as Sharp and Robinson's (2004) and Chong's (2005) studies on teams who practice Extreme Programming provide detailed accounts of the daily work of organisations not unlike my second case study subject, Extreme Inc.

For this thesis, I have made sure that the data from the field notes is authentic, plausible and critical (as discussed in influential IS ethnography by Schultze's (2000)), but the sporadical nature of the observations is why I would not describe this as a full ethnographic study. However, the opportunity to observe multiple meetings with a variety of stakeholders could be considered as a complementary feature in my research access. The observations led me to understand the undertaking vastly better than only interviews would. I was also able to non-invasively observe people who would probably never have appeared on my radar through snowball sampling, due to their roles or personalities. These observations provided me with data that was crucial for understanding the complexities of the program and as a source of interview data that was not prompted by my questions.

3.6.3. Organising Data

Throughout the data-collection phase, the data was organised thematically according to how the objects were applied. Organising the data from the very beginning of data collection ensured I was aware of the broader themes and helped me when it came to conducting the interviews. The first cycle of thematic analysis, emerging from the first interviews of the first case study, focused on the very basic day-to-day use of the objects and the practices linked to their application e.g. what kinds of objects were used, when and in which situations.

The second cycle of analysis was conducted after the second case study had begun and focused on the collaborating partier rather than the objects themselves. The third thematic data analysis cycle was conducted after the third case study was completed. This analysis provided themes such as personal Agile experience, corporate culture, histories of the organisations, and infrastructure and operations emerged organically during the first case study data analysis. The new, emergent themes were incorporated in the later interviews with other case organisations. The new cycle of interviews and data analysis was coded against the new themes until no more higher-level themes emerged. This preliminary organisation of data was especially helpful when it came to the more detailed coding, described in the next step.

3.7. Steps Four, Five and Six: Constructing, Validation Cycle and Coding

The next three steps in the SPS model are 4. constructing and extending the theoretical lens, 5. confirming and validating data, and 6. selective coding. I began the data analysis in tandem with the data collection, as suggested in the SPS approach. The data from the first case study was compared to the insights gained from the literature on Agile software development and boundary objects. For example, literature discussing different perspectives on Agile software development was helpful when analysing the views of the methods held by the informants. Similarly, the boundary object literature helped in identifying the boundary objects and the boundaries these objects were bridging. However, the frameworks developed during the study are novel and the literature was applied to guide the data analysis, not as direct source for the coding or themes. The coding is illustrated in **Table 14**.

First Phase of Coding	Second Phase of Coding	Themes and Theoretical Lenses
Role and personal history Individual roles: team leads, scrum masters, iteration managers, project managers, analysts, testers, designers, product owners Teams	Organisation background and structure Different roles and stakeholders	**Themes** Background and day to day operations of the and the overall Agile perspective of the organisation members **Theory** Boundary objects Agile methods
Organisation culture Organisation change/organisation history Hiring practices Working from home\	Corporate culture and how Agile is applied in the organisation	
First Agile encounter Thoughts on Agile/What is Agile Agile training in organisation Agile process change	Personal Agile experience and perception on practice	
Internal communication External communication Contractor communications Sales and marketing Risks/issues/concerns	Communication practices across different boundaries	**Theme** Organisation of the stakeholder communication **Theory** Boundary objects enhancing collaboration Agile methods enhancing collaboration
Stand-ups/Scrum of Scrums Pair programming Retrospective meetings Other meetings	Applied Agile practices are part of the communication between stakeholders	
Planning Requirements and estimation Design	Product plans and designs as part of project initiation	**Themes** Application of boundary objects in order to collaborate with stakeholders **Theory** Boundary objects enhancing collaboration Agile methods enhancing collaboration
Contracts/business cases Walls/Jira use Chat tool/Flowdock use Prototypes Wiki/Confluence use Testing tools or other tools	Application of artefacts in communication between stakeholders	
Infrastructure Deployment Regulations/Audits	Infrastructure and operations as part of the development process	

Table 14. Interview Data Codes, Themes and Theoretical Lenses

Personal preference led me to forgo qualitative data-processing tools such as NVivo. Instead, I coded the textual data from the transcripts and observation notes in an Excel spreadsheet and performed the second phase of coding with a low-tech solution, Post-it notes (Strauss & Corbin 1990). During the first phase of data coding, I listened the interview taps, read the interview transcripts and noted taken during the interviews, and assigned a codes to the data based on the topic each paragraph, section or phrase was discussing. In the next phase of coding, I organised the data according to the larger themes emerging from the first phase of codes. These larger themes were finally organised into the major themes of the thesis: Agile perspectives, collaboration and boundary objects application from which the theory, discussed later in the sections 7.4 and 7.5 begun to emerge.

In addition to the coding in spreadsheet, I applied visual data analysis techniques such as visual mapping of the major events and stakeholder groups as well as writing up narratives of the events allowed me to trace application of the Agile methods and artefacts in each case (Langley 1999). I drew timelines of events that took place in each organisation, figures that captured different stakeholders and their relationships as well as process diagrams, which illustrated the use of objects, and Agile practices. These visual aids supported the thematically coded data by providing the linkages between the different themes.

Data validation and the sufficiency of data is a difficult topic in qualitative research. Myers and Newman (2007) state that a variety of voices is needed, Tan and Pan (2011) give an exact number for minimum interviews where as the positives authors such as Yin (2013) and Eisenheardt (1989) discuss triangulation and multiple data sources. However, the interpretive authors, such as Walsham (2006), give no definitive guidelines on when to finish the data collection. For my study, there were a few different ways to ensure sufficiency of data.

For the first case, the saturation was reached when I had interviewed someone from every tribe and almost every role in the organisation, excluding only the upper management. For the second case study, the process was simple: I interviewed every major stakeholder of the project, save two who were not available for interviews. In addition to the major stakeholders, I interviewed some people who were temporarily involved in the projects and had insights into the processes and tools. For the third case study, the program organisation was very complex. Interviewing someone from each role across all different streams was not feasible so I focused on the agile coaches and iterations managers who had good oversight on the daily work. In addition, I managed to discuss with few business stakeholders, business analysts and testers across the organisations. In the third case study, the observations supplemented the interview data and confirmed that I had understood correctly the daily program activities.

3.8. Steps Seven and Eight: Ensuring Theory-Data Model Alignment and Writing

The last step of the SPS approach, before writing up the case study, is ensuring that the theory and data model are aligned. This is again an iterative, cyclical process. New pieces of data are compared to the existing theoretical models and if the models are unable to explain the data, the models should be modified. If the data would point to a new type of boundary object that was actively used in one case but not another, the models should be re-evaluated and updated. For example, the participant of the third case study had a more intricate stakeholder structure and used more boundary objects to convey information between stakeholders. These new objects revealed new insight into how the objects forming the project infrastructure – the development, testing and production environments – can

help or hinder the development. Each new case and the subsequent data analysis have brought new insights and the theoretical models have been constantly updated.

Validation for the models has been sought from my key informants in the case organisations, as well as from peer-reviewed conferences in the form of reviews and presentation comments. The models presented in the discussion section of this thesis have transformed from very rudimentary classifications to their current form, in a collaborative effort between myself, the Agile practitioners and the academics in the field of information systems.

The writing in this thesis loosely follows the conventions of the information systems field and the structure detailed by Tan and Pan. Some artistic liberties have been taken when structuring this chapter, but the other chapters follow the standard structure. In order to maximise clarity of the case descriptions, each case chapter follows the same patterns. After the chapter introduction, the organisations are introduced. This is followed by the description of my data collection engagement. Next, quotes that exemplify the Agile approaches and methods of the organisations are presented. After the section on each case's Agile approach, I present the daily activities and the objects applied. The collaboration with the stakeholders is presented after the daily work activities. Each case study chapter is finished with a chapter that ties together the cases and the concepts discussed in the literature section.

For the discussion section, a similar format is applied. The first part compares the Agile approaches, the second objects and methods, and the third compares the stakeholder interactions. These comparisons are then synthesised into a theoretical framework and the thesis is concluded with a traditional set of conclusions, limitations, future research and analysis of the thesis contributions.

3.9. Conclusion

This section has described the research approach and methods that were applied when conducting the three case studies that form the basis of this thesis. First, I detailed the research perspective: why an interpretive research approach was chosen. I examined the notion of casing and the application of casing in this research. Next, I described the rationale for selecting the organisations studied. The following sections were organised according to the eight steps of the SPS approach: 1. access negotiation, 2. conceptualising the phenomenon, 3. collecting and organising the initial data, 4. constructing and extending the theoretical lens, 5. confirming and validating data, 6. selective coding, 7. ensuring theory-data model alignment, and 8. writing the case report.

The next three chapters will describe the case studies. Chapter four describes the daily activities of an organisation with an Avid perspective towards Agile development. Chapter five details methods applied in a project that took place between a customer and a vendor organisation with an Inclusive perspective towards Agile. Chapter six delves into the elaborate structures and methods applied by a large program organisation with a Pragmatic Agile approach.

4. Case Study One: Never Out of Sight, Out of Mind

"With great freedom comes great responsibility – to paraphrase Peter Parker's uncle."

–Test manager, Extreme Inc.

4.1. Introduction

The first case study describes Agile practices in a company which, unlike many other companies that have adopted Agile gradually, was originally founded on Agile principles. The origins of the organisation and the rigorous application of one of the more demanding Agile methods, Extreme Programming, makes Extreme Inc. (pseudonym) a rarity even amongst other Agile organisations where the Scrum method is more prevalent (VersionOne 2016).

In this chapter, I will first provide background information on the organisation and the roles and teams. This is followed by the description of my research with the organisations: who I interviewed and what events took place during my engagement. Next, I have dedicated several sections to a description of how Agile methods, and specifically pair programming, were implemented in the organisation. The overall methods, pair programming, Agile walls, communication and the artefacts that were applied to support the communication are all described in detail with corroborating data from the interviews.

In the last section of this chapter, I will respond to my research questions. I will discuss how perspective towards Agile was manifested and what was the impact of the perspective, how collaboration was organised, and how it was supported by the boundary objects' application. This preliminary discussion is further developed in the cross-case analysis of the discussion chapter, which ties all three cases together and answers the research questions.

4.2. Extreme Inc.

Extreme Inc. is a medium-sized software development company operating in the financial sector. The company develops all products in-house and the organisation consists mostly of developers and other technical experts. The non-technical departments of the company include marketing, customer services and the legal team. All of Extreme Inc.'s employees are located on the same premises, an office space that spans three floors.

Extreme Inc.'s software development work is divided into product tribes and support functions. The organisation was structured after the "Spotify model of organisation" (Kniberg & Ivarsson 2012), dividing the employees into tribes, based on the three main product offerings of Extreme Inc. Each tribe consisted of four teams, which comprised roughly ten members each. A typical team included six or seven developers, a tester and a technical team lead. In addition to tribes, some members of the organisation were part of cross-functional specialist teams. The cross-functional teams were concerned with security upgrades, process refinement or other business improvement. The members of the cross-functional teams split their working hours between their usual work on the products and their specialty areas, depending on their workload and the priorities set by the teams. In general, there were no deadlines for product delivery, save a rare instance when an external stakeholder, for example a regulatory

stakeholder, demanded completion of a task. The focal point of the work was the product, not projects; however, the organisation had internal project initiatives that were run in parallel with product development. For example, during my interview period, Extreme Inc. was strongly investing in their incremental product development and delivery capabilities. One of the goals for the specialist teams was to ensure that the continuous delivery practice (as described by Humble and Farley 2010) was constantly improved from both the technical and process standpoints.

The interviewees described the organisational hierarchy at Extreme Inc. as very flat. My informants considered the culture of Extreme Inc. to be a successful reflection of the values and principles of the Agile Manifesto (Beck et al. 1999). The application of the Agile method 'Extreme Programming' (Beck 1999; 2000) was a dominant force throughout the entire existence of the organisation. In the next sections, I will elaborate more on those I interviewed and how Extreme Programming was applied by the organisation.

4.3. Extreme Inc. Interviews

I first encountered the technology lead of Extreme Inc. at a technology conference, where he was manning a recruitment stand. My interest was piqued when I learned that the organisation was successfully implementing pair programming. Agile literature has examples and studies of XP in practice (see Sharp et al. 2009), but never had I witnessed pair programming with my own two eyes nor discussed it with anyone proficient in the practice.

My enthusiasm regarding the subject matter paid off when the technology lead invited me to interview the members of his organisation. Snowball sampling directed me into discussions with people from different teams and across multiple organisational roles (Biernacki & Waldorf 1981). First, I conducted 13 interviews with people from the development teams, as well as the company's internal business stakeholders. The informants were interested in and supportive of the research. I had no trouble reaching out to the people I was referred to by other informants. Open-ended interview questions (Walsham 1995) revealed the manner in which the informants perceived the Agile methods at Extreme Inc. and what daily activities they were engaging in. During the interviews, I took extensive notes but also digitally recorded and then later transcribed the materials for coding and analysis (Walsham 2006). These notes were an important reminder when I had to check what was discussed in the previous interviews and the transcription was not ready. Transcribing recordings is slow and the transcription of the recordings was not done immediately after each interview but after a set of interviews. The notes were an invaluable, quick data source and facilitated better data quality, as no interviews were meaningless or shallow (as addressed by Biernacki and Waldorf 1981). ***Table 15*** summarises the informants, their roles in the organisation and which themes were discussed during the interviews.

After the first few months, I had created a preliminary theoretical framework and formatted several ideas based on the interviews, and I decided that it was time to have another discussion with the technology lead. The technology lead agreed with many of my observations, but also pointed out two more interviewees whose experience could provide supporting evidence for my theories and who worked in an area my previous interviews had not explored.

Role	Informants	Themes discussed
Interviews from April to July 2016		
Business stakeholders	Head of engineering Department lead Business stakeholder/product manager	Organisational culture and history, hiring practices, role of involved stakeholders, communication tools and mediating artefacts used, Agile practices enacted, audits and regulations
Agile practitioners	Six team/delivery leads Two developers/designers	Systems development process, communications, communication tools and mediating artefacts used, Agile practices enacted, pair programming
Additional interviews in September 2016 **indicates that the person was interviewed a second time*		
DevOps and head of engineering	Head of engineering * Two DevOps managers/leads	Systems development process, communications, communication tools and mediating artefacts used, Agile practices enacted, audits and regulations, verification of preliminary theoretical ideas

Table 15. Extreme Inc. Interviews

A few months prior to the interviews, the organisation had launched a new product initiative and a new product team. The restructuring of the organisation into tribes had also been conducted somewhat recently. In my interviews, I managed to interview representatives from the three product tribes and the DevOps tribe, as well as a group of people from supporting functions such as auditing, risk management and sales. All the interviews were conducted either in the office or in the vicinity and I was treated to several visits and tours around the Extreme Inc. premises.

The rapid organisational growth was one of the prominent themes and it was demonstrated in a very tangible way: when I began my interview process, the office building was in the midst of turmoil. The company was taking up an additional floor and relocating some of the teams from the existing office, which was running out of desk space. A new spiral staircase was being installed in the middle of the office for easier access.

I observed briefly a few of the Agile meetings taking place and witnessed the Agile walls, status screens, charts and other illustrations that were scattered around the three office floors. Many of the interviewees were quite new to the organisation: Extreme Inc. had employed some for a few months only; another prominent feature was that the employees were young, some only recently graduated. The look of the office was reflecting the 'young and hip' crowd: the office was very modern, open and airy, mostly white but with brightly coloured accent furniture items. There was constantly a level of background chatter, buzzing and energy. The layout of the company was structured to accommodate the pivotal element that distinguished Extreme Inc. from other, similar organisations: the pair programming method.

4.4. Agile Approach: Extreme Programming

There are a few distinct features that set Extreme Programming apart from other Agile methods. The Extreme Programming guidelines are presented in ***Table 2*** in Section 2.3.1.6. Collocation of the team and customers, internal

or external, in an open workspace with minimal physical barriers is one of the standards of Extreme Programming (Beck 1999). Another predominant way of working is pair programming, already explained in the literature review. At Extreme Inc., pair programming was a non-negotiable core convention of the organisation. It was seen as the main source of communication between the developers, as a means to train more junior team members and as the method for ensuring code quality. With pair programming, there were always two or more pairs of eyes on every line of code. The technical lead explained how he had introduced the method at Extreme Inc. several years ago:

I thought I had a serious chance of actually getting away with different things because it was all new. Every other company I have been to… It was: 'here is how we do it', Trying to change the way of working was really not possible. At Extreme Inc., I was the second engineer that was hired… So I gave the XP book [Beck 2000] to the founders of the company and they all read it. They were all engineers as well and they could see the benefit in the XP so we took the method on.

My interviewees stated that Extreme Programming methods were demanding and not suitable for everyone. At Extreme Inc., the recruitment process was specially tailored to identify which personalities would fit the teams and the intensive pair programming practice. The pairing was done between the developers but also sometimes between the developers and testers or developers and designers. A manager in DevOps described the organisational culture and what the organisation was looking for in their new hires:

Respectfully challenge people in an environment where we foster that. I mean obviously being open minded to different ways of thinking. No ego. People who come in an ego, hopefully don't pass interview system. And willingness to help people, I think is a big thing. Because we collaborate a lot. So, having people from different parts of the business walking up to you, interrupting you when you're in the middle of something, it happens all the time. So if you have a mentality of 'no, go away, come back in ten minutes or twenty minutes', doesn't really cut it here.

Over time, the methods had been tested, trialled and adjusted at Extreme Inc. Even though Extreme Programming Explained: Embrace Change (Beck 2001) was still required reading for new hires, the interviewees admitted that they were not strictly following everything that was prescribed in the guidebook. Other methods, such as Scrum or Kanban, had influenced ways of working. What was consistently emphasised was the notion of Agile as a set of values rather than just the method. Several interviewees had very strong opinions on Agile and what it meant to them. One of interviewees described the culture enthusiastically:

Here, everyone just does Agile. It is the way of life here. It is the most easy-going and transparent place ever. And everyone works really hard while they're here, but they only work nine to five. And there are no emails outside and there are no political games. People just want to get stuff done.

The Agile methods, especially pair programming, were often cited as both the cause and the effect of the open, collaborative culture. The practice played such a significant part in the organisation that remote working or working from home was, unlike in the other two case organisations, unusual. A developer explained how new hires were introduced to the method:

So every new starter gets the Extreme programming book [meaning Extreme Programming Explained: Embrace Change by Anders and Beck, 2004, 2nd edition]. And which obviously is a form of Agile, it explains all the principles that we follow here… But yeah you get it on day one, the Extreme programming book. And as part of your probation period and your check ins over the first three months, one of the things you need to check off the list is that I've read the book and bring it up ... discussions. And that's a good, even though we don't follow it to the tee, it is good introduction on how we do things at Extreme Inc.

The next section will further explain how the pair programming worked and why it was such a pronounced matter at Extreme Inc.

4.4.1. Pair Programming: Four Eyes on the Code

Pair programming was seen as the primary way of distributing the know-how and tacit knowledge between the members of the organisation. Pairs were swapped after a few days and new pairs got to share their insights. The testers and the designers were partial to this knowledge sharing. When their work was suitable for pairing, they were paired up with the developers; at other times, they worked alone. The practice of pairing required that the physical Agile walls and the office space were organised in a way that enabled the teams to easily sit next to each other. One of the leads described his team's behaviour:

There's a hum of activity. They're talking all the time so if you sit beside a pod where there's a lot of people who are pair programming, who are talking, they are talking all the time. Not all the time, but they are talking and discussing their approach, discussing their implementation. Sometimes they are working a little bit independently because ... for whatever reason, but it's a very collaborative, communicative, noisy, experience, but it's good. I like that.

At Extreme Inc., software developers worked in pairs for the majority of their time. The tasks, written on sticky notes, which were allocated to each pair, were attached to the Agile walls. Each team had its own wall. A delivery lead explained the development process:

We will start a piece of work. We'll pick up a task off the board. You pair up and each pair has two monitors, two keyboard, two mice, but it's one computer…. And from the wall you have picked up a task that you want to work on with your pair. As a pair, we talk through the task, how are we going to implement this. We talk through the test cases that we should consider. Some people like to do what they call ping pong: so one person will write the test, the other person will actually write the code that will make that test pass. Some people like to do everything together. They both write the code and they both write the test, and then they just share between each other who is controlling the keyboard at which particular time.

Developers enjoyed pair programming. Interviewees who were practicing or had practiced pair programming before described it as a highly collaborative activity, even exhausting for the first few months. But once the new developers had gotten used to the pairing, the pair programming was seen as the key to high-quality code and effective problem solving. A developer explained to me why he thought that pair programming was an effective way to develop and maintain the high quality of the code:

You pick up your silly mistakes, you're less likely to get blocked on something trivial. It's a good chance the other person will know what to do. It's those times when you yourself know, would otherwise know what the problem is, but you get in that brain space where you just don't see it, you're just focusing on the wrong spots. So you start to build up productivity through there.

The testers and the designers had a modified way of pairing. Sometimes a tester or designer was paired with one person; sometimes they were added as a third person. One of the developers explained how testers and designers were incorporated into the paired work:

The way of working with the designers, it is a bit different – the same happens with testers, but it's not like pairing with developers, where the pairing happens full time. But rather than them sending you an email, saying 'I want this button moved ten pixels and this other button moved ten pixels, so that they align' and ... stuff, and then you send back and it's still not right. They just sit with us while we're doing the UI changes. That way we are lot more efficient to come to the agreement upon result, rather than sending things back and over. So similar thing happens with testers. So we pair a tester and developer, we finish a story that we're working on and we have embedded testers to all the teams.

One of the ways the developers and the designers interacted was nicknamed 'rubber-ducking'. In this way of working, the developer would describe the way the code was constructed and explain the thinking out loud, as if they would be explaining their thinking to a rubber duck. The technique, as one developer told me, was sometimes used with actual rubber ducks too, in order to help clarifying the thought process that go into the coding. Modified versions of pair programming were extended to the DevOps internal and external collaboration as well. One of the members of the DevOps team explained the process:

So we don't exactly pair program. We pair with changes… Any change in production requires it to be paired. So it is two people from the operations ... implement a change… So either they, sometimes they will sit together and discuss and think about the actual problem that they're trying to solve ... and then implementation.

The tribes and the DevOps team formed the majority of the employees at Extreme Inc. In addition to these technical employees, there were other supporting teams. The business stakeholders comprise a variety of other members of the organisation, such as user experience designers, sales and marketing experts, risk managers, and team leads who participated less in hands on development and more in the planning of the products and processes. These stakeholders were not directly involved in the pair programming activities but worked close to the development teams. The organisational culture which valued physical presence, stemming from the pair programming activities, was seen as an enabler of collaboration with the others stakeholders as well.

For example, the head of internal auditing outlined that their physical presence in the premises, just downstairs from the development teams, helped them in being tightly involved in all development processes:

I don't tell the technical teams how to implement control processes. I just go, 'Basically, this is why we need to do it. This is the principle that we are trying to protect or safeguard against. Do whatever you need to do.' Later, I'll come back and check that the feature is working, how you've intended it to work. And vice versa. They can always come see me downstairs, if they want to implement a new process, or they come up with a new concept. They will ask me

about the feature: 'We are thinking about looking at this. Do you have any tips? Do you have any advice? Do you have any requirements?'

The pair programming method determined how the other Agile methods were applied in order to support it. The next sections will describe how Agile walls and meeting practices were conducted with regards to pair programming.

4.4.2. The Agile Walls: Visualising Everything

Agile walls are literal, physical walls, split in lanes or columns and dotted with sticky notes, prototype drawings and avatar pictures of the team member. In addition to the physical walls, there were also virtual replicas of walls, containing the same information but created in software tools specifically designed for Agile task management. Both types of wall – physical and virtual – were important Agile artefacts for Extreme Inc. The open-plan office space was filled with swim lanes and sticky notes surrounding the collocated teams. Each team had their walls, relevant to their work, close to their physical desk spaces. Physical walls were supplemented with whiteboards, as in many cases the wall space next to the teams was already exhausted.

The walls catered for different information needs. Extreme Inc. used a few distinct versions of the Agile walls, such as Kanban walls that visualised workflows (Anderson 2010), story-mapping walls that were used to describe the product requirements backlog (Schwaber 2004), and other walls that mapped risks and relationships between the product development and higher-level product roadmaps. A product manager detailed the application of the different walls:

We use walls to present different frameworks: hypothesis board, impact map or just general Lean or Kanban canvases or customer value proposition canvases. And those are just the walls I use...The developers use Post-it notes all the time, for everything... Walls are everything. Everything is in a wall here.

At Extreme Inc., variations of walls existed for all the different practices enacted at the organisation: daily stand-ups, retrospective meetings or iteration-planning meetings. A common way to organise a wall was to divide the wall into swim lanes representing different stages of the development. The product requirements were written on sticky notes. The requirements were written in the user story format (i.e. what benefits a user was gaining from the requirement). These sticky notes with the user stories were called story cards. A test lead described the variety of the walls:

So each of the is boards arranged differently. They have different categories, but that's not the word I'm thinking of. Some boards will have 'done', some will have' completed', some will have 'finished', some will have 'what if'. Some chunk it up into horizontal lanes, some have vertical... But it works for the team. And that's the important thing. You'll see a lot of Lego upstairs. So a lot of avatars to present who is working on what. We use a lot of Lego. Superheroes seem to be a favourite avatar.

The freedom that was given to the teams when structuring their Agile practices and walls meant that the walls were not uniform or static. Some teams had walls that were following the principles of Kanban; some teams preferred their walls with fewer prescribed restrictions. The organisation encouraged teams to decide what format was best to facilitate their work, instead of dictating a uniform approach.

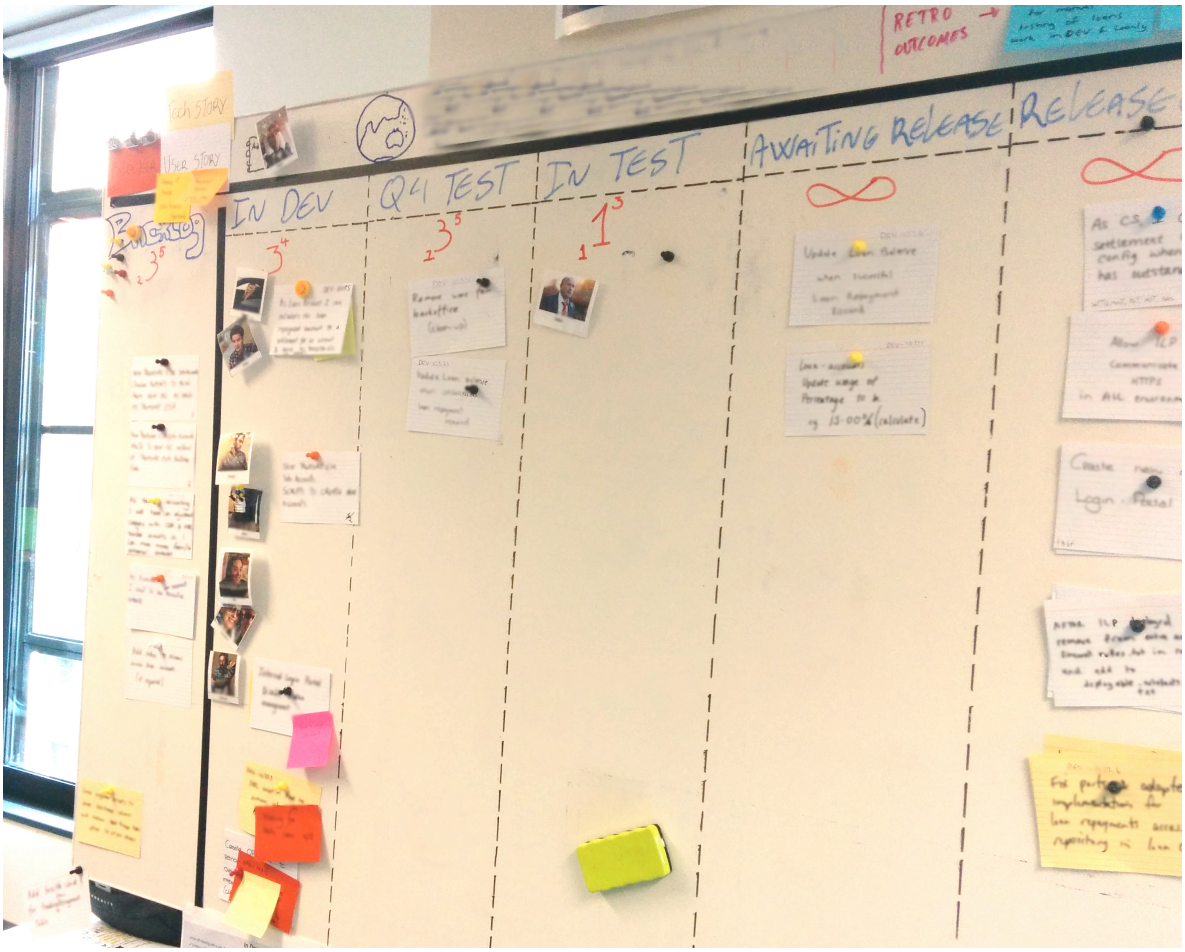

Figure 4. Example of an Extreme Inc. Agile Wall

Figure 4 illustrates one of the many Agile walls at Extreme Inc. One can see the tasks written on sticky notes and swim lanes that indicate the status of each task, beginning from the backlog, where stories that are not currently being worked on are stored. There are photos of the developers to indicate who is working on which tasks.

The information on the walls was partially replicated in virtual wall tools, but the physical instance holds the most recent and up to date information. The virtual wall version was a mandatory practice; the external stakeholders, such as other software companies, as well as regulatory authorities of the financial sector, required more product documentation than what the physical walls could provide. An example of a virtual wall ticket log is presented in ***Figure 5***. The tasks on the cards on the walls, though informative for the teams, were physically located in the office and lacked shareability. A technology leader explained:

We only really started using [a virtual wall tool] recently. The reason was the need of the audit trail. Some teams use it a lot more than others; the developers are not heavy users of [virtual wall tool]. We use it because we have to, not really because we want to, we much prefer the whiteboards and that kind of stuff.

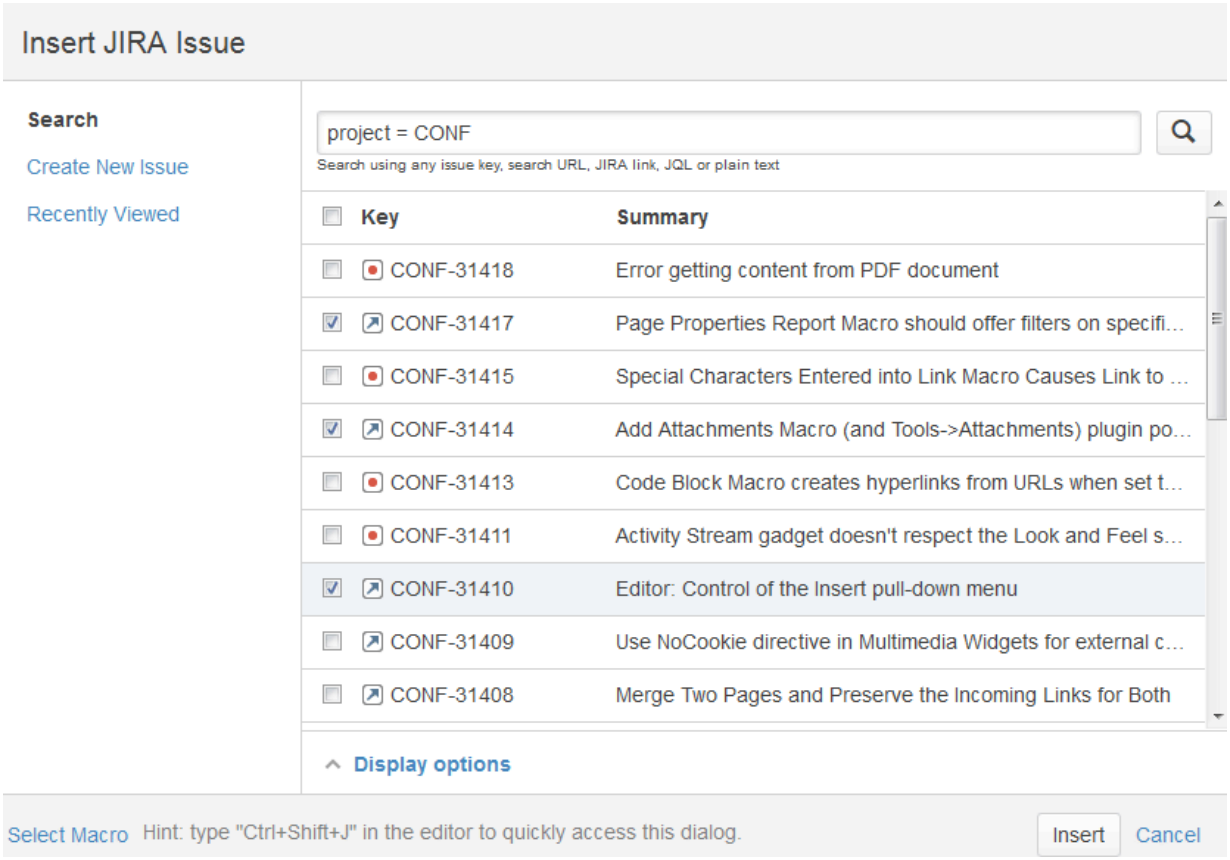

Figure 5. An Example of Virtual Wall Tickets (source: https://confluence.atlassian.com)

For the developers, the virtual walls were extra work, but for the testers and DevOps, the virtual story cards were invaluable sources of information. Interviewees disagreed with the level of detail that the virtual cards should have and how much emphasis should be given to filling the cards, but everyone acknowledged that the virtual tools were an important part of the process. Nevertheless, face-to-face communication was preferred over any virtual tools. The next section discusses how face-to-face communication was facilitated at Extreme Inc.

4.5. Agile Practices and Objects: All About Communication

Even though pair programming was the main method of collaboration, there were other avenues of collaboration. In order to engage all relevant stakeholders, Extreme Inc. had adopted recurring meetings such as daily stand-ups, fortnightly iteration-planning meetings and monthly retrospective meetings, as prescribed by the Scrum method (Schwaber 2004). As already stated in the literature review, Extreme Programming focused more on the development guidelines, whereas Scrum is a managerial framework. These two methods are not mutually exclusive and they are commonly used to supplement each other.

4.5.1. Meeting practices

Daily stand-up meetings were the main enabler of communication between the developers and the other stakeholders, such as product managers, user experience designers and testers. The meetings were conducted with the team actually standing up next to the wall created by the team members themselves to facilitate that specific stand-up. The focus of the daily stand-ups was to discuss the work of that single day. The walls were dotted with sticky notes describing the requirements that the team was currently working on and during the practice these sticky notes were discussed and moved around the wall to indicate the requirement status for the ongoing iteration. The stand-up meetings next to the wall were conducted at Extreme Inc., as the name suggests, standing up. The reasoning behind this is that meetings are shorter when the participants cannot get too comfortable (Stray et al. 2016). The informants told us, albeit in a joking tone, that Extreme Inc. was the main customer of the Post-it notes with extra strong glue.

Higher-level planning and discussion was part of the recurring iteration planning. Iteration-planning meetings charted what the work was to be for the next two weeks. These meetings were again conducted beside a wall that held the sticky notes where the potential requirements were described. Teams also held regular but less frequent retrospective meetings. In these meetings, the focus was on issues that needed improvement. The retrospective meetings had their own walls where sticky notes described discussion points rather than requirements. Business stakeholders regularly took part in the retrospective meetings and the iteration-planning meetings, especially if new requirements or processes were being discussed, but they were occasionally present in daily stand-ups also.

The meetings varied in their levels of abstraction. A delivery lead described this arrangement:

I split my time between my team and my tribe. Normally I have a stand-up between 9 and 10 to make sure that my teams are able to work during that day and during the next day. I organise the [team wall]. And I need to do the same thing for the tribe as well. Make sure that on the higher level there are features that the teams are working on.

Higher-level requirements on walls were informative to the wider stakeholder group, who participated in the meetings where long-term planning was discussed. The more detailed information was accessible to anyone, as all the walls were located openly around the office space, but served the more specific needs of the daily operations of the teams.

New practitioners introduced novel practices or modified the existing practices. A technical lead recalled:

When I joined, the iteration planning was done in long meetings…we were going through all the details in all the stories. The [iteration planning] could go on for hours. Teams have now stopped doing that …We changed the meeting and now we have ad hoc planning and design… whenever we need to flesh out the details of something, that just happens as we go. Instead we always have an hour and a half iteration planning, which is more like prioritisation and estimation.

In addition to the outside influence, the changes to the practices and the objects also emerged within the organisation. Retrospective meetings, where practice improvements are discussed, are one of the integral practices of Agile development (Derby et al. 2006), By applying the practice of retrospective meetings, the organisation is

guaranteed to improve its practices over time. Managers at Extreme Inc. were very adamant about the importance of the retrospective meetings and many improvements had been implemented over the years.

4.5.2. Documentation, Design and Product Development Environments

Extreme Inc. had a company-wide internal wiki system that was used to store miscellaneous documentation. The business manager detailed what was stored in the wiki pages:

Everything from internal policies to social clubs and you can create a space so that's basically like the whole like, on a hard drive it would be like the lending folder. And then we have everything from processes to presentations and outstanding questions.

The internal wiki pages were also the place for the designers to store their sketches and other data. The designers created both wireframes and visual designs of the user interfaces. The wireframes that were used by Extreme Inc. were prototypes that described the structure and flow between the different functionalities of the product, but were intentionally an incomplete, black, white and grey version of the product. The wireframes were never intended to be a complete representation of the end product and even a simple drawing on a piece of paper or whiteboard could have been used as a wireframe. The visual designs differed from the wireframes in their purpose. The visual designs were a collection of examples that showed the visual 'look and feel' of the products. The visual designs were often accompanied with styling guidelines, font suggestions and other details that developers apply in order to create beautiful, trendy or functional products. Neither wireframes nor visual designs are comprehensive representations of the product by themselves, but together these two artefacts present a prototype version of the vision of the final product with both functionality and visual elements. The designers mentioned that they had also created a few separate prototypes for the product managers, so that the product managers could demonstrate these prototypes to their end user contacts and get very early feedback on the ideas.

However, the products that were already out in the market were the main source of feedback from the end users. The products were constantly updated and redesigned after the feedback. New features were added to the products on a regular basis, and designers organised sessions where all the internal stakeholders could have their say on these new features. One of the technical leads explained how new designs were created and how prototypes and mock-ups helped to elicit stakeholder feedback:

Now that we have a design team… before they start, they've conducted a set of product inceptions before they get all the different stakeholders together for – depending on how big the feature is, the work could be a day, could be a week. Part of that is that sometimes in the walk-through sessions, every stakeholder will get an opportunity to draw what the design should be. And this task might be just done on a piece of paper or on a whiteboard. And then the design team will go off and make that a more realistic prototype that we can use for development or use for customer feedback.

The products that were out in the market were constantly updated with new features. In Extreme Inc.'s case, the products were hosted (i.e. the code was located in servers and accessible by users) by an external organisation that ensured the security and accessibility of the systems. Nevertheless, DevOps had access to these environments and they ensured that the new code, which was deployed every two weeks, was not compromising the products. The

servers themselves were 'invisible': the hardware was not a concern of Extreme Inc. and only provided a platform on which to run the code but the status of the product was a different story. Product updates and the workflow of the DevOps team were visualised in different dashboards that were used to display and monitor the 'health' of the systems currently running on the environments. The DevOps managers showed me the screens that were spread across the DevOps team space in the office and explained the dashboard views presented on the screens:

We monitor the infrastructure that runs the application. We also monitor the application itself, for health. They [the developers] also have access to the monitoring for the application health as well…And they have access to that, so that they can see the health over application running and production. This information is displayed on those TVs.

Figure 6 shows an example of a continuous delivery dashboard. The DevOps team was able to visualise a multitude of factors, from number of defects and tests, to memory use and other issues that were concerning the usage of the products.

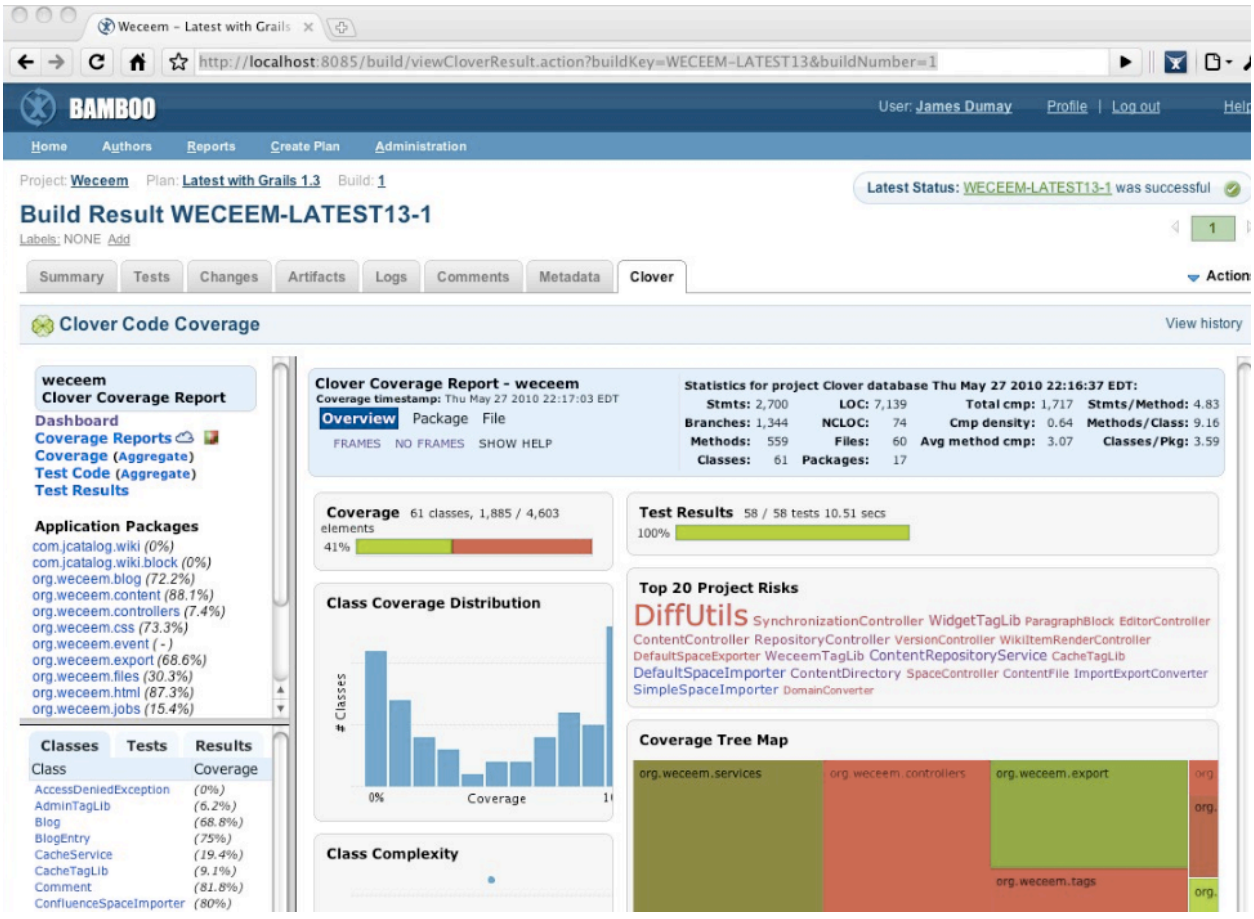

Figure 6. An Example of Continuous Delivery Dashboard (source: atlassian.com)

4.6. Discussion of the Extreme Inc. Case

Extreme Inc. is a rare example of an organisation dedicated to an Agile method from the very inception of the company. The strong organisational culture permeated all aspects of work at Extreme Inc. The office layout and hiring practices were designed to support the chosen Agile development methods. Minimal physical boundaries in the office space, team collocation and the pair programming practice facilitated the culture of face-to-face collaboration. The application of boundary objects was driven by the needs of the developers rather than that of the managers or other stakeholders, save the virtual replicas of the walls. The collaboration with the internal stakeholders was tightly knit. The flow of customer requirements from the business to the developers was secured with the team composition. Product managers and designers, who were on the front lines, engaging with the end customers, were integral parts of the teams and communication between the developers and business stakeholders was seamless. The next sections will explore in more detail the Agile philosophy, object use and stakeholder collaboration at Extreme Inc.

4.6.1. Agile – Rigorous and Uncompromising

Extreme Inc. observed Agile methods in an exceptionally rigorous way. The employees were very engaged with the practice but enjoyed significant freedom to organise their own work. One of the interviewees clarified the approach:

We try to keep things very clean, just to get the stuff done. And at the same time, not compromising ability to deliver, compromise in quality, compromise in security, any of those sense…Coaching, mentoring, leadership. We're really trying to build our people up. We ask them 'What are your goals? What do you want to achieve?'

Employees who had worked in other Agile organisations before joining Extreme Inc. often compared their current experience to their previous jobs. In these comparison stories, the Agile methods applied in the other companies rarely came close to the rigour and diligence of Extreme Inc. The methods were sometimes even completely discarded after changes in the organisations. For example, one of the test leads told a story about his old employer, who had practiced Agile methods earlier but suddenly decided against the Agile ways of working:

When we're in our development phase. And all things considered, we were not as Agile as we pretended to be. However, the testing side of the project was actually very good. Highly explorative, highly collaborative, highly context driven…. And then it went from Agile to, reverted to the old way. It is like stormtroopers came in and then Darth Vader said: 'we're going to do this. Let's go to the dark side. So that was ugly'.

A common element in these accounts was that the organisations had implemented Agile methods but the interviewees felt that something was missing. They told me that they felt that that the organisations were either not committed to the methods or lacked an understanding of the purpose and Agile values. A technical lead recalled how his previous employer compared with Extreme Inc. and how the Agile values were adopted:

But it is interesting when you look at the places that do all of these things, the [Agile] rituals, but are not really Agile. Only Agile in name. Extreme Inc. has been really keen – trying to get back to those first principles. And it was [with the other company]: 'Oh you do Scrum, you have a burn down chart and you do these investigations and things like that'. But you can see, within the burn down chart, it's not something that's designed to minimise a unit

of work and make sure that you get MVP type products to production. It's there to, so that the manager can say 'In two weeks you'll be here.

The openness of the organisational culture and the self-management of the teams were stressed in multiple interviews. The teams were free to explore and try out things as long as they were adherent to the parameters set up in the Agile Manifesto. A test lead at Extreme Inc. explained:

Here they do Extreme Programming. So they have a piece of framework, they know what they want and they'll let the team figure out how to do that. Each team, how they developed their boards and work, it is up to the team… There is a high expectation to try stuff and make sure you learn from it. So if you try something and it didn't work, we will have a [retrospective meeting] instead of 'whose fault was this?'.

The interviewees did acknowledge that the way the work was organised now was not the same 'by-the-book' view of Extreme Programming that it had been years ago, when the company was founded. Nevertheless, the values and principles were the guiding elements of the practice. The role of the organisation as one of the prominent sponsors of Agile meet-ups and their vocal advocacy of the methods but uncompromising attitude meant that Extreme Inc. had a reputation in the Agile community as one of the leading proponents of Agile in the Australian context. As the lead of sales summarised:

There was a big cultural thing, actually. Agile. Maybe you had heard about it and you thought, that's actually a really great way of doing things. Both from a engineering perspective but also from a cultural perspective. Because Agile requires people to have slightly different personalities and requires them to handle themselves in a very different way to that, which is waterfall development. One way is more social, the other way is much more individualistic, I think.

When I was comparing the perspective towards Agile methods presented in the Extreme Inc. case, the similarities to the perspective espoused in early Agile literature and literature that explores the philosophical or worldview side of Agile development are clear. The way the interviewees spoke about their Agile experiences made me understand that the employees in this organisation were striving towards implementation of Agile as a set of values and principles, as a worldview and a thinking paradigm in their daily activities. This perspective on Agile was not surprising, given that the guidebook, Extreme Programming Explained: Embrace Change (2000), that the organisation gave out to every employee and which was repeatedly mentioned as source of inspiration, was by one of the more philosophically inclined proponents of Agile, Kent Beck (see for example discussion on influence of Coyne (1996) in Beck 1999).

The adherence to Agile as a philosophical statement rather than just a managerial method was observable in all aspects of the organisation: the way the interviewees discussed the topic, the historical aspects of the organisation, and the current physical environment of the office premises. The Agile perspective had a significant influence on the way the organisation had organised the stakeholder collaboration as well as the application of the boundary objects. The next sections will detail how the collaboration was configured and how the objects were used to facilitate such rigorous Agile approach.

4.6.2. Business stakeholders – Integrated Collaboration

The Extreme Programming method was designed for software development work, but at Extreme Inc. it was not limited to the development teams. The three product development tribes and the DevOps tribe were supported by sales, marketing, finance, risk and internal auditing functions. The different nature of the work in these functions did not always lend itself to fully fledged application of pair programming or other Extreme Inc. methods, but these stakeholders had adopted the methods to the extent that was suitable for their needs. Their understanding of the methods and the culture that fostered face-to-face collaboration amongst all the members of the organisation ensured that the business stakeholders at Extreme Inc. were very tightly integrated into the daily operations.

However, the tight integration did not mean that all business stakeholders were in constant collaboration with the development teams. The organisation had devised a role of business managers, who held the responsibility of ensuring that the business stakeholders and the teams were aware of information relevant to their work. These business managers were embedded into the technical teams and acted as the conduits between the customer requirements and the technical members of the teams. The customer and end user requirements were sourced from the feedback provided by the end user as part of product support and by the business managers themselves who organised feedback collection sessions. In order to disseminate these user needs to the development teams, the business managers regularly participated in the daily stand-ups and other meetings. A developer explained what role the business managers played in the daily collaboration:

We try to work as closely as we can with product management. So, as part of developing our work, we have to include them in our reviews and stories. Which is, our task should be represented as a user needs or something that provides value to the business. So our product management will join us and talk about their need and what their priorities are. Represent our end users as well. It is bit like a negotiation between resources. Then things are prioritised, we'll give a lot of technical input to make sure that the prioritisations really stick…And occasionally every week depending on what type of project it is and how many different streams, sometimes you need more prioritisation.

However, the role of the product managers was not to bridge boundaries between the business stakeholders and the development team. This responsibility was equally distributed amongst the whole organisation. The development teams were responsible for their own work as part of self-organising and committed teams. When a situation arose where they were required to obtain information from the business stakeholders, the responsibility to collaborate was not transferrable to the product managers, but the developers had to work with the business stakeholders themselves. Similarly, if the business stakeholders were to collaborate with the development, the initiation of the collaboration came from the stakeholders. The organisation had fostered a culture and designed premises that made such collaboration easy. The business stakeholders, such as marketing or sales, were located on the same premises and used visualisation tools such as sales pipelines on walls that resembled the Kanban walls of the developers. The sales manager explained how pairing and walls had been applied in the sales department:

They effectively pair up with ... who's responsible for maintaining and managing their pipeline. At least in the beginning. So they have weekly meetings around how their [sales] pipes are going or not going. So we can quickly

see if someone's like falling behind and try to understand why that is so and redress the problem. We have monthly targets that we set for ourselves.

Figure 7 illustrates the very flat and open structure of Extreme Inc. collaboration. The product tribes and the business stakeholders were tightly knit, sharing understanding of the Agile values. Business stakeholders praised the developers for their proactivity when it came to requirements clarification or other collaboration. The arrows illustrate the collaboration channels between the different stakeholders.

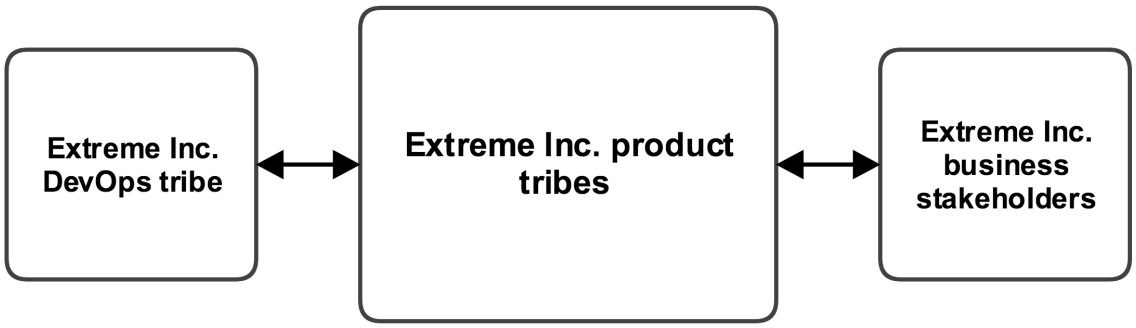

Figure 7. Extreme Inc. Stakeholders

The development teams were very accommodating towards the product managers, testers or any other stakeholders. For example, in one of the teams, a product manager felt that the information on the wall was not clear enough. A delivery lead described what happened at the time:

We changed our board to more like Kanban style. That was mainly after a request from our product manager, he had been reading a lot about Kanban and he felt he could get a better picture if he had a Kanban board. He could see what's in progress, where do you need to put stories in.

In a similar example, a test lead wanted to improve the collaboration capabilities of the objects. He explained:

We've done quite a bit of work in process improvement. So we redesigned our whole Kanban wall. Previously it was a very static wall that didn't tell anything except 'we are working on this' and that was it. There was nothing more. So we redesigned it so that we have pipeline visibility from sales…It is extremely useful when we are talking to the sales team and when we are talking to the management. We can actually tell a story.

The organisation was open to new initiatives that would improve the integration even further. In the established and strong organisational culture of Extreme Inc., these improvements were often small and gradual, but had a significant impact. One of the examples mentioned was the physical location of the product managers. Previously, the managers had been located in one table amongst each other but in recent months, they were moved to sit next to their teams. Interviewees said that this relocation had made communication between the developers and product managers even more effortless.

4.6.3. Artefacts – Supporting Presence

At Extreme Inc., the role of artefacts was both emphasised and downplayed. The pair programming practice, which was the cornerstone of all communication in the organisation, was the focal point of most interviews conducted with the informants and the defining element of the organisation. Nevertheless, the application of artefacts, especially the Agile walls in both physical and virtual format, was seen as an important part of development.

The artefacts had several functions. First, they were applied as a way to mitigate the sense of exclusion created by the method of paired work (Barrett et al. 2012). Even though pair programming was very effectively facilitating collaboration between the developers, the members of the organisation who did not partake in pairing were excluded from the most prominent communication channel. To mitigate exclusion, Extreme Inc. dedicated significant attention to the application of Agile walls. Both physical and virtual walls played a large role in the attempt to bridge the boundary between the paired developers and the other stakeholders. The walls served as a common reference point for mutual collaboration across different stakeholder groups and organisational boundaries (Star & Griesemer 1989) and provided an easily understood, visual representation of the project activities. For the more technically adept, the walls tell a detailed story of the product, where a product manager can see how many tasks are being worked on and what the next steps are. Managers who were not acquainted with the project details could see that the Agile practices were being observed by the teams and that there was progress being made in the work.

In addition to the physical walls, which were the most ubiquitous boundary object in the Extreme Inc. office, the organisation used a few other tools to enhance collaboration. A virtual wall tool was used to capture the same information as what was pinned to the walls, representing the walls in the virtual space. There were two main reasons cited as to why a second version of the tasks was needed when the physical walls were the main tools. First, there was a requirement from the regulators of the industry that there had to be a traceable log of the changes that were done to the products. The trace was achieved by making sure that no user story was deployed to production without the approval of a DevOps person and a signoff in the virtual wall tool. Secondly, the testers used the virtual story cards. As one of the testers explained:

And even now with Kanban, for anything to move into the done column, you shouldn't be moving unless you have a [virtual wall] ticket and that's actually made of a benefit for a tester. So if they need to look up why are we doing this or if they come across some changes, if they have the [virtual wall] tickets number there, they tend to looks up whatever information is there. And at the end of the day, a sticky will get lost and get thrown out and or do nothing with it.

The last category of artefacts that had a significant role at Extreme Inc. was the product development and production environments. By environments, the members of the Extreme Inc. meant the software development infrastructure, the domain of the DevOps team. The infrastructure encompassed both the virtual environments where the product code ran and different scripts and tools that enabled monitoring of the 'health' and the performance of the products in these environments. Due to restrictions and regulations, the DevOps team acted as both a gatekeeper and enabler of code changes and had an active role in product development. The developers had a restricted access to the 'live' product environments, which meant that they could only access certain environments with a DevOps person as their pair.

The significance of the environments was combined with a slight disconnect between the developers and DevOps. Even though there were pairing and other activities to maintain the communication, some interviewees named DevOps as their blind spot, the topic they knew the least about. One of the DevOps managers contemplated the current role of the environments and how their work could be communicated better to the other members of the organisation:

If we did more TedTalks on infrastructure more frequently, I think it would give them a better picture. It would benefit them in terms of being able to understand the infrastructure better when they do write their applications, I think.

All artefacts served specific purposes. The walls, both physical and virtual, ensured that the expectations of all the members of organisation were aligned. The Post-it notes describing the work items – user stories – were constantly visible for anyone interested in the work that the teams were conducting. The business stakeholders were welcome to participate in the meetings and to approach the development teams whenever they felt that a requirement might need clarification or elaboration. In addition, the virtual walls ensured that the organisation was fulfilling the auditory requirements mandated by the regulatory authorities.

The Agile activities, the boundary and other objects that were linked to the activities, and the purposes of the objects are summarised in ***Table 16***.

Agile Activities	Supporting Artefacts	Purpose of Artefacts
Daily meetings Pair programming Retrospective meetings Showcases Other meetings	The physical Agile walls Virtual walls	Alignment of expectations Abiding to regulatory needs
Prototyping	Wireframes/prototypes Visual design The products	Identification of different aspects of the problem Exploration of the shared goals
Continuous delivery Deployment process	Environments	Facilitating development Abiding to regulatory needs

Table 16. Summary: Extreme Inc. Agile Methods and Artefacts

The prototypes, visual designs and the products themselves provided a baseline for discussions of the different aspects of the issues that the product development might have at any point in time. They served as a visual, concrete artefact on which to base exploration of the shared goals between the teams, product managers and the other stakeholders. Lastly, the environments as a development artefact ensured that the regulatory requirements were met from a security and auditory perspective and that the development was fluent and undisrupted.

4.6.4. Conclusion

My first case study, of Extreme Inc., is a study of organisation where Agile software development was deeply rooted in all aspects of the daily work. Extreme Inc. adheres to a rigorous and strident approach towards Agile software development. It resembles the views of the authors who see Agile development as a philosophy and a set of values and principles to abide by. The organisation strives to live up to the guidelines detailed in the Extreme Programming method, described by Beck (2000), and has applied the methods how they were originally intended, with minimal modification.

Nevertheless, Extreme Inc. is by no means the only organisation that has applied Extreme Programming and has a lengthy history of Agile development. There are accounts in literature of similar organisations where Extreme Programming is applied throughout all work (for example Sharp et al. 2009). However, I argue that the case of Extreme Inc. provides an unique account for several reasons: Extreme Inc. is not a FinTech start-up, but an established and reputable mid-sized organisation which has applied Agile methods in the financial industry almost since the inception of Agile methods. It has managed to foster its Agile perspective and collaborative culture in this environment, despite regulations and other financial authority demands that put constraints on how the software can be developed. Extreme Inc. has succeeded in navigating through this industry and remains a rare example of an organisation that has managed to maintain the integrity of its Agile perspective during organisational growth periods as well.

Yet, in order to understand the Agile environment better, a single study does not help with creation of a truly holistic view. As seen from the literature review, the perspectives towards Agile exist in a spectrum. Extreme Inc. is an organisation that conducts internal product development, but this is not the only model of software development. The work is often done in collaboration between two or more organisations and the relationships between the organisations vary from customer-vendor relationship to different partnerships. The second case study will present a case where two Agile development organisations collaborate in a product development project.

5. Case Study Two: Virtually Agile

"We've got, every living organism has a social network, the monkey-sphere... Now, how do you jump in? You create a personal bond… And that's the important thing. That I'm there all the time and they need to know that I'm there, all the time. And that's the purpose of communication."

–Product Owner, Escapade

5.1. Introduction

The second case study examines a product development project, where Agile methods were utilised to ensure mutual understanding and customer satisfaction. Two organisations, both practicing Agile development, were involved: a customer, an online retailer called Escapade (pseudonym); and a vendor, the Agile development consultancy, Carmine (pseudonym). The project was described as straightforward product development that involved no ground-breaking technologies or product concept complexities. The product being developed was a new version of an older eCommerce platform, commissioned by Escapade and designed and developed by Carmine. The second case study illustrates a mature and frictionless application of Agile methods for collaboration and clear, shared goals in a virtual organisation setting.

The second case study chapter follows the structure of the first. However, there are some differences when it comes to the context of the case. The second case study consists of two organisations, which are both introduced before I present how I collected the data. After the introductions, I shift the focus from the two organisations to the project and discuss how the perspective towards Agile by members of both organisations formed the perspective of the project as well. This section is followed by a detailed account of the application of the Agile methods and artefacts. This description intertwines the project events, artefacts created and applied and the stakeholder collaboration. The section concludes similarly to the previous case study, with an exploration of the key elements of the case which answer my research questions on behalf of this case study: the perspective towards Agile, the stakeholder collaboration configuration and the application of boundary objects.

5.2. Case Organisations

The second case study is a study of Agile software development in a virtual environment. The focus of the study is not on the organisations themselves but on the project these two organisations conducted together. Nevertheless, the differences and similarities between these two organisations with different collaboration methods and experience of Agile development influenced the project collaboration and outcomes. In this section, I will describe the project and the two organisations involved, Escapade and Carmine.

5.2.1. The Customer: Escapade

Escapade is an online retailer, located in Sydney, Australia. Escapade's main revenue comes from its consumer sales, but in addition to the business-to-consumer transactions, Escapade also has business-to-business clients.

Escapade's website is its primary sales outlet, created and maintained by an internal development team, but the business-to-business offering is a service rather than a product. It is an internally developed eCommerce platform, which provides business clients with employee management features.

Founded in the early 2000s, Escapade has grown to be a company of approximately eighty employees and has over 2 million Australian dollars in online sales transactions. The company consists of an IT development team, customer service department, business-to-business client department, marketing department, finance department and management team. Most members of the company work from the company office building in Sydney. The company has an internal chat tool and capabilities for online meetings, but physical presence was the prevalent method of communication.

Escapade is known for its customer service excellence and its online store. A back-end eCommerce portal has a significant role in providing this customer service. Escapade's internal software development team is responsible for the development and maintenance of the online store and other products. For the last few years the development team of Escapade had adopted a combination of the Scrum (Schwaber 2004) and Kanban methods (Anderson 2010). The use of Agile methods in the development team has dispersed the awareness of the Agile methods throughout the employees of the organisation. Escapade business functions teams such marketing, sales and customer service teams were knowledgeable when it came to Agile development, even if the methods were not directly applied in their daily work.

The project Escapade had embarked upon when I started my study was conducted with Agile methods as well. The goal of the project was to completely redesign and redevelop the eCommerce platform product that Escapade was marketing to its business-to-business customers. The old product was based on out-dated technology and the usability of the product was not up to modern standards. Escapade's long-term goal was to expand the business-to-business market, and in order to facilitate this expansion a new product development project was necessary. The main issue with the project was resourcing that had to be resolved: the company's internal development team was working on a different project and was thus unavailable for reworking the eCommerce product. Escapade decided that an outside vendor was needed, as the internal resources were not sufficient. After negotiations with a few companies, Escapade's management team selected Carmine Co. as their development vendor.

Even though the project would be outsourced to an external vendor, Escapade's managers acknowledged that the project would also require significant allocations of time from the Escapade employees. A product owner was appointed to the project, as well as two project sponsors, who would be responsible for the project requirements and schedule monitoring. The Escapade chief technology officer (CTO) was tasked with managing the costs and high-level communications between the two organisations, whereas the product owner would be responsible for day-to-day operations. The product owner and project sponsor were also responsible for the internal communications about the project and they selected a team of employees from different departments who were familiar with the older product that was being replaced and who could help with the design and testing.

5.2.2. The Consultancy: Carmine

Carmine is a software development consultancy that provides user experience design services, prototyping of websites, and application and website development, as well as maintenance and support. Carmine's primary offerings are software development projects carried out with an open source development framework, Ruby on Rails. The selection of the Ruby on Rails framework ties Carmine into the open source community (for software ecosystem discussion on Ruby on Rails, see for example Kabbedijk & Jansen 2011) and shapes Carmine's brand as a software developer consultancy (Fitzgerald et al. 2006). Ruby on Rails was and still is, at the point of writing this thesis, a trendy framework used by companies such as AirBnB and eBay and the community is active and engaged (Hartl 2016). Selecting Ruby on Rails signals to potential customers that Carmine is modern, up-to-date with latest technologies and focused on visually impressive products or websites.

The company consists of over 30 people, with a mixture of managers, designers and developers. The employees of Carmine either work from home or are located on the customer's premises. The company has no physical offices; all company interactions are conducted online via chat tools and VoIP (Voice over IP, e.g. Skype) calls.

Carmine had allocated a project manager, two developers, a user experience designer and a visual designer for the Escapade project. The Carmine project manager explained that even though there were no project manager roles prescribed in their chosen Agile methods, the project manager was needed for this project to monitor costs and ensure communication between the management of the two companies. The development team and the designers were self-sufficient when it came to the technical details of the work and product owner communication. The more senior of the two developers was also filling the role of Scrum Master in the development team meetings. The development team engaged in the project were working from several different locations. The user interface designer made several visits to the Escapade office, but worked mostly from home. The visual designer and the project manager were located in different states and one of the two developers was located in Canada for a lengthy period of time during the development. The other developer was located in Sydney and worked occasionally in the Escapade office. The solution for effective communication in this mixture of locations and time zones is discussed in a further section.

5.3. Escapade and Carmine Interviews

I came to know about the project via mutual acquaintances, who introduced me to the product owner from Escapade. The project had begun several months earlier and the work was well under way at the time I conducted my first interviews. During the time I was conducting the interviews, the product was released for a limited group of testers. These testers were a carefully selected team of Escapade employees, whose feedback was used to finalise the first version of the product, released a few months later. I concluded the first round of interviews with a small presentation of my initial findings with the product owner and the CTO. After the first round, the project progressed into a 'minimum viable product' stage, where the system was released to a small number of external users.

The second round of interviews was conducted between April and May 2015, a few months after the initial release of the new system. Three additional interviews were conducted with an emphasis on clarification of the findings, the new project events that had occurred since and verifying our preliminary theoretical ideas (Klein & Myers 1999).

The product owner was interviewed a third time and this interview led me to two additional stakeholders, who were members of the Escapade development team. The Escapade internal development team became a major stakeholder after the project was completed from Carmine's perspective and handed over to Escapade for further development and maintenance. ***Table 17*** summarises the timing of the interviews, the informants from the two companies and which themes were discussed during the interviews.

Organisation	Informants	Themes Discussed
Interview Round 1, April to June 2014 *indicates that the person was interviewed twice		
Escapade	Manager A*, CTO*, Two project sponsors Two testers Administrative user	Project events, organisation of communications, role of involved stakeholders, communication tools and mediating artefacts used, Agile practices enacted
Carmine	Manager, UX designer Developer	Systems development process, organisation of communications, role of development team members, communication tools and mediating artefacts used, Agile practices enacted
Interview Round 2, April to May 2015, *indicates that the person was interviewed third time		
Escapade	Manager A* Manager C Internal system user	Communication tools and mediating artefacts used, role of involved stakeholders, new project events since round 1, verification of preliminary theoretical ideas

Table 17. The Escaped and Carmine Interviews

In total, I conducted twelve interviews with seven different informants from Escapade and three Carmine informants, thus capturing the majority of the Carmine stakeholders. The informants represented a good mix of product users and testers, project sponsors, developers and project managers.

The interview questions of both rounds were open-ended and non-leading (Walsham 1995), with a mirroring technique (Myers & Newman 2007) applied when using project-specific terminology. Interview questions were adjusted based on the role of the informant, and the majority of the interviews with Escapade informants were held at the premises of the company or nearby cafés to create a relaxed and familiar environment (Myers & Newman 2007). The informants from Carmine lacked office premises, so I opted to meet two out of the three informants in cafés. The third interview was held over Skype due to the remote location of the informant. All of the interviews were digitally recorded and transcribed (Walsham 2006), including the Skype interview.

The interviews with members of both organisations gave me an understanding of how the project was unfolding and what the key events were that drove the project forward. In the next section, I will describe the project and the tools that were used to ensure collaboration between the two organisations.

5.4. Agile Approach: Making Agile Work Virtually

Traditional approaches, such as the waterfall method, were never considered for this project. Escapade stakeholders were adamant on utilising Agile. The Escapade management and project sponsors were all critical of traditional project management methods and eager to experiment with Agile methods. A project sponsor from Escapade explained their reasoning:

We didn't want to sit there, waiting for nine months for them to deliver an end product, we wanted something now. And to see it working and start getting paid users on it as quickly as possible so it starts to pay for itself. And that was, you know, what we saw the Lean framework gave us, that ability to start getting your software paid for sooner rather than later.

Other members of Escapade had even stronger preferences towards using Agile. The DevOps manager, whose responsibility was to eventually take over the product from Carmine, told me his views:

Agile methodology is great. Really, irreplaceable. Assuming you've actually got the stakeholders, developers, everyone realises that when you start a piece of work, that you've actually got to finish a certain minimum amount of that work.

Escapade began the project by crafting a project business case, which detailed the very high-level ideas and schedule outlines. After the business case was approved by the company board, Escapade started the search for a suitable vendor. The selection of Carmine was based on its reputation and references, as well as the Agile methods Carmine was known for. The CTO of Escapade highlighted Agile and quality as the criteria:

I was trying to find a company that would be aligned with Escapade. One of the challenges is that there are many 'body shops' consultancies that are just in-and-out. They've got no sense of quality and they've got no sense on Agile, our selection criteria. So we ended up picking up Carmine as the chosen vendor.

The interviews with management indicated that there was a clear drive for succeeding with the Agile processes. Escapade's managers had trialled the methods with their previous employers, but for both the product owner and the CTO, this was the first larger Agile project at Escapade. Both had strong opinions of how they wanted the project to run and what they thought about Agile. When discussing the vendor selection with the Escapade CTO, he explained:

I know when companies tell me that 'we do Agile'. It's a buzzword and I actually hate that word nowadays! I would rather say that I do Lean, I do Kanban and we have a Scrum board and we do iterations! I've got a lot of experience in Agile, eight years, so for me it was easy to, when I was speaking to vendors, it was easy for me to pick out the ones that were um, how should I put it, bullshitting me. Just selling me 'I know we're Agile' so I would talk to them 'describe your way of Agile' and their being Agile was to move faster, which is not Agile. I obviously know, based on my experience, that when they say 'oh, we can change things whenever you want', that is not Agile they are talking about.

Carmine's employees had similar views when it came to project management methods. For example, the Carmine manager explained that the relationship required a lot of trust from both sides and described his distaste for traditional projects:

I really don't believe that a fixed price development, fixed price, fixed scope, is a very good way to run a project at all. Because otherwise; from the very beginning, you do that, you are going to be fighting the customer. You are not working together, you are working against each other right from the get go. So, there is a lot of benefits from doing it with the Agile method but convincing a customer that it's the way to go is ah, can be tricky.

Crafting an Agile contract was one of the ways the two companies ensured an Agile engagement. Managers in both companies were aware of the challenges of Agile contracting and they told me that they had put a lot of thought into the contract, making sure that the contract was designed to accommodate the flexibility inherent in Agile processes. The scope of the project was kept deliberately ambiguous. While the contract outlined the schedule and budget, it allowed the customer to terminate the project at any time. The relationship, as the Carmine manager had explained, was based on the trust and the goodwill Carmine had acquired from other projects by delivering what was promised. The CTO of Escapade was very pleased with the contract. He told me of an earlier discussion with Carmine:

I've said to [Carmine CEO]: for the benefit of the both companies, we want to make sure we don't lock you into something and you don't lock yourselves, that is the best way… We did write that we would aim to have a certain amount done in a certain period but it was very loosely written… So there was the expectation and goals set but it wasn't set in stone.

Besides the Agile contract, the project team had to agree on the Agile methods. Carmine was proficient in Scrum practices and Escapade agreed with their chosen method. Internally, Escapade had introduced Kanban practices such as Kanban walls to support their Scrum practices, but the wall was sparsely used and only as an internal communication tool. Scrum was the main method and the organisations agreed that the language of Scrum was to be used to describe inter-organisational practices, with terms such as 'sprints', 'user stories' and 'daily scrums'. Interviewees from Carmine told me that they were accustomed to using the tools the customers were most comfortable with. In the Escapade project's case, it only meant switching the backlog tool from their preferred one, whereas with other customers, more compromises had to be made. The only no-compromise element was the customer involvement. The Carmine manager explained:

So, if we found a client who would not be contributing as much as we like, then, that's when a warning bell starts ringing and we just, we don't start a project without them committing to working closely with us. Because we know that it is such an important part of a process.

Escapade customers were driven for project success and willing to communicate. Although there were occasional visits to the Escapade office, the development team of Carmine was a virtual team: each member of the team had connectivity via the Internet and collaborated with their colleagues and customers via online collaboration tools. In addition to the usual VoIP conference calls, the product owner and the CTO of Escapade had access to a few chat rooms in a chat tool used by Carmine and were eager to communicate with the developers and the designers via this chat. The more traditional email communication was supplemented and partially replaced with instant messaging, chat rooms and VoIP calls. The Carmine project manager described:

We've been experimenting with a lot of tools for video chats as well. Um, we've had a bit of success with a new tool. We've been using that for our daily stand ups… And that will allow you to do video chat with voice. Ah, and instant messaging quite easily. You just have to send someone the link and they can join up.

The project manager praised how dutifully their customers were applying the tools:

The backlog tool is a big part of our process…The client, the product owner, in this case is always inside our backlog tool where he gets to watch all the stories, what is their status.

Similarly, the Carmine developer I interviewed was also happy with the use of the other communication channel – the chat tool – that they shared with the customer:

The guy whom I'm working with, the client representative, he is on that, he is on our chat tool all day.

Interestingly, the virtuality of Carmine was never seen as a hindrance to the application of Agile methods. Research on remote work has shown that working virtually can create issues with trust and complicate the feeling of belonging (Hafermalz 2016). Software development is hard enough even when everyone is present in the same office, let alone in a fully virtual organisation like Carmine. Knowing the potential dark side of virtual organisations, I was surprised at the overall positive attitude all interviewees had towards the setup of the development organisation. The internal communication of the organisation was not taken for granted. Carmine had established practices that ensured their employees had a relationship with each other, as well as their clients. For example, a virtual meeting room in the communication tool compensated for a lack of office. The manager at Carmine explained:

We've got a room [in the chat tool], a room we call the Lounge, the Buffalo Lounge to be precise. And we use that as a virtual water cooler where we all convene and when we start the day and end the day we sing in and sing out of that room and announce our presence. It is also a place where people will share interesting new stories, funny pictures or I don't know, anything that's grabbed their attention.

The chat tools and VoIP calls were not the only means of collaboration. The project applied many Agile methods to ensure that the product would satisfy the customers' needs. The next section will describe in detail how the Agile methods were exercised.

5.5. Agile Practices and Objects: From Discovery to Delivery

Projects are commonly defined as discrete endeavours with a definitive beginning and end (PMBOK 2015) but Agile development is putting this definition to the test. Modern product development never truly ends: new versions are constantly rolled out, bugs are fixed and updates are applied to existing products (Humble & Farley 2010). Modern software products, from small applications to operating systems, evolve until they become obsolete when a replacement is developed.

The Escapade-Carmine project is no different from other continuous delivery projects: it is challenging to describe the project from the beginning to the finish when there is no finish. The second round of interviews with the Escapade informants was conducted for this reason. I was curious to hear about the latest developments and how the project continued after the main vendor engagement with Carmine had ended. New project iterations had taken place after my last interviews, but additional stakeholders – the business customers – were creating new challenges for Escapade. The allures of a simplistic, sequential waterfall model become clear when one tries to explain iterative

and incremental projects in the simplest way possible. How to describe what was happening when everything happens in parallel and in cycles?

I therefore decided that instead of a chronological description of events, I would focus on detailing the daily work, how the Agile methods were applied, what objects were used and who was involved. Nevertheless, time creeps into the narrative, so the story of daily activities is told in three parts: what happened in the two stages of engagement with Carmine (Discovery and Development) and what methods were applied after the minimum viable product (MVP) had been released.

5.5.1. Discovery: Show, Don't Tell

The goal of the discovery phase was to establish a common understanding of the product and Escapade's requirements. The user experience designer from Carmine explained:

I use it to say that someone has an idea and turning that idea into something tangible that can be estimated. That's what I'm calling the discovery phase… Getting it out of the client head, out of everyone else's... What they want. Translating that to prototype or to wireframe or a site map or anything.

During discovery, the user experience designer, the visual designer and the developers began gathering, consolidating and clarifying the requirements for the system design and functionalities. The user experience designer organised workshops with the Escapade product owner and the project sponsors, where the initial requirements were discussed. For these workshops, the user experience designer visited the Escapade stakeholders at their office. After the workshops, the designer created a set of interactive wireframes that were a prototype of the product and were tested and trialled by several stakeholders from Escapade. A manager from the Carmine organisation described this process:

Our user experience designer would create a number of interactive wireframes that she presented to the business and they could click through and play with the wireframe to get a feel of what we had in mind.

After these test sessions, the designer created a new and revised version of the wireframe. An example of one of the pages of the wireframes is presented in ***Figure 8***. The wireframe pages were mostly black and white, designed to be easily distinguishable from the visual designs and the product. Main functionalities were mocked up and users could browse different product pages in the wireframe model, but actual functionalities were not implemented.

Another part of the discovery phase was the creation of the 'look and feel' of the product. The 'look and feel' meant the design of the colours, fonts, icons and images, applied on top of the product layout. This work was the domain of the visual designer. With the help of the wireframes and the visual designs, the two designers – the user experience designer and the visual designer – captured the feedback given by Escapade stakeholders in their feedback notes. They would refer back to these notes when they were working on improvements to the wireframes and visual designs. The user experience designer explained the collaboration between the two designers:

Our visual designer took the wireframes and the developers started to work on features already by the time when the visual designer started with this project. She designed what the product should look like. We collaborated a lot.

Normally the visual designer and me, there was always a lot of back and forth. I showed her the wireframes and she told me 'Yeah, okay. I can do that' or 'That won't work that well', so if she disagreed, I changed the wireframes.

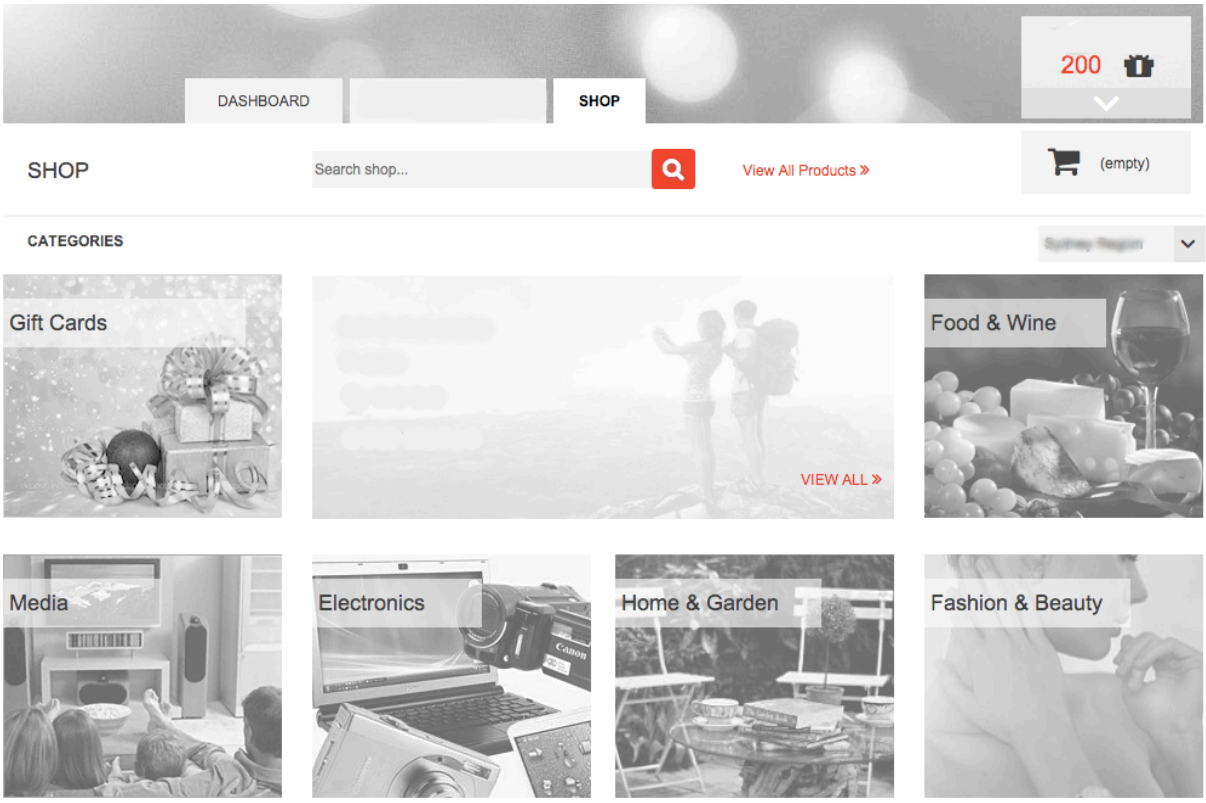

Figure 8. An Example Page of the Wireframes

All collaboration between the two designers was conducted via virtual tools; they were located in different cities. In addition, the visual designer and the project manager overseeing the first phase were both located in different states, so all the communication regarding design and, later, development had to be conducted with the help of a set of different communication tools: the chat tool and tools that enabled sharing the wireframes and designs.

The feedback from the customer, as well as the functionalities of the wireframes, were translated into user stories – short descriptions of functionalities – which were the main source of information for the developers. Whilst the Carmine designers were working with the customer, the developers (one located in Australia, the other in Canada) had been preparing for the development to begin. Once the Escapade management agreed that the work of the designers was a good enough representation of the product, the developers could begin their part. They had stored the user stories into a backlog tool, which also acted as the source of status information for the Escapade managers.

5.5.2. Development: Get It To the Customer

Like the discovery phase, the development phase also unfolded through iterative interactions between the two organisations. The development team had short daily conference calls to discuss the requirements they had

completed during the previous day, as well as upcoming work and pending issues. The Escapade manager was also present for these meetings, as well as other meetings where the requirements were prioritised or the past sprint was analysed. The Escapade product owner was satisfied with the communication channels:

There are many forms of communication that we have. In terms of running with our general Scrumban type of methodology, we have our daily stand ups, we have prioritising every week, we have retros{spective meetings}. In addition we communicate normally via an online chat tool. We find it is fantastic for what we do. Especially keen on the fact that all the developers, external developer team is located all over the world.

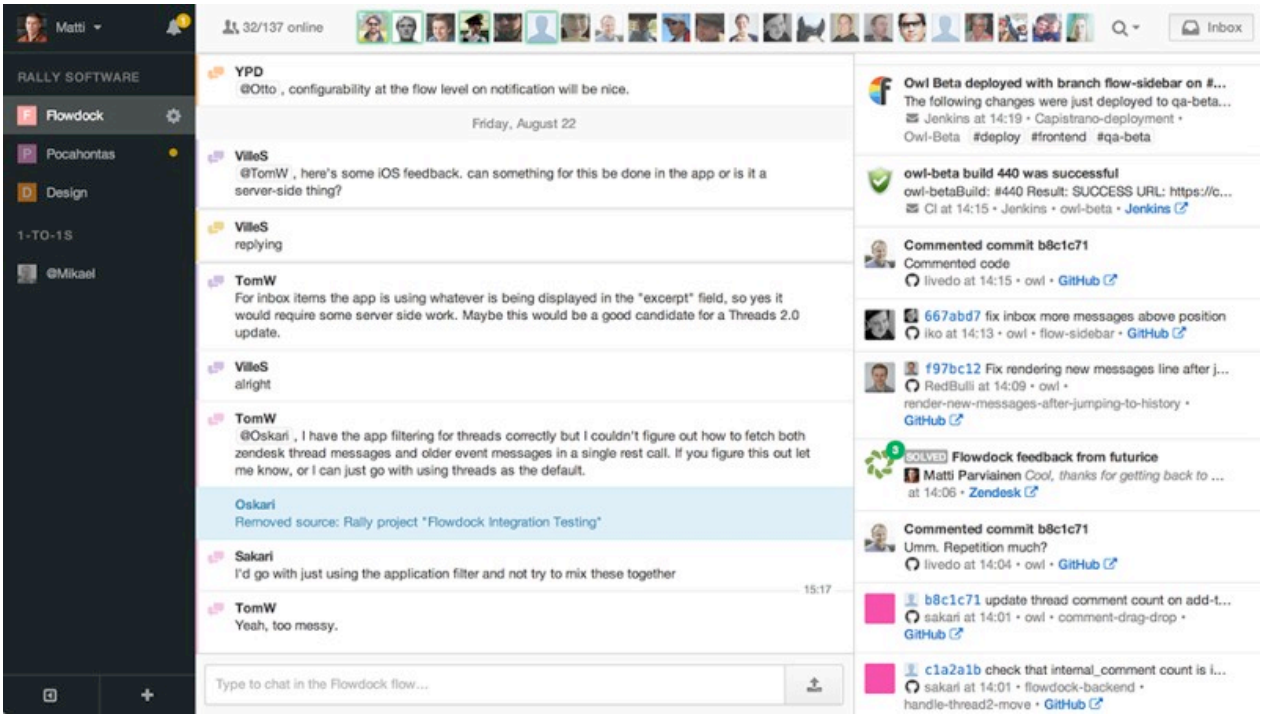

Figure 9. An Example View of the Chat Tool (source: flowdock.com)

The chat tool and the backlog tool were complementary to each other. The chat tool facilitated online discussions between the two organisations, as well as between the developers and designers internally. The discussion with the customers revolved around the requirements in the backlog tool. A manager from Escapade explained the relationships between the backlog tool and the chat tool:

We communicate normally via an online chat tool. The way that we communicate is through threads and we find it is fantastic for what we do... In fact it is more efficient than getting updates in meetings and all that sort of thing…[The backlog tool] is for our backlog management and our general priorities management as well. But we do put communication into [the chat tool] because it is just easier to reference to a feature than within [the backlog tool].

An example view of the chat tool is presented in ***Figure 9*** and the backlog tool applied in the project is presented in ***Figure 10***.

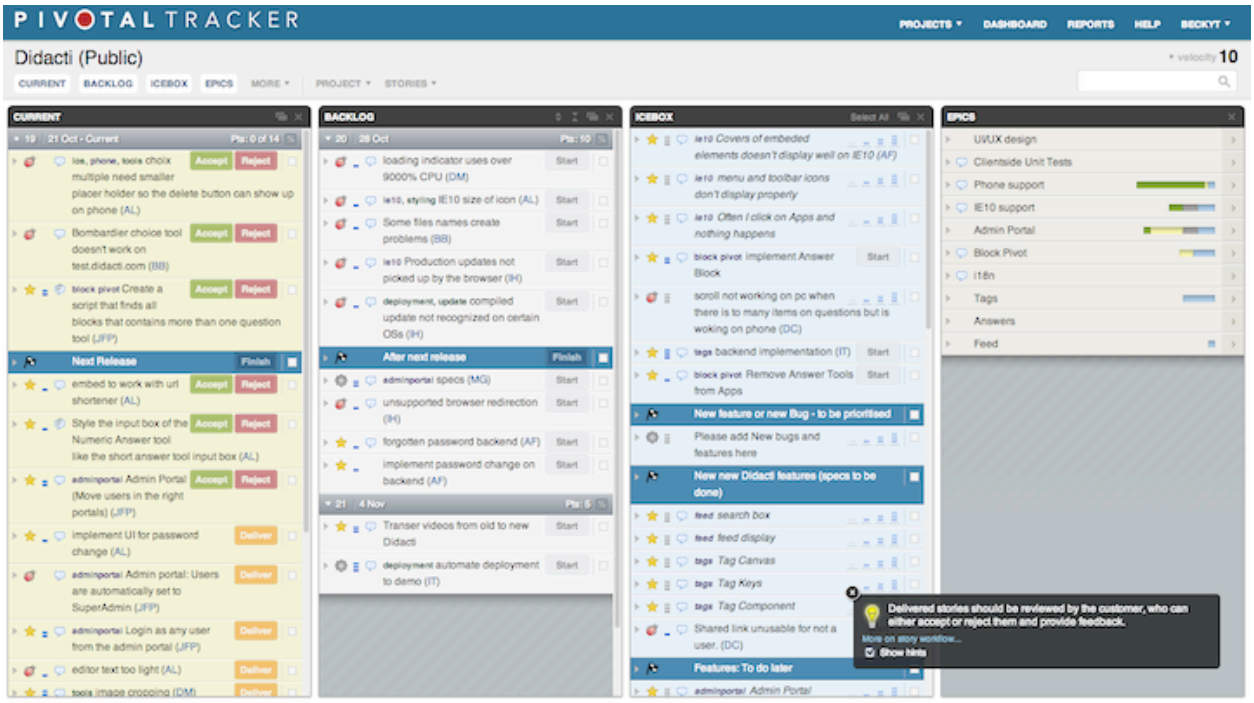

Figure 10. An Example View of the Backlog Tool (source: pivotaltracker.com)

At the end of each week, the Carmine development team released a new functional version of the system. The customer then reviewed it and provided feedback to the vendor using a more refined version of feedback notes used in the discovery phase: a testing spreadsheet which all the testers had access to and which recorded their feedback. The requirements were then updated based on the customer feedback. In addition to the daily and weekly cycles, monthly iterations consisted of reorganising the larger contours of the project, such as the priorities of wider scope requirements, according to the customer feedback.

From the customer perspective, one of the main milestones was the point when the functioning system itself replaced the wireframes as the main communicative point of reference between the parties. Because the system was updated and released weekly, each new version captured customer feedback more faithfully and accurately than the previous versions. The released versions of the system quickly surpassed the wireframes as a representation of the desired system. Consequently, the wireframes became obsolete and were no longer used as a reference point. The visual designs, on the other hand, were merged with the system and were no longer treated as a separate artefact but an integral part of the system.

During the time I was conducting the first round of interviews, Carmine had just released a version of the product that had most of the functionalities. The Escapade product owner called this the ‘beta release’. The Escapade product management had organised a group of testers who were now digging into the product in search of defects.

Some of these testers had already been involved in the discovery phase and they had seen the product evolve from the wireframes into the functional system they were now testing. A project sponsor from Escapade explained how they were able to follow the development by accessing the product itself:

So it went from the wireframes to a sort of mock up. Then the guys were just building features and then from there, as soon as there was something to show us, we could dib in and dib out and test and have a look and see what is being developed.

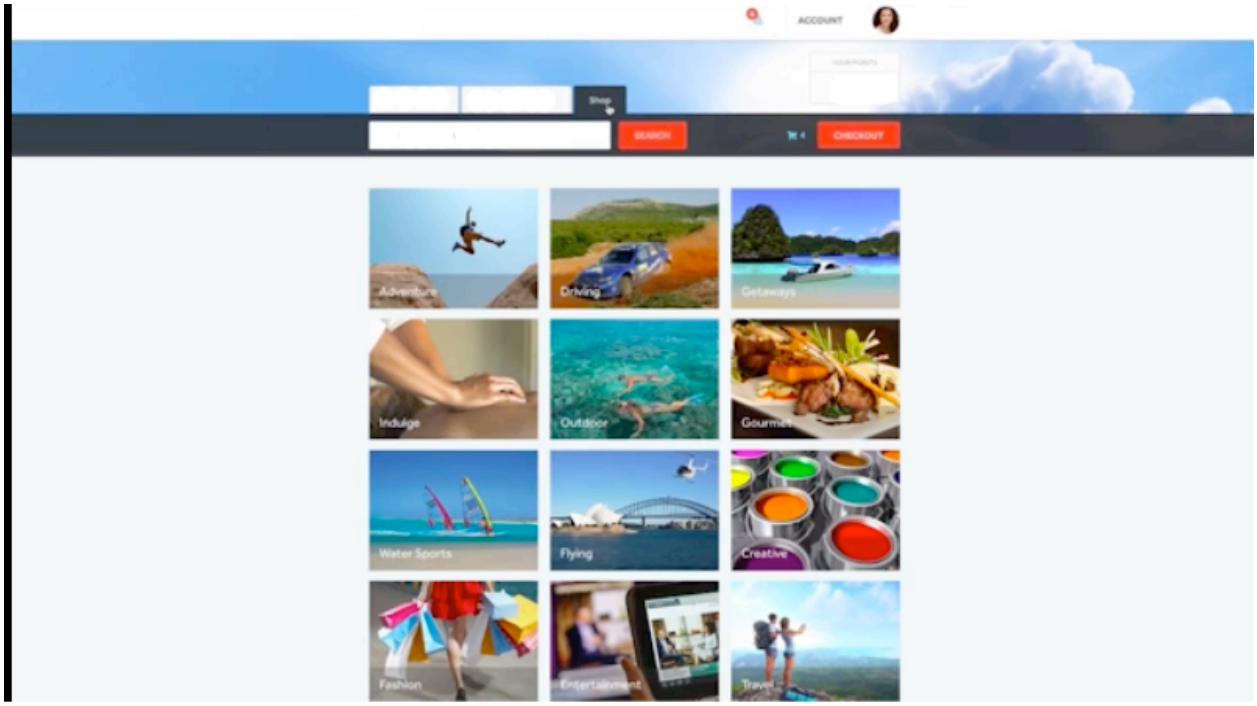

Figure 11. The Landing Page of the First Release of the Product

Figure 11 illustrates the landing page of the product during a latter phase of the development. This version was much more complete in features and functionality than the black and white wireframe.

The product owner had set up an online spreadsheet for the testers, where they could easily log the testing results. The backlog tool could have been applied for testing purposes but a spreadsheet was selected because it was more familiar to the testers, most of whom were not part of the Escapade internal development team. During and after the testing of the beta release, the defects found, emerging ideas, usability improvements and technical specifications continued to bring about further changes. The requirement updates were stored in the backlog tool, which also contained the status of existing and future features. The vendor manager described the constant process of the requirements update, which was actioned via the backlog tool:

If the customer finds a bug, they [the product owner on behalf of the testers] will create a ticket, a bug ticket, in backlog tool. And then that ticket will get assigned to somebody. Our scrum master will assign it, usually based on who was responsible for that piece of functionality. Then that developer will work through that bug. The bug fix is

delivered the same way as a regular piece of functionality. Our customer will get notified when the fix or functionality will be ready for acceptance testing. The Escapade product owner can log in and check the fix or feature. And they can also involve the other testing team people. This communication process is not just for bugs but also feature changes. The product owner can request new features in the backlog tool. And they can have a quick look at it and say if we hit the mark or not with the feature.

A significant part of the development effort was the process of continuous delivery and continuous integration. The code was developed in the development environments and when completed, pushed to the production environment. This process required there to be environments capable of handling the process and monitoring tools that reported any code issues. The development team and the DevOps from Escapade were the main custodians of the environments, but the product owner also had access to the environments and the monitoring tools. He explained why it was important to monitor the status of the systems and react to anomalies:

All that sort of thing is really important, not just for the developers, but for me to understand as well. I'm across what needs to be prioritised in the backlogs and what needs to be pushed ahead because if it's affecting too many clients, we need to make sure that we're moving quickly on those sorts of things.

During the first round of interviews, I enquired from Escapade managers what they thought might be the future issues, after the project would be in the stage of a minimum viable product. There were multiple predictions and ideas of where the project would go next, which made me curious to continue with the case. I decided to re-engage with the project a few months later and study what had happened in the time between the interviews. The next section tells what happened after the release of the beta version and the minimum viable product release.

5.5.3. Beyond the MVP: The Changes Never End

The first near-complete version – the beta version of the system – was released for the internal use of everyone at Escapade. The development of the remaining requirements continued while Escapade stakeholders were using the beta release. The development team captured the feedback from the internal users and monitored their user experience. The feedback that stemmed from those comments was again communicated to the development team through the backlog tool as a set of requirements or changes to existing requirements.

This progress also prompted changes in the contract. Once external clients became involved in the project, the contract was adjusted to accommodate the new requirements that followed. The customer product owner recalled the events:

At the end of July, the first client was brought on board. They were a very large organisation, outside our original target client profile… So there were many things that we had to again jump on, iterate on and make sure they would work... We had to add an additional three weeks of development that was not in the scope [of the contract].

The new phase of the project changed the role of the product owner as well. After the release of the MVP and the first product end users, the development was still ongoing and Carmine was working on the project, although their team was reduced to consist only of developers. The product owner explained the changes:

My role at the moment is to become a lot more, I suppose, strategic compared to what it used be. Before, I was a traditional product owner dealing with, initially it was the product gathering and also the mission statement or mission roadmap and all that sort of thing…We're starting to manage a variety of different conversations, so the product of ours can be great. A lot of the work which I'm doing now is, I mean there's still back office management and everything like that, but because we're running a pretty tight ship development-wise now with the developer, things are moving smoothly in that direction.

The changes in the project contract and stakeholders also prompted a re-examination of the project environments. The product was designed to run in an environment that was fairly modern, but the customers of Escapade were using older environments. The DevOps manager explained why the requirements for existing environments had to be considered when the customers came on board:

There were certainly implications. These things, we've had to go back and do some redirect, make sure they got prepared. The customers, they're very sensitive about data sovereignty, meaning, will my data stay in my country? They will ask: 'Do you have a disaster recovery plan? Can you please document your database snapshotting process? Your disaster recovery process…' But we had all that already.

At the time my engagement with the two organisations subsided, several old and new customers were using the product. The product development continued and Agile methods were applied. The project had succeeded in creating a collaborative customer-vendor team, which mainly communicated over virtual tools. Not a small feat.

5.6. Discussion of the Carmine and Escapade Case

The project between Carmine and Escapade is an example of an Inclusive perspective towards Agile methods and demonstrates how artefacts can be efficiently used to facilitate collaboration between project stakeholders. Next, I will discuss the elements of the project in more details.

5.6.1. Agile – Inclusive

Both Escapade and Carmine were committed to Agile methods, but their members were not zealots, or demanding uncompromising method application. The virtuality of Escapade meant that compromises had to be made when it came to observing any Agile methods by the book. Abiding by all Agile principles, especially the principles of face-to-face collaboration or business stakeholders working daily with the development team, were not feasible for the project. This did not mean that the project was not striving to be as Agile as possible. Workarounds that covered for the lack of constant face-to-face collaboration had been put in place and Carmine especially had trialled these workarounds in numerous other projects before the Escapade engagements. Carmine employees were comfortable with their ways of working and expressed their satisfaction in the ways the project was conducted on their behalf.

The satisfaction was shared. Escapade employees were content in the way the project was progressing. There was mutual trust between the organisations, trust that the work would be done and that the quality would be high. As one of the managers explained:

Part of hiring a company like Carmine, you don't want to have to sit there and examine every little decision that they make…Really, you just steer it vaguely wherever you need to go.

Both organisations were profound in their Agile application, but the needs of the project always came first. The Carmine project manager elaborated on his views on development with customers who were less technically knowledgeable:

Well, usually we try to aim our functionality, the most important functionality is the functionality people can see. Software is a bit like an iceberg where a lot of things happen in the background. Then the cap, they can tell that a lot of stuff is happening in the background but usually the result is somehow visible to the user… You can allude to things that happen in the background... Our clients they come to us because we have the expertise. So if they are not experts in software, they usually are going to trust that we have the expertise.

In both organisations, Agile methods were thought to enhance collaboration and understanding between the different parties. The Carmine user experience designer summarised why she was so fond of the methods:

Daily stand ups and sprint planning helps plan out your work and helps define what gets delivered and when.

The product owner shared the sentiment. The crucial thing for him about Agile development was the greater visibility of the functionalities of the actual product and the ability to get visible results faster. He described to me enthusiastically his view on Agile:

Now, the good thing about Agile is that in the short term everything is very clear. There is a lot of clarity about what is in the backlog, what is done, I suppose in the next sprint so the next week or next couple of weeks…. Right, so, which is really great for building up to a MVP [minimum viable product] within such a short timeframe you know, how we can actually move things down the line, we can start getting, how quickly things can move down the line in a visual format now which is a really fantastic thing to do.

The collaboration and communication as the essence of Agile, combined with the flexible attitude towards the Agile principles, parallels the literature introduced in section 2.4.2 – the section that outlines literature where Agile methods are observed in empirical, constrained settings. The perspective taken in these papers strives to find ways to accommodate the circumstances of the organisations when it comes to the application of Agile methods, rather than the other way round. All stakeholders are included in the best possible way, even if the teams sometimes need to compromise or find workarounds. The literature with a centrist view often describes projects similar to the case of Escapade and Carmine, where a customer and vendor are looking for common ground in a virtual environment with the help of Agile methods (e.g. Sarker & Sarker 2009, Modi et al. 2013). The next section will examine these stakeholders and the collaboration between the organisations.

5.6.2. Stakeholders – Present But Peripheral

The configuration of stakeholders in the Escapade-Carmine project was influenced by the project roles and the constraints posed by the physical locations of the members of the project. Since the development team of Carmine was dispersed across Australia and Canada, face-to-face collaboration between the development team and all Escapade stakeholders was not viable. The physical and contextual boundaries had to be bridged and both

organisations had assigned few boundary spanners. The role of the boundary spanner, as defined by Levina and Vaast (2005), was to establish and facilitate the collaboration. Escapade had their product owner and Carmine had a counterpart, a product manager. Other members of both organisations took part in the collaboration, but most of the communication between the organisations was flowing through the Escapade product owner. Even though some interviewees saw this arrangement as a potential bottleneck, the members of the organisations were generally satisfied with the levels of collaboration. The project manager of Carmine praised the arrangement:

I think we are communicating really well. That is I think is one of the strong things with Escapade. Our whole communication is one of the most important parts of the project. We are in close communications with the project stakeholders and… the product owner is probably talking with my team right now. They are always talking. It is good, we've got very good communication on this project.

The product owner provided the link between the Escapade stakeholders and the project team. The product owner ensured that the comments and feedback were always delivered to the Carmine team. The Escapade product owner participated in the daily meetings with the Carmine development team, discussed the project status internally at Escapade with the project sponsors, sent out a company-wide newsletter that specified the project highlights and upcoming changes, and maintained a visual board that showed the project status in the office premises on a weekly basis. Many interviewees, both from Escapade and Carmine, praised the product owner as a very effective collaborator. There was a consensus amongst the interviewees on the approachability and accessibility of the product owner. The Escapade stakeholders told me that they always knew that they could contact the product owner in case of emergent issues but seldom had to, since the product owner had kept them so well informed. One of the project sponsors described the working relationship with the product owner:

Our product owner has a weekly one-on-one with me on Mondays. So we will go through where he is up to with the project. He will show me the product as is and walk me through the new features that have been released. What's coming up, what's next and this is what we've got to get done.

Sharing the feedback that was instigated by the product appraisal done by the testers, project sponsors and the product owner themselves, was one of the crucial collaboration points. The project manager explained the rationale of using only one person as feedback conduit:

All the feedback, it is funneling through the product owner at the moment. If there's a single point of contact on that sort of a thing, especially someone who has overall vision for the whole product. Otherwise you get a lot of disparate inputs and not everything would be moving the product in the right direction.

The project manager from Carmine was responsible for the completion of agreed development activities; however, the role of the project manager was not a full-time position, as the project manager worked with several customers. The role was created to fulfil the boundary-spanning activities rather than actual management of the development. The development team was self-managed and the Agile methods were enforced by a senior developer who held the role of Scrum Master. The development team handled the customer communication mostly without the project manager's involvement. The main role of the project manager was to keep Escapade up to date on budgeting issues and to react when the Escapade stakeholders had concerns or issues. The project manager from Carmine described his relationship with the Escapade CTO:

I communicate with their CTO quite a bit but it is less frequently. It is more like a weekly basis. I'll give him updates every month on how much budget has been spent and he also raises issues. If he's got any concerns he can contact me and likewise I can contact him. But so far this project has a very smooth process so there hasn't been any big issues to raise.

In addition to Escapade's product owner and Carmine's project manager and team, Escapade's CTO had a prominent role in the technical side of the project. Even though the CTO was less involved in the day-to-day discussions regarding the project, the CTO was the person who was ultimately responsible for the budget and schedule and who could inject more money into the project work. There was only a handful of occasions where he had to intervene with the project and the priorities. One such instance was described by one of the project sponsors. This sponsor told me that she felt that some of the requirements were not given the priority they should have from a marketing perspective and that the original visual designs were lacking when it came to characteristics that would make the designs more suitable for the Escapade image. The intervention by the CTO, instigated by the project sponsor, ensured that the visual designs were given more priority and attention than originally allocated.

In addition to the higher-level project responsibilities, such as the prioritisation of resources, the CTO was granted access to participate in the daily project activities via chat rooms where the Escapade product owner and the developers discussed project details, but he seldom participated in such discussion. The CTO explained why he felt that he did not have to participate in the daily work.

Thing for my level, if I get too much into the detail, with every project… I have major projects that I manage and I get a bit overwhelmed so that's why our product owner is really good. I have a lot of one-on-one meetings with him every week. We go through how he is tracking, how he feels how he is going, any major obstacles, anything I can help with… I usually, we just have a little chat about how it's going, workshop, what are we trying to release, what are the main goals…

The stakeholder interaction patterns are illustrated in ***Figure 12***. Escapade stakeholders are marked in blue and Carmine stakeholders in green. The arrows show who was engaging in more frequent collaboration. The boundary spanner roles of the Escapade and Carmine management ensured that there were links for collaboration between the Carmine development team and the Escapade stakeholders. The Escapade business stakeholders were familiar with Agile project development methods but less familiar with the technical aspects of the project and were not necessarily applying Agile methods in their daily work. The role of the boundary spanners was to ensure that both parties were aware of project events and that the stakeholders from Escapade were always accessible for the Carmine development team as well. The boundary spanners facilitated a project environment where the stakeholders of both organisations always had easy access to engage with the other parties if needed, creating an environment where the less-frequently collaborating stakeholders were constantly in each other's peripheral vision but not necessarily in direct contact.

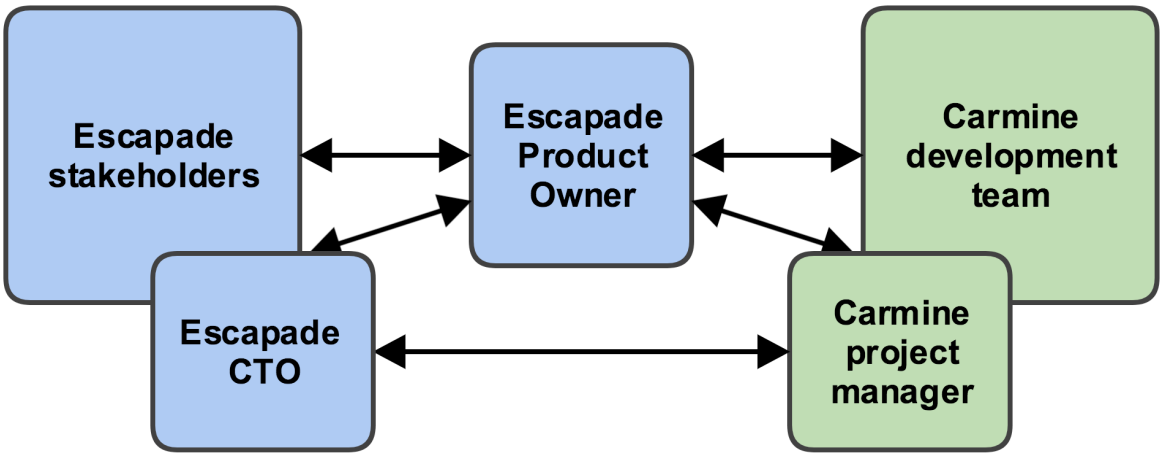

Figure 12. Escapade-Carmine Stakeholder Collaboration

The peripheral view was strongly supported by a variety of boundary objects, which the boundary spanners applied to enhance the mutual feeling of presence for both parties. The next section will discuss in more detail how this feeling of presence was achieved in the virtual project environment.

5.6.3. Objects – Creating Presence

Even straightforward and fairly simple software development projects, such as the Escapade-Carmine project, require a multitude of artefacts to support the development efforts. Several objects were utilised as boundary objects, conveying information between the different parties. Objects ranged from the contract and visual objects such as the wireframes, designs and the system itself, to the chat tool, backlog tool and other collaboration tools.

The contract was used to establish a formal project relationship between the companies. It was the first artefact where the project goals were communicated, albeit in vague terms. The wireframes and the visual designs were used to communicate what the final product could and should look like and what the main functionalities were. During the later stages of the development project, the product itself replaced the wireframes as the main visualisation. This occurred when the product had surpassed the wireframe in details and provided a better reference artefact for the developers and testers alike. Wireframes, visual designs and the product were all applied by stakeholders from both organisations in order to test the usability of the proposed solution, obtain user feedback and to monitor the progress of the work.

The backlog tool, which contained the requirements, was used for capturing requirements as user stories, to visually display the status of each requirement, for example: ready, in development, waiting, etc. It was also used as a platform for testing feedback and comments from Escapade. The chat tool was used to discuss the requirements and issues. In addition, the testing team had a spreadsheet, shared internally online, where the testers captured their test findings and could see if their findings were new or already reported. The product owner communicated the test

findings to the development team, via the backlog tool. This means that the testing process required not only one artefact, but two, in order to properly accommodate all stakeholders.

Other objects were used as well and had a significant role, but even though their application was mutually conducted, these objects would be classified only occasionally as boundary objects. This category includes all the different environments and the tools supporting these environments. They were used to enable the continuous delivery, that is, constant development and testing cycles. The environments also provided tools that monitored the status of the system – information that was shared between the Escapade product owner and DevOps and the Carmine developers.

The environments, including the monitoring tools (illustrated example in ***Figure 6***), were necessary for the virtual development team collaboration. Access to the systems from any location was required in order to facilitate the development efforts by the dispersed team, located across Australian cities and Canada. In addition to development, the environments provided the infrastructure for the testing activities as well. During the early stages of the project, a selected set of Escapade stakeholders was able to follow the product development progress throughout the whole project. When the first, more complete, version of the product was released, the project granted all Escapade stakeholders access to the latest version of the product. In addition to the product, the wireframes and visual designs were also hosted in the environments. The testing activities conducted with the wireframes, designs and the product helped the organisations to establish mutual understanding of the existing and future goals for the project.

Agile Activities	Supporting Artefacts	Purpose of the Artefacts
Daily scrum meetings Weekly planning meetings Customer feedback	Backlog tool Chat tool Testing spread sheet	Alignment of expectations
Prototyping	Wireframes Visual design The system	Identification of different aspects of the problem Exploration of the shared goals
Prioritisation Continuous delivery	Contract Environments	Facilitating development Establish shared goals

Table 18. The Escaped and Carmine Activities and Artefacts

The Agile activities and the objects used to support them are summarised in ***Table 18***. The function of the tools was to support collaboration and create a presence despite the virtual setting. The chat rooms provided a virtual space for the employees of Carmine to socialise. The chat rooms shared with the Escapade stakeholders were fostering the sense that the developers had a constant direct communication channel with the customer. By 'hanging out' in the chat tool, the product owner reminded the development team that they could always ask him if anything puzzled them. The backlog tool illustrated the project's progress and the requirements and provided an opportunity for the alignment of expectations towards the project requirements. The updates and comments on the tasks provided assurance that something was being done, even when the work was still invisible. The constant updates in the

system bestowed even more trust in the team and in the project's overall success, as one could observe with their own eyes when more and more functionalities became available.

5.6.4. Conclusion

The Escapade-Carmine project is a case of successful application of Agile methods in a virtual environment. The flexible and communication-focused approach towards Agile methods and the willingness to create and maintain a virtual presence with rigorous application of artefacts define the collaborative relationship between the two organisations. The Carmine development team always had access to necessary information and ways to obtain feedback from the business stakeholders at Escapade. The collaboration between two organisations in a customer-vendor relationship meant that the dynamics between the organisations were different from a purely intra-organisational setting, but in this case, the relationship was based on mutual trust and understanding of the project goals.

The second case study provides an example of a project where two organisations managed to achieve satisfactory results by applying the right amount of Agile methods in order to facilitate effective collaboration. I see this project as an example of a mature Agile development environment, where professionals were able to perform and achieve the goals set at the beginning of the project. It presents a different scenario for Agile software development, a different level of Agile application and a novel setting.

I argue that the second case is exceptional for one additional reason: every stakeholder agreed that the project was conducted without major issues and that the end results were as expected or even better. The levels of collaboration and trust between the organisations were also rather unique. The real success or failure figures of software development projects have been debated and the statistics have been contested (e.g. Eveleens & Verhoef 2010). However, there seems to be a consensus on the benefits of Agile development, which has improved success rates of software development projects over the last decade. Still, the success rates of projects are still far from being perfect (Scrum Alliance 2015, VersionOne 2016). In this light, a successful project such as the Escapade-Carmine product development can still be seen as a good, educational example on how to conduct Agile projects (Zaitsev & Tan 2016).

The first two case studies have provided examples of product development in smaller teams with relatively simple stakeholder structures. The next chapter will describe my final case study, which encompassed a more complex stakeholder environment with multiple parties collaborating in an equally complex software development program.

6. Case Study Three: Balancing the Stakeholders

"I heard a lot of comments about how this project was the hardest project anyone has ever worked on ... definitely, even thought they have a lot of experience."

–An iteration manager, PrecautionCorp

6.1. Introduction

The third case study tells a story of application of Agile in a large-scale product development project with an intricate stakeholder structure. This case study differs from the previous two in two major ways. First, the organisation discussed in this case study invited me to conduct observation-based research. I was welcome to conduct observations of their meetings, interview the people involved in the program and spend time in their office. This arrangement allowed me to form an understanding of the complex program, the stakeholders and the events. My interviews and observations were conducted throughout a whole year and I could see the program unravelling from the midway until the first releases. The opportunity to follow the program for lengthy periods of time makes this study longitudinal, even more than with the first case study.

Secondly, the case organisation described in this section was significantly larger than the other two. The program I studied had almost the same number of stakeholders than the whole of Extreme Inc. and even the smallest sub-streams were notably larger than in the project discussed in the first case study. Unlike the other two cases, the organisation in this case was spread across multiple locations and time zones, which further added to the complexity.

The data presented in the case description is a mixture of both interview data and observation data. Nevertheless, the first sections follow the familiar format. First, I provide background information on the main organisation involved, describe the goals of the case program and detail the different stakeholder groups. Next, I describe who was interviewed and observed and when. Then, I discuss the Agile approach prevalent in the program and present the Agile methods that were applied. The sections discussing Agile methods and artefacts provide examples of interview data, but I have also included two vignettes that illustrate Agile meetings I had the privilege to observe. Finally, the section concludes with a discussion that ties the case study back to the research questions: I analyse the Agile perspective, the stakeholder collaboration and the artefact application.

6.2. PrecautionCorp

PrecautionCorp (pseudonym) is a large organisation that focuses on different activities in the financial sector across Australia. The organisation was one of the early adopters of Agile software development methods and has since been one of the leading Agile development advocates amongst financial organisations. PrecautionCorp sponsors many events for the Agile community and its reputation is widespread amongst the Agile community.

The company has a strong focus on internal information systems development with the aid of long-term strategic partnerships with offshore IT consultancies. The strategic partner organisations have a history of Agile method

application and their consultants have been trained in the method internally. The strong commitment to Agile methods is evident at PrecautionCorp: a tailored selection of Agile methods is applied to projects conducted in the business parts of the company, not only the IT development. The company has a strong emphasis on the role of Agile coaching and training of the employees in Agile methods. Agile coaches have their own internal network, Agile Guild, which they use to discuss and analyse the state of the practices in different parts of the organisation.

The case study discussed in this research focused on a large product development program, which involved multiple departments and various stakeholders throughout PrecautionCorp, as well as a product vendor organisation. The goal of the program was to replace an existing product with a new, tailored product and simultaneously streamline business processes intertwined with the product and the services it offered. The main stakeholders involved in the program were the business and technical departments of PrecautionCorp, strategic partners and the vendor of the new product, who consulted on the development and configuration activities. In addition, the program had a silent but important stakeholder: the regulators who govern the ways information systems should be built in the financial sector.

The program was organised into several development streams. One of the streams, the business stream, focused on the business requirements and process streamlining activities, whereas the other streams had different technical scopes such as the product configuration, data integration or the front-end look-and-feel design and implementation. The business requirements and process streamlining activities consisted of a range of topics from definitions of the current and new, simplified product workflows to updated legal documentation and customer communication required by the regulations. The business stream encompassed stakeholders that were responsible for the financial activities of the organisation, as well as legal and marketing teams. The development teams were a mixture of offshore and onsite teams. A mixed team of PrecautionCorp employees and strategic partners from India, supported by the consultants of the product vendor organisation, conducted the back-end development of the product. The front-end development was tasked to an internal PrecautionCorp development team, who worked in a strategic partnership with offshore employees in China.

The stakeholders are illustrated in ***Figure 13***. In order to best convey the difference in stakeholders of the PrecautionCorp program, I have applied colour coding that describes the roles of the stakeholders. The PrecautionCorp teams that were involved in the technical development of the product are presented in green. The external vendor organisation, consulting the technical teams, is marked in pink. The internal business stakeholders, who took part in the project but conducted non-technical activities, are marked in blue. Finally, the regulators, who were not part of the PrecautionCorp program but who provided guidelines for the product development, are connected to the program with a dotted line and marked in purple.

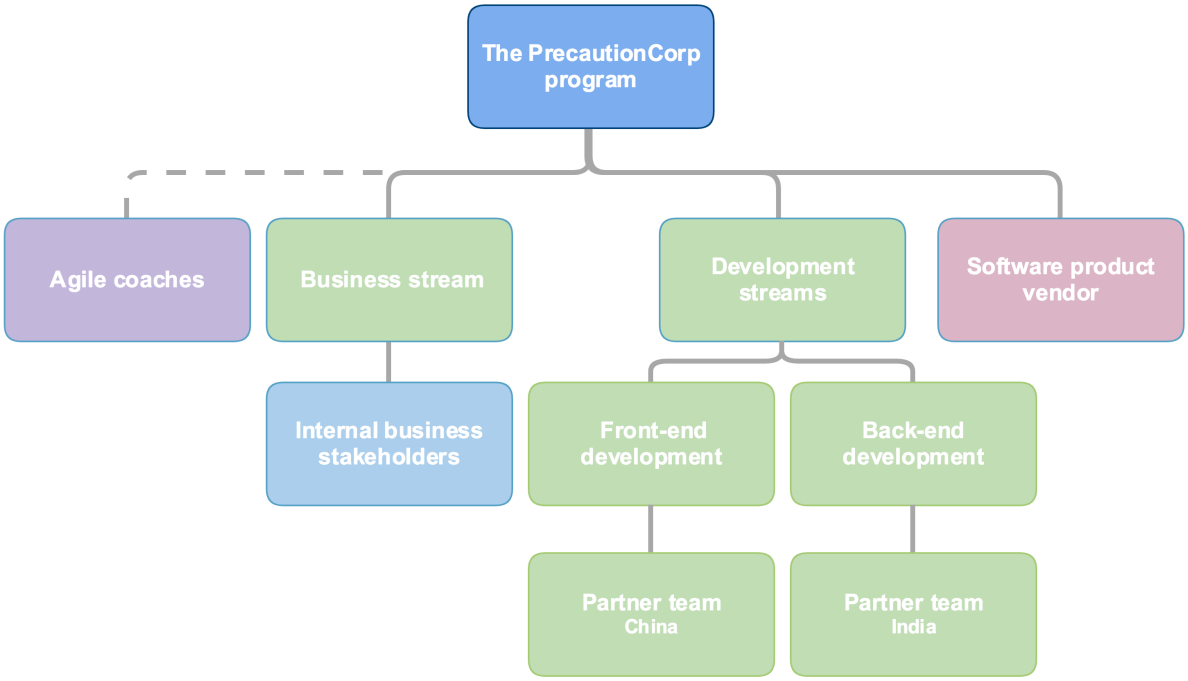

Figure 13. PrecautionCorp Program Stakeholders

An iteration manager, often supported by a more technical delivery lead, helmed each stream. The responsibilities of the iteration managers were described as a mixture of project manager and Scrum Master (Schwaber 2004). All other stream teams, except the business stream, consisted of a combination of team members from both PrecautionCorp and technical consultants from strategic partners from either India or China, supported by the vendor, who were mainly located in Australia. The sizes of the teams in each stream varied throughout the length of the program, but the smallest of the streams was ten people strong and the larger streams had over forty members. Some of the offshore partner employees were temporarily working in the Sydney offices and some of the PrecautionCorp employees were temporarily working from the offshore locations in order to bridge cultural and temporal distances.

The business stream consisted of internal PrecautionCorp stakeholders, working on the project part time, in addition to their usual business roles. They were located across the PrecautionCorp offices. Some teams had assigned product owners or other business stakeholders; some only had access to the lead of the business stream. The differences between the teams contributed to differences between the work practices, as the teams were allowed to organise their own work as long as it fit the larger Agile framework, taught and maintained by the Agile coaches.

6.3. Precaution Corp Interviews and Observation

My engagement with PrecautionCorp began when one of the employees of the organisation became interested in my research topic at a technology community event. She mentioned that the organisation had a large program that was in its early stages, a great opportunity for a longitudinal study. I was introduced to the Agile coaches of the program,

who then invited me to visit the office. Once the formal agreements were in place, I agreed on the first set of interviews with the employees and scheduled observations of a selection of meetings that would help to understand the program and the stakeholder structure. My primary contact and some of the employees whom I interviewed in the beginning of the engagement suggested which meetings would be beneficial to attend.

Soon I also learned that the team was scheduled to move to another office mid-program and I could see how their physical boundary objects would be transferred across. I was able to observe meetings and boundary object application in two different locations and compare how the space impacted the Agile environment.

For the observations, I had a contact person who could let me onto the premises and with whom I could check when and where further interesting meetings were taking place. My access to the premises was limited by the availability of my contacts, but once in the office, I was allowed to wander around and observe the work and the meetings. During the interview and observation visits, I had a chance to interview at length several people who were employed by different departments of PrecautionCorp and in addition, I conducted interviews with employees of the strategic partners and freelance consultants who worked with the onsite team. The interviews were accompanied by less formal chats with familiar employees, exchanges of common pleasantries, but also some insightful status updates and offhand comments, which informed me on the process of the project and the current sentiment of the program members.

However, even though I had access to the meetings, my observation capabilities were limited by the time constraints of my informants and in order to be non-intrusive, I chose to focus on the larger meetings with multiple stakeholder groups so as to better understand the internal stakeholder communications. The two meeting types I observed were the showcase meetings and Scrum of Scrums. The showcases were larger gatherings, presenting information on the program. All the program stakeholders were invited and the agenda consisted of updates from different stakeholders. The topics varied from program goals to product demonstrations.

The Scrum of Scrums meetings, following the structure of daily Scrum meetings (as discussed by Schwaber 2004) were meetings between the iteration managers, technical leads and program management. They were held in open-plan office space, in front of the program wall, which captured the activities of all the different streams. The meeting had a regular time and space that was known to program participants, so anyone interested in the current status of the streams could attend the meeting. By observing the showcases and Scrum of Scrums meetings, I acquired insights into the organisation and the different departments and stakeholders, as well as the strategic partners, the domain expertise of the industry and the workings of the product itself. A summary of interviews and observations is presented in ***Table 19***.

Organisation	Informants	Themes discussed/Observed
Interviews in April to July 2016 *indicates that the person was interviewed twice		
Business stakeholders	Four program managers/business stakeholders Two other members of the organisation	Organisational culture and history, hiring practices, role of involved stakeholders, communication tools and mediating artefacts used, Agile practices enacted, audits and regulations
Agile practitioners	Four iteration managers Two tech leads/delivery managers Two business analysts Two Agile coaches* One lead Agile coach	Systems development process, communications, communication tools and mediating artefacts used, Agile practices enacted
Meeting observations and office visits		
December 2015	Office visit and introduction to the program, Interviews in the office	The office space, the program wall
January 2016	Interviews in the office	Program progress
March 2016	Scrum of Scrums and two showcases observed, an online call with the Agile Guild	Stakeholders, Agile processes, program progress
April 2016	Scrum of Scrums observed	Program progress
May 2016	Two Scrum of Scrums observed	Program progress
July 2016	Scrum of Scrums observed	Program progress, office moving plans
July 2016	Interviews in the office	New office premises
August 2016	Scrum of Scrums observed	Use of the wall in new premises
September 2016	Interviews in the office	Program progress
December 2016	Catch-up interview	Program progress

Table 19. PrecautionCorp Interviews and Observations

6.4. Agile Approach: a Balancing Act

'Complex' was among the first words I heard from my contacts when they began describing the program at PrecautionCorp. The sheer number of stakeholders would have been enough to create a very complicated development environment but in addition, the product that was being developed was technically complicated and related to a major restructuring of product offerings of PrecautionCorp, product offerings regulated by the government. The goal of the project was to replace an older product, which had overly complicated offerings for the customers, with a simpler, newer version from a different vendor. The program consisted of the software product tailoring elements, as the off-the-shelf version of the new product was not usable for PrecautionCorp, as well as restructuring of business product offerings and business workflows inside the organisation.

In the next sections, I will discuss the different elements of the project. First, I will describe how Agile methods were applied and how they were modified to fit the globally distributed environment. Then, I will describe the daily activities of the project and how different stakeholders were incorporated in the program. I will discuss what artefacts were used during the program and illustrate the meeting practices via two vignettes.

6.4.1. Offshore Development: Agile from Afar

PrecautionCorp was a known frontrunner of Agile in the Australian financial industry. The organisation had adapted the methods and created their own set of methods and best practice, applied both by the technical software development teams and the business teams of the organisation. PrecautionCorp's methods and best practices resembled a tailored version of Scrum, with some elements borrowed from other Agile methods. The method collection was taught to all employees by an internal team of Agile coaches, who also supported teams with their method application. The Agile methods were specifically tailored to fit large-scale development projects with offshore teams involved. According to my informants, PrecautionCorp was amongst the first large organisations to adopt Agile in any industry sector in Australia. Unlike in many other Agile adoption stories that were told by my informants, Agile was not introduced to PrecautionCorp gradually as a grassroots movement, but as a top-down effort. The adoption was driven by managers who had seen that conducting projects with Agile methods could provide great results and improve both the quality and schedules of projects. When I began observing the program, the software development functions at PrecautionCorp had been practicing Agile for close to ten years. Thinking of Agile development as the default development method had permeated all development functions as well as other parts of the organisation.

Application of the Agile methods were supported by a cohort of Agile coaches. The coaches supported the teams both by embedding themselves in the teams and by forming their own team of Agile coaches, an Agile Guild, that allowed exchange of experiences across the boundaries of the departments. In addition, the coaches engaged in collaboration with the strategic partners as well. One of the coaches described the cross-organisational role:

We are part of the agile coaches team but our role is to support the development teams. But we also have what we called centre of excellence, where the agile coach from [PrecautionCorp] get together with agile coaches from all our partners, from [the partner offshoring in India] and [the partner offshoring in China] and [other partners]... So we get together every two weeks, then we discuss the more complex issues we face as agile coaches.

All new people joining PrecautionCorp had induction training on Agile methods and were also encouraged to participate in 'cultural training' as the interviewees called the classes that were aimed at enhancing communication skills when collaborating with the offshore partners. PrecautionCorp had their own Agile training materials and Agile methods specifically tailored to fit the distributed development model. One of the Agile coaches provided me with a slide deck that explained the Agile model of PrecautionCorp, detailing how the development should be conducted based on the best practices of global development. The training included concrete tips for collaboration, as illustrated in one of the training slides in ***Figure 14***.

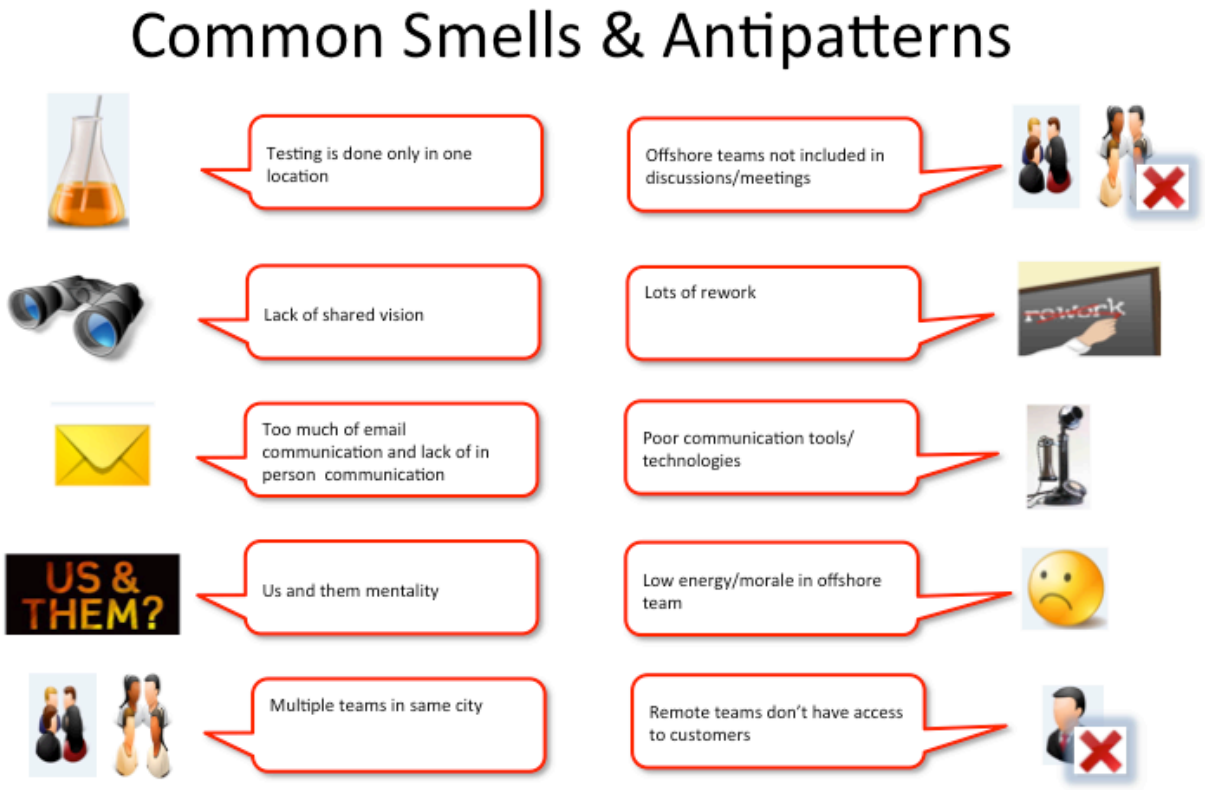

Figure 14. Distributed Agile Training Example Slide from PrecautionCorp Training Deck

Both partner organisations involved in the program were practicing Agile development as well, but I was told they were conducting their own Agile training. The teams and departments were not mandated to follow any specific set of methods. The coaches explained their role in supporting the development:

We encourage being different. So it's ok to not have the same look and feel as other programmes. It's ok, whatever works best for that local context of course. So as agile coach what we do is we go in, we look at the context of the problem and then change the behaviour around that context. Instead of, this is a process you need to follow and then you get a – so agile is not a process as you know, it's all – it's very important that we understand that. You have to be there on the ground, on the site of context and adjust your way of working accordingly.

The Agile coaches and the members of the partner organisations had had years of experience in Agile development in other organisations, but many other employees, especially the business stream stakeholders were encountering Agile methods for the first time at PrecautionCorp. Even though as an organisation, PrecautionCorp could be considered 'mature', when it came to the application of Agile methods, many employees who had joined the program were either from the business functions and thus less exposed to the methods or fairly new graduates, fresh from university. One of the business stakeholders contemplated his Agile experiences in the program:

I've been involved in other projects and understood it {meaning Agile}, but it's not really til I've got involved here that I really understood what Agile was about. To me, originally, I thought it was just about putting cards on the wall and trying to track it. It's a lot more than that.

He attributed a lot of his learning to the engagement of the Agile coaches and the program leaders:

I'll have to admit, I wasn't all that great with Agile at the start. I've learned a lot since the beginning and if I knew what I know now at the beginning it'd be different. In terms of their involvement, it's been great. They've been very much engaged in the project, they've been very hungry in terms of wanting to know more about it, be involved. Which is really good. I've really been surprised by that. I've been really happy with that and giving feedback and knowing what's going on. That's been great.

Learning and flexibility were appreciated traits amongst the interviewees and the organisational culture was seen as positive, encouraging and innovative. One of the iteration managers, who had joined the organisation a few years previously, was delighted at how easy it was to move around the organisation:

I would like to say, working for PrecautionCorp, they require the people to be more flexible. That means that you need to be moving around all the time. In the past, I most of the time stuck to one or two departments. Here I came, I'm moving. I'm moving a lot already. So innovate, it's part of the life already.

Although balancing Agile methods and offshore development can be difficult (Paasivaara et al. 2012), PrecautionCorp was described as having a successful and mature working relationship with their offshore partners. The two partners, located in China and India, were from different organisations, operating in different areas of the program. The stream responsible for the user interface had partnered with a well-known and reputable global Agile consultancy. Their offshore team was located in one of the large cities in China. The collaboration between the teams was mostly described as well-functioning and mature, but a small number of disgruntled voices were not always happy with all of the quality of work, stating that the team could be more self-sufficient and require less guidance. Nevertheless, everyone agreed that the communication was well established.

The team in India was split across several sub-teams. The Indian offshore development was mostly involved in the back-end development, integration and migration of data aspects. A significant time difference was the main complaint when discussing the collaboration with the Indian developers. The overlapping hours had to be utilised effectively and the teams used a variety of different communication tools from videoconferencing calls to chat tools and emails.

The third major stakeholder, in addition to the two offshore partners, was a software product provider, a vendor of the product that was being used as the core for the new solution. A technical lead described the difference between the vendor and strategic partner relationship:

Being a vendor is… if you want to purchase some, if you go and purchase that TV, a Samsung TV, you go to the shop, pay the money, buy the TV. That's the one-time transaction. Now, a strategic partner is more like a cable connection. If you take a cable connection to watch the channels on that particular TV, then it is not a one-time transaction. You pay them for their services to be provided so that you pay them month by month, or year by year, to watch the channels. The service providers are the partners, really.

The vendors were the only stakeholders who were not already practicing Agile development. An Agile coach described the relationship between PrecautionCorp and the vendor and what had been planned originally:

There's always friction and tension between them and the customer. Well, nothing's ever perfect… They are not an Agile shop. The million-dollar question is how do you get a waterfall window operating successfully with an Agile team? That was one of my challenges that I was working through very early on… We came up with what we thought was a good way of managing it, but ultimately things are not done as quickly as we do with our methods.

The mismatch between the software methods of PrecautionCorp and the vendor forced the managers of PrecautionCorp to apply more controls than they would usually apply in their own methods, as was described by one of the iteration managers who had worked in other projects at PrecautionCorp for years. Nevertheless, applying such a hybrid perspective of control and Agile was not unusual for the management. The program managers and the Agile coaches were aware of the situation but confident that this combination could function.

6.5. Agile Practices and Artefacts: Agile Methods, the Large-Scale Way

The Agile methods were closely observed at PrecautionCorp. Each stream held daily Scrum meetings separately and each stream had its own Agile wall, physically or virtually, depending on the makeup of the team members. Some streams had employees scattered between two or more onshore locations, located in home offices in the same city or in other large Australian cities, whereas other teams were globally distributed, team members located in either India or China. None of the streams had all project stakeholders physically present all the time, even when the members were all located in the same city. Working from home was commonplace at PrecautionCorp. The teams had a roster in place, showing the days when people where working from home or from the office. The remote work meant that meetings had to cater for both face-to-face discussions as well as people who were located offsite, who commonly dialled in via VoIP calls.

The daily stand-up meetings were formatted according to the Scrum guidelines. An Agile coach explained the common meeting structure:

If you look at the standard, the standard is: 'what do you do today, what are you working on, what are you going to work on next, and what issues or problems do you have'. That's the format of the standards.

In addition to the stand-ups, the PrecautionCorp teams had recurring iteration planning meetings and retrospective meetings. The notes and action items from these two meetings were usually stored in a document repository system, not on task cards in a backlog tool. An iteration manager explained why the stand-up meeting information was stored separately from retrospective meetings:

We've got our retro[spective meetings]. So we do use [a document repository tool] for that. It is just more so that people offshore can set aside what they've thought about the iteration, what made them happy about it. They can put their input into it. We have created a page in the repository and it's going well, what didn't go well. What do you think could be done better.

The backlog and documentation tools had large roles as collaboration-facilitating artefacts. One of the iteration managers discussed the documentation that was stored and who the contact people were for pieces of information:

We have vision and design artefacts posted online, in our document depository. So if anyone needs to know what, let's say for example, the identity manager team, is actually working on. Anyone interested can reference all these documents there, in the repository. And the teams, they are available to chat as well. Everyone sees the documents, everyone is transparent. If there's any issues, any problems that we find half way through the work, tech leads are usually the first point of contact there. So tech lead can point the direction for developer who know the best and can help.

In addition to designs and vision documentation, the tools hosted a set of architectural plans, product release plans and other documents. The intricacies of the program meant that information had to be stored for people joining later and the lack of face-to-face communication opportunities further increased the importance of easily accessible, online information. One of the newer iteration managers, who had stepped into the role mid-project, explained how she found the existing plans very helpful:

My predecessor had already created the flight plan done, the high-level schedule. That was helpful. She had whole spreadsheet with the stories broken down, estimates, everything was there. And there was already a lot of work in done from the analysis side of things, I think they just started testing. So it was pretty, it had already a good foundation in place.

Furthermore, the program tasks and progress were very visibly presented in form of Agile walls, both physical and virtual. The most prominent of the walls was the program wall (presented in ***Figure 15***) that combined all the different streams and provided a high-level perspective of all the different activities and their status. The program wall was used mainly as the backdrop for weekly meetings between the iteration managers and other stakeholders in meetings such as Scrum of Scrums, facilitated by the program manager. The showcases, where the results and outcomes were demonstrated to a wider company audience, were also held next to the wall, but the wall was not referred to in these meetings. An Agile coach was showing me the program wall:

The program here has a massive wall because it shows you all the iterations that we planned out and the various streams working on them. Sort of your master game chart, everything is summarised. Each of those lines there is a different team, and a team will have dozens of task cards in the [backlog tool], so that's not duplicated on the wall. So it's a different level. It avoids duplication a little bit. But why bother [with this wall]? Because, as I said, [the backlog tool] is not very good at following management. So, that's why we like to make it visible, that people can easily see where we are at.

Despite the physical distance, many teams chose to create their own physical walls and post them around the office. These physical walls were replicated as a virtual wall in the online backlog and documentation tool. PrecautionCorp had shared the access to the backlog and documentation tool with the partners and the vendors and the virtual walls were accessible to the offsite stakeholders online. One of the iteration managers explained the relationships between the physical and the virtual walls, when showing me the virtual version:

So this is the example of the wall, you would have seen similar physical walls around the place. {Shows a screen with virtual wall}. So for me, I work globally. My team has people in other cities and in China and India. As far as the programmers are concerned we're about as spread out as you can be… I prefer to use a [virtual wall tool]. It's our virtual wall.

Another iteration manager discussed the diminished importance of a physical wall:

The [physical] wall is not as required as it was. We can use the virtual [wall], it's just as good. I update the scrum walls. So I've always got my cards up there with exactly what was claimed for that iteration. Risk is up on my [backlog tool] wall. I've also got risks up on the program wall if it's going to impact the program.

Figure 15. Program Wall at PrecautionCorp Old Office

In the next two sections, I will provide two more detailed accounts of two meetings based on an edited version of my observation notes. I have also included two drawings of the meetings. These drawings are based on my sketches from the meetings. They illustrate the locations of the walls, screens, people and the other work that was taking place in the office during the meetings.

6.5.1.1. Vignette 1: A Showcase Meeting

This vignette illustrates the way the showcases were conducted in the old PrecautionCorp office space. During this showcase, the business stakeholders, who worked in marketing, the legal department, the finances department and customer support, shared the results of their work.

> Showcases and other larger meetings are held in a space between the two main office areas. On one side of this space, there are two small, glass-walled meeting rooms. On the opposite side, there are a few desks, a sofa set and the massive program wall taped, pinned and Blu-tacked to the actual wall.

The showcase is scheduled for early afternoon and it begins almost on time, save a few minutes. There are probably twenty to twenty-five people present at the beginning but more people are coming over from other parts of the office. The overall spirit is very upbeat and everyone seems exited to hear the upcoming presentations. The space is not big enough for this many people to comfortably sit around the table, located in the middle of the room. Most of the people are standing in the two open spaces between the two walls; some are sitting on the couches in front of the screen. A few people have brought extra chairs into the meeting area. There is a screen set up on the desks, opposite the large program wall. A few people have brought their laptops with them but they remain closed at the beginning. (***Figure 16***. presents a stylised version of my original sketch of the meeting area.)

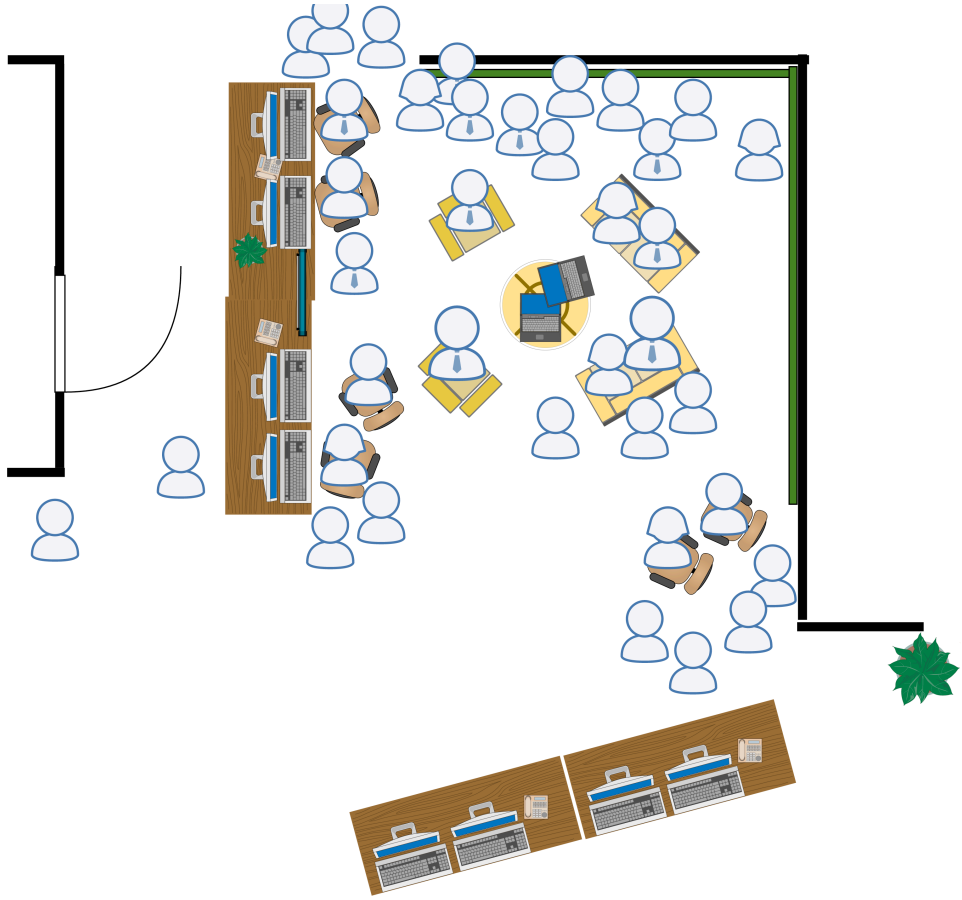

Figure 16. The Meeting and Program Wall Area

The lead of the department opens the meeting and introduces the presenters. The presentation topics vary from team structures to product details and future marketing plans. The first presenter discusses a new support team structure. She states in an excited tone: “We are building the knowledge about how to build the products. We have to be ready to deliver to the end customer. We are pushing people.”

A representative from the company’s marketing department is presenting next. She allows the employees to get a sneak peek into the ways that marketing operates. She begins her presentation by discussing the customer proposition and the naming process of the product. “We put a lot of

work in that name, we had about 70 different names out of which we selected the current one." She elaborates what the market research has shown about the customer mindset of the product: "They don't see it as easy. They want someone accessible, with low cost and a lot of control." The marketing presentation wraps up and the department lead reminds the audience how the new product will affect their customers. It has to generate a great referral rate: "We can't just build this and wait for them to come. We are giving our customers a brighter future".

The next presenter is a company lawyer, who discusses the legal aspects of the project. At this point, people start shuffling around a bit. The presentation is complicated with multiple acronyms which some of the audience (including myself) are probably not too familiar with. The department lead praises the ease and simplicity of the new agreements and introduces the next presenters, the stakeholders who represent the finance department.

The financial stakeholders both position themselves around the screen. They show a complex slide that represents the project untangling. "This is really what our objective was." He describes that the program will "Simplify reporting, remove risk, less custodian costs and less transaction costs and also less payments for the regulators."

The time is now half past two and people seem to be getting tired and fidgety. The team lead wraps up the showcase by praising people who have been involved for their hard work on the case: "One night they were here until 2 o'clock in the morning!" He tells an Enron-related joke, and then: "It's all about the legal structure, investment structure. We are the oldest, bloody 100 years of legal stuff."

Next the lead Agile coach discusses the Agile health check. The coach is located in Brisbane, so he is not physically present but calls in via teleconference. He discusses the results of the health check and lists firstly all the good things: "We also need to celebrate success, we must remember that." A deck of slides with the health check results is displayed on the screen. The health check report ends with some remarks from the lead Agile coach: "Use the word 'thank you' twice a day. And be nice to each other, say thank you to your colleagues." The department lead adds: "We should all channel Bill and Ted. Be excellent to each other." The audience chuckles at this remark.

Finally, the department lead introduces a few new people and several people are awarded with movie tickets as a small token of their hard work. "Cheers everyone and good luck", the lead wishes. The audience applauds both occasions. The meeting adjourns and people break out into smaller groups to discuss with each other.

The program had been divided into three major releases – the first release marking the time when the minimum viable product version of the product would be released to a limited set of users. The showcase meeting described in the vignette took place in the middle of the first release. A few people had already seen the product and had a chance to test out the functionalities but for the majority of the people present, the next showcase was be the first proper look at the product. At the time of the showcase, the program was progressing as planned and there were no major obstacles.

The meeting described in the second vignette took place in the new office space, shortly after the program team moved into new premises. This meeting offers a small peek into the program dynamics at a time when the program was closing on the first deadlines and pressure to meet the deadlines was starting to build.

6.5.1.2. Vignette 2: A Scrum of Scrums

The following vignette is a description of Scrum of Scrums, a meeting between the iteration managers and partner managers. By this time I had interviewed half of the participants of the meeting and was aware of their individual roles and the overall program goals. The understanding of the program itself helped me to observe the dynamics between the people better, when less effort went into understanding what was discussed. This meeting took place in the new office premises, so I sketched another illustration of the location of the meeting, presented in ***Figure 17***.

> There are three iterations left before the planned R1 release. The new office is more open, with fewer walls between the long rows of desks. Meeting rooms are not visible next to the wall; they are tucked away around the kitchen and breakout area. Many employees now have two screens. The office space looks sparse; there are no personalisation items there yet. Some people have to work from home to free up desk space. There are no sofas or chairs or squeaky pig toys next to the wall, just a row of impersonal desks.

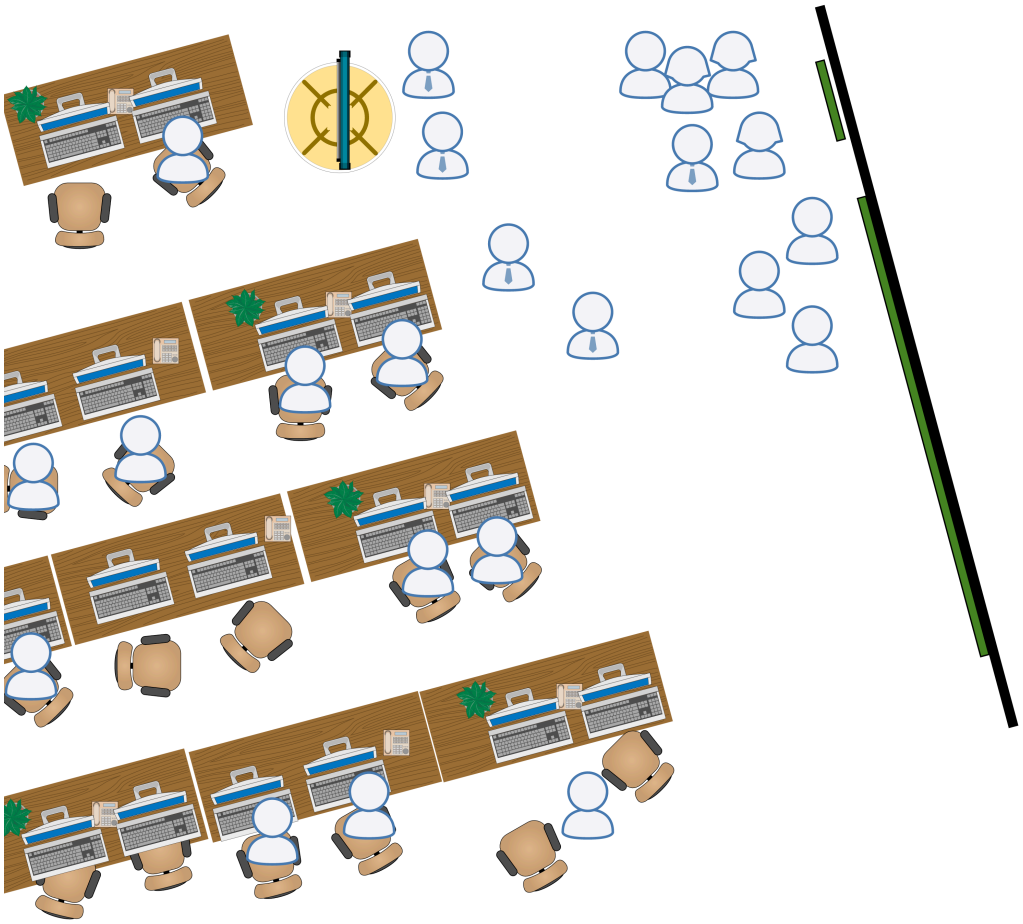

Figure 17. The Scrum of Scrums Meeting

> The program wall dominates the corner and is clearly visible once one enters the office space. The wall has been replicated identically to the old office. The task cards are all moved and the other diagrams and charts, such as risk charts and Gantt charts are also placed next to the wall.

Next to the wall there are a few screens for conference calls, but in this meeting, no-one is dialling in. There is more noise than in the old office coming from other parts of the office and it makes it harder to hear other people. People are having discussions across the room and phones are ringing.

The program manager begins the meeting by stating that there is one week left for the code freeze. His tone of voice is firm and urgent. The meeting begins by addressing the risky streams, which are marked by red dot stickers applied on the task cards. The program manager urges everyone to be diligent with the use of the tools: "If you are in red and not working this weekend, you need to explain to me why. Everyone needs to update the backlog tools at the end of each iteration. Our business stream manager needs this to happen."

Each stream is discussed in turn. People walk to the board and pick their critical tasks from the board one by one. I observed a similar meeting two weeks ago and this time the people seem less relaxed. Maybe the deadline pressures are creeping up on them? The circle, in which everyone is gathered around the person explaining their stream task cards, is tight. The managers are huddled together.

It appears that one of the streams has more issues and unexpected new plans which most of the other managers were unfamiliar with. The other meeting attendants begin asking for clarifications about the new plans and the new scope, as well as the dependencies it might have. The discussion shifts from higher-level status information into a more detailed level, until the program manager decides that these discussions have to happen in some other forum.

There are risks related to new employees, as well as missing issues. The issues seem to concern only a small portion of the meeting attendance, as some of the attendees break away from the meeting and continue their discussions amongst each other a few metres away from the main meeting crowd.

The program manager begins wrapping up the meeting with final reminders of the pending tasks: "Is everyone ok? Yes, ok, we will see how it goes. I've got two more things; make sure your walls are up tomorrow. I need your risks to be up to date. Take some time today. We need to work this tighter [speaking of risks and dependencies]. If you feel like you need to take five off… scream at someone, stand outside for two minutes, punch a wall…" The punching comment elicits jokes from the iteration managers about who was going to volunteer for punching. Finally, the program manager closes the meeting by pointing at the massive jar of various sweets on a table near the wall. "There are chocolates and lollies, keep your energy up!"

The two vignettes – one from the old office and from a large showcase meeting, another from the new office and Scrum of Scrums meeting – demonstrate the use of the office space, technology and the physical walls. These two meetings took place several months apart. Between these two meetings, the product development had progressed from the very early versions to the first version that was going to be released to stakeholders.

6.5.2. Next Steps After the First Product Releases

My observations of the PrecautionCorp teams lasted for several months and during that time, the program progressed from a stage when the development had just begun to a first actual product release. Many changes occurred during my time spent interviewing and observing the program. The iteration managers were swapped across the streams and some people left the program and joined other programs at PrecautionCorp. The role of the partners also changed throughout the program. One of the iteration managers explained the change:

Initially, the Indian strategic partner was engaged to work as the reverse testers and business analysts. Eventually, as we progressed, then the Indian partner was taking up higher role in terms of doing the project management. Now we have also taken up for funding the role of development manager.

Similarly, there were changes in the stakeholder structure. The closer the program got to the first releases, the less the business stakeholders were needed to work on the program. Their input was very important during the design and development, but once the product plans were set, the engagement between the developers and the business stakeholders became more sporadic. On the other hand, the role of the vendors was important across the whole program. Even though the methods that the vendors applied were not fully compatible with the PrecautionCorp Agile way of working, the interviewees thought that the vendors were doing their best to work with them. As one of the iteration managers stated:

I have a lot of respect for them. I think their consultants are incredibly professional, I think the service they provide is incredibly professional. They're very diligent; they're passionate about what they do and the product that they look after. I respect what they do.

The first release of the program was successful but not every original target was met. Most people were happy with how the first part of the program had gone, even though many were saying that the final few weeks had been exhausting. After the first release, the program underwent a set of changes. Iteration managers were shuffled across the streams and the Agile practices were reinforced by adding a new Agile coach as the facilitator for the meetings and development practices. The new coach explained:

You'll also notice the programme wall is empty. We're calling it a programme reset, which again, it's a positive thing. What we missed, there were a lot of challenges up to the last hour and a lot things that we could learn and do better, going forward.

During my final interviews, the program had reached the majority of the goals they had set for their first two releases. The next releases were related to data migration, that is, transferring customer information from old system to new, and not as much about product development, although there were minor design and development activities that would continue indefinitely. The business stakeholder groups began to withdraw from the daily activities and the work was conducted across fewer boundaries. At this point, I decided that there was enough data that covered the different aspects of the program and that the shift in the focus of the work was a natural finishing point for the study as well.

6.6. Discussion of The PrecautionCorp Case

PrecautionCorp and the partners, vendors and other stakeholders presented a complex case of Agile development in a large program that combined both onshore and offshore development in several locations. PrecautionCorp's perspective towards Agile development as a philosophy was pragmatic. The values were seen as an important part of the development, but the very large scale and intricacy forced PrecautionCorp to apply Agile methods differently, depending on the situation. The artefacts supported virtual teams and offshore development activities, but were applied in the collocated office environments as well. The business stakeholder communication was organised by appointed proxies, such as product owners and the business stream iteration manager. The following sections will explore the elements of the case in more detail.

6.6.1. Agile – Realistic and Business Focused

The Agile perspective at PrecautionCorp was enthusiastic but realistic. The members of the program acknowledged that they had to juggle their own Agile methods, the waterfall ways of the vendor and the different Agile approaches their strategic partners had. The offshore Agile was focused on very clear communication between the teams. The program members were aware of the potential issues with globally distributed software development, such as time zone constraints and cultural differences (Holmström et al. 2006).

Several informants explained to me that the Agile development methods at PrecautionCorp were originally introduced by a top manager, who was enthused by the idea of Agile development and wanted to transform the organisation. The manager had championed the methods and managed to convince other members of top management of the benefits of Agile adoption. The Agile coaches were introduced to the organisation to ensure that the transformation was permanent. The gradual Agile diffusion and new hiring practices meant that the perspective towards Agile at PrecautionCorp was based on solid foundations and understanding of the core concepts.

The strategic partners were practitioners of Agile methods but applied their own methods. The interviewees thought the vendor, who provided the product that formed the core of the program, was practicing waterfall development. One of the managers stated that working with vendor was quite traditional:

We need to give our vendors clear requirements to get a cost in a time commitment. You just can't go to another organisation and say we have this project we can't really tell you what it is, we are not going to pay for them until they start working. Give me that commitment and tell me how much you think that will be, it does not work that way, so that is where we constantly come from.

The interviewees and other members of the organisation whom I observed in the meetings shared a similar perspective towards Agile. The application of the methods was meticulous, but the members of the organisation were conscious of the limitations their environment posed to the application of Agile methods. Some of the employees acknowledged that the methods were not applied as strictly as they might have wanted, or that modifications had been made to the ways certain methods were utilised. Special circumstances and project complexity took precedence over strict application of the methods. An exchange between the iteration managers and the program lead during one of the Scrum of Scrum meetings illustrates the friction between two opposing views: strict Agile method application and deadline pressures. The program manager felt that there were too many meetings and not enough time to finish the work that was scheduled for the first release and wanted to cut down unnecessary meetings. Iteration managers were generally on board with fewer meetings but were not happy about the mandate to cut out the retrospective meetings, which they felt were an essential part of Agile. When the meetings were discussed, one of the iteration managers asked in a surprised, even a bit disappointed tone: "No more retros, really?" As the meeting adjourned, some iteration managers were clearly unhappy with the directions that went against the Agile method application. This discussion and further discussions with iteration managers and Agile coaches implied that the employees of PrecautionCorp wanted to apply the Agile values as faithfully as possible, but they were realistic when it came to compromises with the needs of the program, which were occasionally opposing the needs of the 'pure' Agile process. One of the iteration managers described how Agile methods had to be modified to accommodate the stakeholders:

And I know that's not very Agile, but I think part of it was just that it was so dependent on the vendor delivery which wasn't always fortnightly… Like having the Agile practices in place but you have to make it flexible just to fit the programme and having it once a month makes more sense, I think.

6.6.2. Business Stakeholder - Collaboration via Boundary Spanners

The stakeholder configuration in the PrecautionCorp program was established to ensure that different stakeholder groups had the means to collaborate with other parties, even if the collaboration was not necessarily directly between the individuals. All parties had assigned boundary spanners who were responsible for information distribution across the boundaries between the organisations and the different teams and streams. These boundaries between the parties existed for several reasons: the technical teams had different goals and the teams differed by the expertise needed for different types of work and by their means of engagement, virtual or physical; the teams consisting of business stakeholders came from different parts of PrecautionCorp and it was important to provide them with accessible contacts to ease their collaboration with the technical teams, teams they usually would not have much to do with.

There were several types of collaboration arrangements between the teams. The offshore development was arranged according to the task breakdown of the development project: the front-end development team – the team responsible for the visuals and usability of the product – was collaborating with the Chinese partners; the back-end teams, split into several streams, were working on the integration of the product, feature configuration and data migration, and predominantly collaborating with the Indian partners. Neither team was working in isolation: the work of the front-end and the back-end was interconnected. The iteration managers paid special attention to coordination efforts that ensured that the dependencies between the teams did not turn into hindrances. The stream dependencies were regarded as crucial discussion points and significant time was spent in discussing these dependencies during the Scrum of Scrum meetings. Iteration managers and the business teams were also supported by the Agile coaches, who acted as boundary spanners between these two groups. The Agile coaches ensured that the Agile methods were understood by the business stakeholders and monitored the application of the methods in the technical teams.

The interviewees had mixed feelings on the boundary spanner collaboration model. The iteration managers were mostly well-informed on the program events and knew where to get information if needed, but the business analysts and testers were unclear on what was happening in the other streams. On the other hand, they thought that the information they were given was sufficient and their product owners were accessible if they had any business requirement related issues. One of the iteration managers explained how her team's product owner was the main conduit for the information from the business stakeholders:

…we have our product owner. Basically, she is our middleman to engage all the business people, so she has to transfer the information to us, but at the same time, I know that in release one, we got a lot of feedback that when the system is rolling out, and then they have no clue how the system work, and looks like it's hard for them to what they're supposed to, like that. Normally, they would on the spot. They would give the feedback, and then we'll capture them, so within we're able to make it, we will raise a tag, and then make the change. If we are unable to do it, our product owner will explain it to them {the business stakeholders}.

The business stakeholders were satisfied with the level of information they were given through their iteration manager. Their main concern was around balancing their own stakeholders and getting the required information from the departments they represented. One of the business stakeholders explained his role:

My job is to look after the investments on a day-to-day basis for the pension trustee. Basically, that involves being the product owner on the investment side, so the go-to person. I do regular reporting to management and to the trustee on the underlying assets of the pension fund. I do the relationship management with investment managers who manage the money. I do research on new investment strategies, do recommendations based on demand, or advisor feedback et cetera on new strategies for the fund.

The vendors had a similar boundary spanner arrangement in place as well. The teams that were more involved with the product development had stronger connections with the vendor organisation. One of the iteration managers explained how her team was collaborating with the vendors:

We have someone from the vendor, who's got a lot of product knowledge. He's embedded in our stream as well and he sort of acts as a BA, as whatever we need him for. As well as configuring the things within the product that we need him to do, say enabling the triggers for the {client} comm{unication}s {in the product}, extracting the XML files, doing a bit of … that sort of stuff.

.

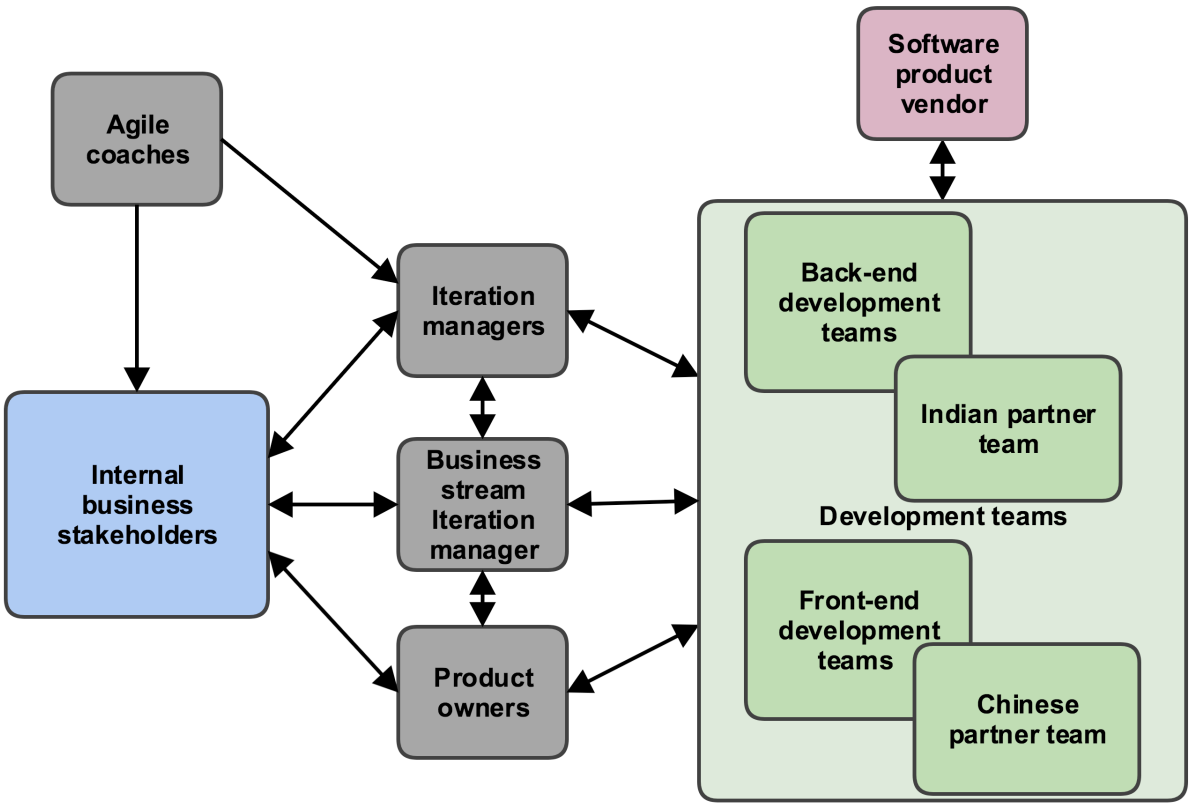

Figure 18. Boundary Spanning Configuration of PrecautionCorp

The boundary spanner stakeholder collaboration model of PrecautionCorp is illustrated in ***Figure 18***. The boundary spanners – iteration managers, business iteration managers and product owners – spanned the boundaries between the PrecautionCorp business stakeholders and the PrecautionCorp development teams. The technical teams and the boundary spanners affiliated with the technical work are marked in green. The business stakeholders, non-technical members of PrecautionCorp, who also worked on the program deliverables, are marked in blue colour. The other two stakeholder groups – the vendor and the coaches – are marked in red and purple respectively

6.6.3. Artefacts – Establishing Control

The primary use of the boundary objects in the PrecautionCorp program was to ensure that everyone was informed, that no information was lost in translation or because of misunderstandings, and that the iteration managers remained in control of the expectations of the work assigned to their streams. Daily meetings between the teams and weekly meetings between the iteration managers were held with the help of videoconferencing tools. Videoconferencing supported non-verbal communication cues: team members could see each other and interpret body language better than with text-based collaboration. Results of these meetings, iteration planning meetings, daily stand-ups and retrospective meetings were diligently stored in virtual document management tools, as well as on the physical wall in the form of Post-it notes.

Application of artefacts other than videoconferencing and document sharing was driven by the context of the work. The streams that were working with the product back-end had less need for wireframes or visual designs but used architectural plans and other blueprints instead. The front-end team, who did have significant visual components in their work, utilised wireframe versions of the web pages of the product in their communication between the offshore developers in China and the business analysis people and testers in Australia.

Adjustments to the processes were constant. One of the iteration managers described another iteration manager:

…I think the previous iteration manager set it up in a way that adhered to Agile practices...I didn't think it was working, and one of the main things was the way the sessions were running… I was used to having elaboration sessions with a lot more detail and a lot more pre-planning… Eventually I worked for the team to work in a better way, a method of elaboration session to get the testing more involved… Through feedback I think we figured out a way to make the elaboration session more thorough and detailed and then eventually the business analysts started writing things in [task cards], they knew to attach requirements and documents to [backlog tool]. Have the walkthroughs, include the business…

In addition to the contracts that were in place between the vendors and the strategic partners, the program structure also meant that the front-end team was internally contracted to work for the program and there was an internal contract in place. This was not uncommon for the front-end team: PrecautionCorp often 'hired' parts of their internal IT development to work on large programs such as the case program, in addition to their day-to-day IT development.

The final parts of the puzzle between the different streams were the environments: development, testing, staging, production and the duplicate versions of these environments. Each stream had their own testing and development

environments but other environments were shared across the program. One of the iteration managers explained which environments her iteration was associated with:

Environments are kept – we have two environments for keeping different versions of the system for the different releases: we call them Week 1 and Week 2. Then we have a user-acceptance-testing environment, we have a development environment. So we've got four environments.

Different versions of the environments were maintained to ensure that the teams who were working in the same environments had access to the data that was appropriate to the stage of the development they were in. Some streams were further ahead in their work and required advanced testing data; other streams were still operating with an older data set. The iteration manager explained the rationale behind the separate environments:

So you know, that's why you need to be very clear about what, what release these products are going in. So they ... in the same environment in the product, but they come from a different environment. From a testing point of view. And the reason why we have to have two separate ones, is because we have to, because normally what would have happened is we would have done release two and release three separately. But because we're doing it in parallel, that's why we need the two environments, so we understand exactly where the work is progressing to.

The case shows that the artefacts had several purposes: the physical and virtual walls, the document storages and contracts were applied in order to ensure the alignment of the expectations of different stakeholder groups, to coordinate the work effort between the streams and to provide visibility on the program progress to all stakeholders. The contracts were especially applicable when shared goals were established at the beginning of the program. The agreements between all different parties formed the baseline understanding of the program and provided means for PrecautionCorp to control their partners and vendors as well.

The visual artefacts, such as wireframes, visual designs, the system itself or the architectural plans of the system (a kind of prototype), provided a reference point for the exploration of the goals shared by the different parties. These intermediary versions of the product enabled discussions where the stakeholders could identify what the different aspects of their problems were, or in this case, the different aspects of the product.

Similar to the other two cases, the environments enabled the development work, which was distributed across several locations, countries and continents. The access to the different environments was granted on the basis of the needs of the streams. By controlling access to the environments, the PrecautionCorp teams were able to maintain control over the versions and data that were being applied during the development efforts that took place concurrently. The environments also had an enabling role in the alignment of expectations, as the product was shared across the teams and stakeholders, and stakeholders could monitor and observe the product development in real time. The Agile activities performed by the organisation, the artefacts and the purpose of the artefacts are summarised in ***Table 20***.

Agile Activities	Supporting Artefacts	Purpose of Artefacts
Daily Scrum meetings Scrum of Scrums Iteration planning meetings Showcases Requirement elaboration sessions Retrospective meetings Prioritisation meeting Testing	Physical Agile walls Virtual walls Document storage Contracts	Alignment of expectations Coordination of work efforts Visibility of progress Alignment of development efforts Establishing control and shared goals
Prototyping	Wireframes Visual design The system Architectural plans	Identification of different aspects of the problem Exploration of the shared goals
Deployment	Environments	Facilitation of the development Alignment of expectations Control of data access

Table 20. Summary: PrecautionCorp Agile Activities and Artefacts

6.6.4. Conclusion

The case of PrecautionCorp presents a study of application of Agile methods in a large-scale program with numerous stakeholders from multiple organisations. The Agile perspective held by the interviewees was a pragmatic and realistically tailored set of program environments. The adapted Agile methods were seen as the key facilitator of effective collaboration between the parties. The business-benefit-focused perspective towards Agile allowed PrecautionCorp to apply its own set of Agile methods, without having to compromise on the methods excessively in the environment where the strategic partners and vendors followed their own methods and each party had different approach towards Agile development. The stakeholders were connected via boundary spanners – people who were assigned to ensure the flow of information and collaboration. The boundary spanners ensured that the artefacts that were designed to aid the collaboration were applied to their full extent, in order to capture and convey the program information across different organisations' boundaries. The use of different Agile artefacts created a sea of information which program participants had to navigate. The boundary spanners – iteration managers and Agile coaches – supported the navigation and guided stakeholders through the program.

The case of PrecautionCorp presents a study of a complex program in the financial industry, a program that could have gone awry in many different ways. Literature provides several examples of similar cross-cultural and global projects which have ended with less satisfactory results. Either collaboration is an issue (Subrahmanian et al. 2003), methods are misapplied (Paasivaara et al. 2012) or stakeholders are troublesome (Lehtinen et al. 2014). In the light of these potential pitfalls, I find that the PrecautionCorp program demonstrated maturity and understanding of both

the Agile methods and overall stakeholder management. The Agile coaches contributed to the maturity of Agile practices and understanding of Agile methods across the department boundaries in the organisation. The PrecautionCorp case is an example of how larger organisations can successfully change their project development culture and embrace Agile methods.

The three previous chapters have presented the case studies that I have analysed in order to find the answer to my research questions. Each case has shown a different perspective towards Agile development, different stakeholder configuration and different application of boundary objects; however, in order to derive a holistic view of Agile methods, these individual accounts have to be synthesised and compared. The next chapter will compare and contrast the three cases and present the theoretical framework that is derived based on this synthesis.

7. Synthesis and Discussion

"In my previous life, when I was a consultant, trying to convince people to try Agile, the hard part for me was what I call 'The Path of the Saints'. This is a thing in the Catholic Church; you need three miracles to become a saint. You need to prove three miracles. Whenever I join some new project as a consultant, I try to do three miracles and then say, 'Now you believe what I'm talking about? Now you believe in me?'"

– Technical team lead, Extreme Inc.

7.1. Introduction

The three case studies, described and analysed in the three previous chapters, have provided an abundance of insights into the world of Agile development in very distinct settings; however, in order to create a holistic response to my research questions, a synthesis is required. This synthesis and discussion chapter will address the three questions one by one. First, I will discuss the different Agile approaches and how these approaches have influenced the collaboration. Next, I will analyse the differences and similarities between the different collaboration configurations. Then, I will address the application of boundary objects and identify three distinct categories of objects that provide a theoretical frame for their application.

Finally, the discussion section concludes by presenting the theory of Agile engagement, a holistic theory that answers the research questions and presents a way that the three aspects of the research are interconnected and how these elements influence the Agile development process.

7.2. The Differences in Collaboration Under Different Agile Approach

In the literature review, I established that there is a spectrum of perspectives towards Agile development methods, which can be observed by analysing how both the researchers and their case organisations view Agile methods. The literature review addressed literature that can be found all across the spectrum, from the philosophical approach to the business-benefit-oriented approaches (see ***Table 11*** for summary of the literature topics). Similarly, the three case studies provide accounts that can be placed along the spectrum of Agile views. The spectrum of different views encompasses the different ways organisations that already have adopted Agile and are Agile practitioners approach their chosen methods and how much attention is given to the values and principles of the Agile Manifesto. The spectrum does not include organisations, which have not adopted Agile methods and view Agile software development from outside the practice. Organisations which are on their path to Agile adoption will eventually fall to a point on the spectrum, but while they are not yet part of the Agile practice, their views on the methods are out of the scope of this study. I am strictly focusing on the differences between organisations who are identified as Agile practitioners by the members of the organisations themselves as well as the outside Agile community. The case organisations, selected due to their strong Agile method application but also due to their differences, present an overview of the Agile perspective spectrum.

In order to distinguish between the perspectives presented in the three cases, I have labelled them: 1) Avid, 2) Inclusive and 3) Pragmatic Agile. Each label attempts to capture the perspective held by the organisations as well as reflects the views apparent in the Agile literature. The Avid perspective towards Agile development presents an approach where the Agile methods are avidly followed and rigorously implemented in the organisation. Avid Agilists are devotedly following the values and principles of Agile Manifesto and the methods' guidelines as originally defined. This does not mean that the Avid Agile perspective is not open to modifications if the need arises; however, the modifications have to fit the Avid Agile approach and not compromise the integrity of the Agile methods applied. Agile 'purists' – strict advocates of Agile methods – would see every Agile organisation adopt the Avid perspective with their chosen method.

The Inclusive Agile perspective refers to a view that embraces Agile methods as described by the Agile advocates, but does not shy from modifications which might enable application of the methods in environments that stray from the ideal collocated development spaces. The Inclusive Agile perspective is inclusive when it comes to different Agile methods applied by different organisations. Rather than focus on single method, the perspective allows collaboration parties to mix the Agile methods and amend them according to the needs of the parties involved.

The Pragmatic perspective towards Agile development is the most lenient when it comes to methodical application of the methods. In an environment where the Pragmatic perspective is adopted, one could see a first team practicing very rigorous Scrum, whereas a second team could be in their early stages of adopting Agile methods and a third team could be practicing traditional waterfall management. In the Avid approach, such combinations would not be possible and organisations with the Inclusive Agile approach would feel uncomfortable as well.

7.2.1. Avid Agile of Extreme Inc.

The first case study, the daily Agile activities at Extreme Inc., presents a case of the Avid Agile approach. Agile development was primarily seen as a philosophy and as an important part of the engineering culture of the organisation. All the Agile activities performed at Extreme Inc. were designed to foster the Agile culture and represent the values and principles of the Manifesto. Some might call the employees of Extreme Inc. 'Agile purists', although not everyone I interviewed from the organisation expressed their views on Agile quite as strongly. However, the consensus amongst the interviewees was that the way Agile methods were practiced and enforced in the organisation was an important factor in creating the collaboration environment with the stakeholders and in enabling the product development activities. Selection of Extreme Programming as the main method, and especially the rigorous practice of pair programming, is anomalous, even amongst organisations with lengthy history of Agile application; recent studies point towards a declining interest in the method (VersionOne 2017, Dingsøyr & Lassenius 2016).

Similar views are presented in the small sample of empirical papers where the Agile perspective could be described as Avid. Unsurprisingly, most of these papers seem to favour Extreme Programming over other methods (Sharp & Robinson 2004; Chong 2005; Vidgen & Wang 2009). A notable difference between the Avid perspective of the Agile activities and the other perspectives was the strict focus on a particular method – pair programming – and the willingness to organise all other activities to support the methods. Every aspect of Extreme Inc. was catering for the

Agile methods, from human resources to the office design. Collaboration between the business stakeholders followed the same Agile method guidelines as the development work. The feedback loop between the product development teams, product managers and end users was kept short, with the practice of continuous delivery and tight collaboration between each party involved in the end user interactions.

The Avid perspective chosen and carefully maintained by Extreme Inc. is an example of the most devout followers of the Agile Manifesto and of an organisation whose members have a philosophically oriented view, falling on the extreme end of the Agile perspective spectrum. While one could safely speculate that most organisations would fall into less absolutist ends of the spectrum, Extreme Inc. is a rare example of an organisation willing to implement and maintain the most rigorous methods and maintain faithful to the ethos of the Manifesto. Most organisations have adopted the less stringent perspective. Next, I will detail how the Inclusive perspective towards Agile development manifested in practice.

7.2.2. Inclusive Agile of Escapade and Carmine

The second case, a virtual Agile development project with Escapade and Carmine, is an example of an Inclusive perspective towards Agile. The Agile methods were applied to foster inclusion. There were several reasons why the organisations were less Avid and more Inclusive in their approach.

First, the employees working for Carmine knew that by joining a virtual organisation, they would not be able to apply the most stringent Agile methods that the catalogue of methods offered, such as Extreme Programming. Instead, they applied the Scrum methods (Schwaber 2004), which are often suggested and applied for virtual and globally distributed projects (Sutherland et al. 2008; Modi et al. 2013; Paasivaara et al. 2012; VersionOne 2017). The employees of Carmine were satisfied with the level of Agile method application and the flexibility and freedom that the structure of the organisation provided them; the members of Escapade shared this approach.

Even though the employees at Escapade were mostly collocated, they had experimented with Scrum and Kanban and were pleased with the improvements in project speed and quality that the adoption of these two Agile methods had provided. However, they were not planning on implementing more rigorous Agile methods internally or with their vendors. The level of 'agility' was seen as satisfactory. Both organisations saw that it was more important to apply the extant Agile methods, mainly Scrum methods, to the extent that was possible and focus on the tasks at hand rather than process improvement. Customer inclusion, an aspect of Agile that is often mismanaged (Hoda et al. 2011), was the key driver of the methods' application. Literature discussing other virtual, globally distributed projects shares the Inclusive perspective (e.g. Holmström et al. 2006; Korkala & Abrahamsson 2007).

By holding an Inclusive perspective towards Agile development, the organisations were able to facilitate an environment where Agile methods were implemented in virtual settings. Members of organisations with a strict, Avid Agile perspective would have felt uncomfortable with the levels of amendments Escapade and Carmine had implemented to facilitate their collaboration and product development. The difference between the Avid and Inclusive approaches was also evident in the way the feedback was obtained and implemented. Where the whole organisation of Extreme Inc. operated as well-oiled feedback machinery, the feedback between Escapade and Carmine was more sporadically acquired and reviewed. The intermittent deliveries of the new versions of the

product had to be communicated across the project stakeholders and the boundary spanners had to ensure that progress was made.

This does not mean that the Agile perspective was lacking in rigour. Even though the collaboration was centred on key members of the Escapade and Carmine development team, the other project stakeholders, for example the testers, were flexible and willing to engage whenever their input was needed, as customers should in Agile development projects. The Inclusive approach fostered an Agile environment which was as close to the spirit of the Manifesto as was possible, due to the restraints posed by the virtual nature of the project. It should be also noted that Escapade had made a conscious decision to engage in outsourcing activities and working with a vendor organisation. Had they chosen to build their product in-house, the perspective towards Agile might have been different or more likely; the perspective would remain unchanged. The circumstances of the project were not the main determining factor when it comes to the perspective on the Agile but the sum of the individuals' views practicing the method and shaping the views inside their organisations.

If Avid Agile is at one of the extreme ends of the Agile spectrum, the Inclusive Agile perspective falls in the middle. Lastly, the Pragmatic perspective, at the other end of the Agile spectrum, is discussed.

7.2.3. Pragmatic Agile of PrecautionCorp

The third case study, PrecautionCorp program, is an example of a Pragmatic Agile perspective. The practitioners at PrecautionCorp knew that their environment was complex and chaotic and they were applying Agile methods in order to be ready for changes and chaos, or as described by Goh et al. (2013, pp. 749), the "focus is to develop capabilities that can flexibly respond to unpredictable project uncertainties stated above in order to meet the urgency to complete the project and to develop capabilities that allow learning from that experience." The more senior members of the organisation, who I interviewed, told me that they had learned over the course of their careers that the Agile approach was more suitable for the programs and projects they were conducting and the support for Agile adoption from the upper management of PrecautionCorp was appreciated. The Agile coaches at PrecautionCorp had created their own interpretation of Agile, specifically tailored for PrecautionCorp, and the organisation applied this version in order to facilitate collaboration in their program with Agile and non-Agile stakeholders. Many members of PrecautionCorp had experience in non-Agile organisations and they were happy to adopt the PrecautionCorp way and were not advocating for radical changes in the methods' application. In addition, the more junior members, recent graduates and other employees who were new to Agile, told me that the PrecautionCorp model was striking a good balance between control and flexibility of the Agile methods' application and focused on the right aspects of the development – interactions between the different parties (similar to Martini et al. 2016)

Unlike in the Inclusive approach, where the Agile method choices are made between very rigorous adherence to the selected Agile method and flexibility and involving everyone, the Pragmatic view has to choose between rigorous application of Agile methods and controls of stakeholders (Lee & Xia 2010; Goh et al. 2013). The Pragmatic perspective towards Agile acknowledges that Agile methods are generally well suited for software development projects, but amendments and compromises to the most extreme methods are needed when the project size and/or the number of stakeholders grows. The perspective was further refined by the Agile coaches, who championed the

specifically modified Agile methods applied at PrecautionCorp. The modifications were designed, among other things, to accommodate the diverse set of strategic partners, who practiced their own variants of Agile methods; however, even the modified methods were not tailored to fit all possible cases. In the case program, PrecautionCorp had to collaborate with a vendor organisation, which was more traditional in its software development methods. A very Avid perspective towards Agile, or even an Inclusive view, would not have been a fruitful way to ensure mutual collaboration with all the stakeholders. Clashing opinions of what constituted Agile development and how work should have been conducted were evident even in the very Pragmatic environment which fosters the highest tolerance of deviation from the Agile Manifesto values and principles. However, the Agile coaches and other program stakeholders were aware of the method differences and were willing to accommodate each stakeholder group to the best of their abilities. This openness to accommodation is a key distinguishing character of the Pragmatic perspective and enables collaboration across the boundaries created by method differences.

Each development stream was applying a variation of the Agile methods, depending on the needs and requirements of their technical domain. User interface teams who were working on the front-end development resembled more closely the classic Scrum method application, whereas the business stream, which worked mostly on other types of work than software development, had the most tweaked variant of the Agile method. Nevertheless, the organisation was clear on its Agile development utilisation message and on the application of the Agile methods, constantly supported by the Agile coaches and the upper management.

Table 21 on the next page summarises the organisations observed in the three cases and the views they hold on Agile development. This distinction does not imply that one view is better than another. Adopting a certain philosophical view on the world does not necessarily make software development more or less successful, but acknowledging differences between the views provides a lens for understanding why organisations collaborate and apply boundary objects and Agile methods in the ways they do. It would also be too simplistic to state that the Agile perspectives are only borne of circumstances the organisations operate in. The differences observed in the Agile perspectives are result of both the circumstances the organisations as well as strategic decisions taken by the stakeholders in the organisation.

In all three cases, the implementation of Agile development methods has been a deliberate decision, either made in the very early days of the organisation (e.g. Extreme Inc.) or during an organisational change (e.g. PrecautionCorp). The Agile perspective of the organisation can be influenced during the adoption of methods, as evident by the Carmine case where Agile was introduced and successfully implemented into form that would resemble Avid perspective if not for the openness to collaborate with virtual organisations. However, it should be also noted that it would be difficult for an organisation the size of PrecautionCorp with lengthy history of waterfall projects and complex offshoring partnerships, operating in the finance sector, to adopt an extreme position and align with the Avid view of Agile. Organisational change that had taken place at PrecautionCorp, as discussed with the informants, was still ongoing and the practices were actively promoted to stakeholders. Yet, this transformation should not be seen as an impossible task; Extreme Inc., a sizable company, has managed to operate with the strictest possible variant of Agile in the same finance industry and Carmine has been able to maintain Agile rigour in fully virtual and distributed environment since the company inception.

Case	Agile Perspective	Agile Perspective Discussed
Extreme Inc. Avid Agile	Agile approach is by the book, following the guidelines of Extreme Programming closely. Product development based on the guidelines of Extreme Programming and the pair programming method. The product development teams are the focus of the organisation; other stakeholders are organised to support the development team. Feedback cycle is short; continuous delivery ensures that feedback from all parties is quickly obtained, assessed and implemented	Discussing the Agile development at Extreme Inc.: *I thought, knowing how like how we had been Agile back in the banking days... I thought we were pretty close to being Agile. So in terms of the amount of real hard-core development and Agile...But, when I met these guys, I thought: 'They're not messing around. These guys are awesome.' They are doing everything; everyone is working actually in a way that, back in London, I was saying we should be doing.* – Extreme Inc. team lead
Escapade-Carmine Inclusive Agile	**Escapade:** A product development organisation, which has adopted an Agile approach, which is a mixture of Scrum and Kanban methods. The organisation has gradually transformed from waterfall to Agile throughout the years. **Carmine:** A consulting organisation that has applied virtual Scrum method across all projects. The organisation was founded on the premises of Agile development as the default method. **Both:** Case project development following Scrum method, in the virtual collaboration environment.Feedback is obtained from Escapade stakeholders. The length of the feedback cycle varies depending on the development stage.	Discussing Agile: *I think it's ... It makes a lot of sense. If you look at Agile as opposed to what, simply being Waterfall, for example Waterfall is stupid. Products take so long to deliver, that if you use something like Waterfall, by the time you get to the end, your requirements and your scope change. There's no point locking in scope and never changing it, because at the end of the project you've got something that you'll be using. Agile is a necessity: you need to be able to change your requirements and your scope at a moment's notice. If you keep up with needs, you need to be able to reprioritise.* –Escapade DevOps manager
PrecautionCorp Pragmatic Agile	Transition to Agile as top-down initiative. Agile coaching and guild of Agile coaches support the application of the methods. All strategic partners apply Agile method but vendors might not. Development teams are organised according to the Scrum method; larger gatherings follow the guidelines of a modified variant of so-called Enterprise Scrum (Schwaber 2007). Feedback is sought first and foremost from business stakeholders, product owners and other internal stakeholders. Feedback cycle varies on the stakeholder configuration.	Discussing experiences as part of the hiring process at PrecautionCorp: *I hire on people who have Agile experience and the right mindset. I need just ask a few questions and see if their language is not right. So people, who use languages like: 'I manage, I manage, I deliver'. So very singular sentences. 'I, I, I, me, me, me'. That's not what we are looking. What we are looking for, are teammates. 'We, my team, our team collaborated on blaa blaa blaa. We delivered, it was fabulous to see the team come together on this delivery and the best thing that we learned about this blah blah blah'* – PrecautionCorp iteration manager

Table 21. Perspectives on Agile

Next section will discuss in more detail how the different organisations applied their Agile methods in the collaboration efforts between the different stakeholders and how their perspectives influenced it.

7.3. Collaboration Configurations Between the Stakeholders

The three cases illustrate three different stakeholder collaboration models and organisational boundaries. The boundaries between the parties in my case studies existed due to different project and organisation contexts. In order to understand why every organisation, even the strongly integrated Extreme Inc., has boundaries and thus requires boundary objects, we need to return to the definition of boundaries of practice, as discussed by Levina and Vaast (2005) and Carlile (2002), among others. The boundaries exist between the different stakeholders who are grouped into practitioner communities – communities which share knowledge, that is, a set of methods and rules and understanding how the work specific to their community is conducted (Carlile 2002). For example, a person who engages in marketing activities shares a practice with people who engage in similar activities, understand the theories and best practices of marketing and apply tools designed for marketing. These members of the marketing community might have difficulty in collaborating with members of the software development community, since their work objectives and daily work routines are different and lack alignment of goals. Boundary spanners are people that can help these two separate communities of practice to gain mutual understanding (Levina & Vaast 2005). The next section will discuss in detail how such boundaries were mitigated in each case study.

7.3.1. Boundaries Between the Stakeholders

In each case, the stakeholder configuration and application of boundary objects were designed to bridge different boundaries between the stakeholders. This section will discuss the boundaries between the teams and organisations in each case.

7.3.1.1. Boundaries of Extreme Inc.

Extreme Inc.'s model of engagement was a tight integration between the development teams and the business stakeholders. The organisation consisted of the three product development tribes, the fourth tribe of DevOps, and the support functions or business stakeholders. The three development tribes all shared the key method of Extreme Inc. – pair programming – whereas DevOps and business stakeholders were excluded from pairing practices. This crucial difference between the ways of working divided the development tribes from the other two groups. Another difference between the development and supporting functions was the focus of the work. The tribes were developing products, whereas the activities of supporting functions ranged from sales and marketing to auditing and risk analysis. There was a clear distinction between the daily activities of the business stakeholders and development.

The difference in the daily activities between the DevOps tribe and development tribes was less drastic: both were working on technically oriented tasks, but the daily activities performed by the tribes served different functions. Where the development tribes were more focused on the external factors – production of products – the DevOps team ensured that the internal activities of Extreme Inc. were functioning. The focus on internal or external work created tension between the tribes. The product developers wanted to get their features out to the customers with the

least amount of work, whereas DevOps had to maintain processes while simultaneously helping the developers in their frictionless feature delivery.

7.3.1.2. Boundaries of Escapade-Carmine

The collaboration in the Escapade-Carmine case took place between the two organisations. The organisations of Escapade and Carmine differed in several ways. Escapade was first and foremost an eBusiness organisation, not a software development company. Large parts of the organisation were dedicated to commercial activities and even though the IT team was sizable, software development was only one aspect of the organisation. Nevertheless, the software development employees had had a significant impact on the organisation when it came to the Agile perspective. As already stated, everyone in the organisation was at least knowledgeable about the methods, if not applying them in their own work.

On the contrary, Carmine was purely a software development company and everyone was proficient in Agile methods. These two approaches and organisational goals distinguish Escapade and Carmine employees from each other and form a boundary between the two organisations. In addition, the organisations were physically separated and required careful planning of collaboration that would not be hindered by the physical distance.

However, the boundaries between these organisations were not the only boundaries that required bridging. In addition to the daily collaboration between the two organisations, the times of wireframes or product testing required additional boundary-spanning efforts. The technical team of Carmine had to collaborate with additional stakeholders, the internal testers at Escapade. Direct collaboration between the organisations was especially prevalent during the discovery phase, when user testing took place, which was conducted by the UX designer, but later testing efforts were conducted via the product owner, who applied a set of collaboration tools.

7.3.1.3. Boundaries of PrecautionCorp

The large scale of the PrecautionCorp program, the context of the work that spanned several parts of the organisation, and the sheer number of the stakeholder created multitude of boundaries that required bridging.

The boundaries existed between the organisations as well as between internal stakeholders. The two offshore development teams were working on the different parts of the product, focusing on different aspects of development. The difference in work, as well as the physical location in two different countries, India and China, meant that there was a boundary between the local, Australian team who was more focused on the managerial side of the program.

In addition, the local extensions of the development teams – the front-end development team with their Chinese offshore developers and the local back-end development team with their Indian offshore developers – were separated from the business stakeholders.

7.3.2. Boundary Spanners

7.3.2.1. Boundary Spanning at Extreme Inc.

Even though the boundaries of Extreme Inc. were not based on radical differences between the tribes and other stakeholders, Extreme Inc. further strived to reduce the impact of these boundaries in several ways. The responsibility for communication was shared by everyone in the organisation and business stakeholders were not

exempt from application of Agile methods. They had been part of the Agile practices since the inception of the company. The organisation was free from legacy structures and processes and there was never a period of transformation, where a set of traditional development methods would have clashed with the Agile development methods. Everyone who was employed by the Extreme Inc. knew what the environment was going to be from the very beginning.

The combination of the history of the organisation, the application of the methods and hiring practices, emphasis on the engineering culture and personal responsibility all attributed to the collaborative culture. The business stakeholders were as invested in the Agile development as everyone else and there was less need for arrangements outside Agile methods that would ensure collaboration, save the collocation of everyone in the company, which facilitated face-to-face interactions.

7.3.2.2. Boundary spanning at Escapade-Carmine

The tightly integrated, boundary-spanner-less model of collaboration worked well for Extreme Inc. in the XP-driven Agile culture, where the business stakeholders were located in the same office premises as the technical teams. In the virtual environment of Escapade-Carmine, similar tight integration was neither feasible nor possible. To facilitate and sustain collaboration, both organisations appointed several different boundary spanners for different boundary-spanning purposes. These boundary spanners actively encouraged the application of the boundary objects, transforming them from existing, intended boundary objects into boundary objects in use (Levina & Vaast 2005). The main boundary spanner – the Escapade product owner – straddled between the two organisations. The product owner actively took part in product development and was in daily communications with the development team. The product owner was, as Agile proponents advocate (Beck 1999; Schwaber 2004), part of the development team, an embedded customer representative. Their boundary-spanning role meant that they were not clearly part of either community of practice, rather bridging the two organisations and their stakeholders. The CTO of Escapade was acting in a similar role but to a lesser extent. The CTO was responsible for creating the initial boundary objects, such as the contract and environments. On the Escapade side of the project organisation, the other stakeholders were non-technical business stakeholders.

On the Carmine side of the project, the Carmine project manager acted as the boundary spanner who took part in the development and steered the project. The project manager had a light touch on the Carmine development team – the developers and designers – giving them freedom to organise their work and activities as they thought best. Their main goal was to collaborate with the customer – Escapade – and ensure their active presence, rather than manage the development team.

The boundary spanners ensured that the boundary objects were applied and kept up to date; however, there was less enforcement of Agile methods from either side. Escapade, as mentioned in the case description, had applied Agile methods internally for several years and the Escapade business stakeholders had a good understanding of Agile methods, even if they did not apply them daily in their own work. Similarly, the development team of Carmine was very professional in their Agile methods, with years of experience and what felt to me as solid understanding of the application of such methods, their benefits and potential pitfalls.

7.3.2.3. Boundary Spanning at PrecautionCorp

The PrecautionCorp program – a complex set of stakeholders and boundaries – had appointed several different boundary spanner roles. First, the development team iteration managers bridged the boundaries between the technical streams and the teams in their stream. These iteration managers, who came from the technical side of the organisation, collaborated mainly with other boundary-spanning iteration managers, ensuring that the boundaries between back-end and front-end work that took place across the different streams were bridged. Secondly, there were several product owners embedded in the technical streams. These product owners came from the business side of the organisation and engaged directly with the technical teams. Finally, the iteration manager of the business stream was responsible for bridging the gap between other iteration managers of the technical team and the business team, but occasionally also directly between the technical teams and the business team. The business stream iteration manager had to ensure that the business stakeholders, who formed a loosely defined working group, finished their tasks and communicated their work to the technical teams. In the PrecautionCorp boundary spanner model, the developers, designers or other more technical team members rarely were in direct contact with the business stakeholders and vice versa; however, the materials provided by each collaborative party were stored in shared repositories. The members of the organisation who held boundary-spanning roles had control over the storing of these materials and also control over the structures of the repositories. Boundary-spanning roles and activities that these roles entailed were put in place to prevent mismatches of work among the technical streams and mismatches between the technical streams and other parties (Martini et al. 2016).

In addition to the boundary spanners who were directly engaged in the daily activities, the Agile coaches acted as additional boundary spanners, ensuring the application of Agile methods. In addition, they monitored the application of boundary objects, particularly the objects that were created to foster the Agile methods' application, such as the Agile walls or other product development or knowledge-sharing artefacts. This perspective was common in all projects and programs across PrecautionCorp, embedded in the Agile culture of the organisation. The existence of the Agile coaches, who sometimes took the roles of the iteration managers as well, created basic awareness of Agile in all the members of the organisation. The organisation had intentionally transformed from more traditional into Agile and Agile coaches were there to support and maintain the transition for all the members of the organisation.

7.3.3. Stakeholder Collaboration Configurations

The case studies outline three different approaches for stakeholder collaboration. First, there is an integrated model, as presented in the Extreme Inc. case. The mutual understanding of Agile methods meant that there was less need for boundary spanning or designated boundary spanners who would ensure object application (Levina & Vaast 2005). Strode et al. (2012) argue that boundary spanning is not addressed enough by research and that the XP method does not provide guidance on boundary spanning. I would argue that applying XP reduces the need for boundary spanners or eliminates it. In the Extreme Inc. case, the role of boundary spanners was not delegated to specific employees but spread across the stakeholder groups. The employees adhered to the strong Agile corporate culture of Extreme Inc. There was no reason to appoint anyone to specifically enforce the application of Agile boundary objects; the responsibility was shared by everyone throughout the organisation, as advised by the Agile proponents (Highsmith

2002). The business stakeholders were responsible for their own understanding of the messages the boundary objects were conveying, without constant advocacy from technical members of the organisation.

Second, the case of Escapade and Carmine presents a peripheral collaboration model, a configuration, which allows stakeholders to always exist at the periphery but not necessarily be directly approached too often. The organisations maintain a strong relationship via some selected boundary spanners and their skills in communication and in application of boundary objects. These boundary spanners not only ensure collaboration (Levina & Vaast 2005), but also alleviate the differences that arose from the customer-vendor relationship between the two organisations and from the differences in experience in Agile method application, by fostering an environment of mutual trust (Levina & Vaast 2008).

Third, the PrecautionCorp configuration of collaboration relies on boundary spanners that ensure that communication and collaboration between all different stakeholders persist. A similar model for collaboration has been recommended in literature discussing large, distributed projects and has been applied in a similar setting (Sutherland et al. 2008; Paasivaara et al. 2012). The virtuality of the circumstances created the need to rely more on the collaboration artefacts: the boundary objects. This in turn created a heightened need for boundary spanners as well. The more complex the project, the more boundary spanners were required; and complexity added additional levels of boundaries that had to be spanned in order to ensure collaboration and guarantee that designated boundary objects became boundary-objects-in-use (Levina & Vaast 2005).

Case	Boundaries	Boundary spanners	Stakeholder configuration
Extreme Inc.	Technical teams who pair program – DevOps team Technical teams – business teams	Internal development: All members of the organisation External stakeholders: Product managers	Integrated
Escapade-Carmine	Escapade – Carmine The development team – internal testers	Escapade product owner Escapade CTO Carmine project manager	Peripheral
PrecautionCorp.	The Agile teams – business stakeholders Technical streams – business stream PrecautionCorp – strategic partners PrecautionCorp – vendor PrecautionCorp – product vendor	Technical stream iteration managers Business stream iteration manager Product owners Vendor representatives Agile coaches	Boundary spanning

Table 22. Stakeholder Collaboration Configuration

The cases, boundaries, boundary spanners and the type of stakeholder collaboration configuration are summarised in **Table 22**. The next section will discuss how the boundary spanners applied boundary objects to support their boundary crossing.

7.4. Three Types of Boundary Objects

The case study findings reveal that three types of boundary objects are critical in facilitating collaboration in Agile software development: (1) infrastructural, (2) projective and (3) process boundary objects (Zaitsev et al. 2016). The three categories are created to theorise on the boundary object application and address the second research question: **How are boundary objects used in collaboration between these stakeholders?** By mapping the objects together in these three categories, we can analyse how the objects are applied to support the collaboration effort, but also *why* the objects were applied in these cases.

The three object types – infrastructural, projective and process boundary objects – are categorised by their application and function. First, infrastructural objects facilitate collaboration before the commencement of the project, and its articulation forms the basis of the project's existence (Nicolini et al. 2012). Without an established and mutually recognised infrastructure, a project would simply not exist. Infrastructural objects provide a foundation that enables the emergence of other object types in the latter phases of the project lifecycle. As the development progresses, some infrastructural objects such as contracts or environment recede into the background when a common understanding between the organisations has been established and the development work is being enabled. However, if breakdowns occur, the objects are brought back to foreground (Star & Ruhleder 1996).

Second, the projective objects – wireframes or prototypes and visual designs in the project studied – are used to establish a common understanding of the end goals of the project. Poppendieck and Poppendieck (2003, pp.27) assert that "a prototype synchronises the efforts toward a well-understood short-term goal" and that they "provide a focal point around which cross functional communication can and must occur". This is in line with my findings, which suggest that projective objects may undergo several rapid and significant changes to accommodate ongoing shifts in the discussion about the project requirements.

Process objects, on the other hand, are used to capture the information discussed between the collaborating parties in a process informed by projective objects. The information stored in the process objects reflects the information in the projective objects and influence how the projective objects evolve over time. Customers and vendors are able to use projective objects as a sounding board for new ideas and customer feedback (Ciriello et al. 2014). The requirements elicited by the application of projective objects are then recorded as user stories in the process objects (Cohn 2004).

7.4.1. Functions of Boundary Objects

There are several reasons why organisations have to apply the different boundary object categories. Literature shows that different Agile methods focus on different artefacts (Kuhrmann et al. 2013); however, the differences between the methods are not the only reasons why I observed differences in the application of artefacts, boundary objects.

Infrastructural objects are typically crafted at the beginning of the project in order to inform project participants' shared understanding of what the project will encompass, shape their interactions and guide their practices throughout the project (Koskinen & Mäkinen 2009). In the three case studies, there were several examples that fall into this category. Contracts, regulatory guidelines and software development environments were applied in different organisations in different ways, but they share the same rationale for the application: enabling the foundations for the work and enabling the application of the other objects that supported other collaboration activities.

Despite their foundational role, infrastructural objects are not necessarily static. More specifically, my study suggests that the creation of infrastructural objects should be a collaborative effort so as to create flexible and suitable objects that would best suit the needs of an Agile project. Star and Ruhleder (1996) argue that infrastructure needs to link, but also shape and be shaped by, the conventions of the communities of practice. In the three case studies this would mean a mixture of the conventions of the Agile communities, the IT industry and the financial sector. For example, the financial industry regulation documents are standardised forms (Star & Griesemer 1986): boundary objects that are standard for every party who is taking part in an endeavour. The industry regulations are non-negotiable, standardised forms that everyone who wishes to participate in the industry needs to adhere to; however, the conventions of an Agile development environment require flexibility and mutability from all boundary objects. Rigidity of infrastructural boundary objects can potentially hinder the Agile development process with overhead activities, to the point that the Agile environment begins to resemble a traditional, control-centric project environment (Conboy et al. 2010; Cao et al. 2010). This was a risk acknowledged by several informants across the three case studies, but such instances were not reported.

The projective boundary objects provide a preview of the project goals. They are used to convey the desired outcomes of the project – the end state of the system – relative to its current state. They include different visual representations of the developed system, such as wireframes, visual design documentation and the actual functional system (i.e. the work-in-progress). Here, the wireframes, visual designs and system versions were used as prototypes that facilitated the discussion of the project goals in all three cases (Carlile 2002; Poppendieck & Poppendieck 2003).

Projective boundary objects provide material for feedback, captured in process boundary objects. They tend to be malleable because they reflect ongoing requirement changes. Some projective objects, such as wireframes and other prototypes, are intentionally incomplete and are used to facilitate iterative feedback cycles. When not applied for feedback gathering but rather utilised as part of the development, the prototypes are applied to share design knowledge between the collaborating parties (Di Marco et al. 2012). Similarly, more complete projective objects, such as the actual system, are released in their incomplete state to elicit feedback from users (Boujut & Blanco 2003; Ciriello et al. 2014).

Desired outcomes are iteratively refined, clarified, and incorporated into functional versions of the system, and this evolution continues until the end of the project lifecycle. The high-level view of the system's main functionalities allows the stakeholders or prototype testers to focus on the 'big picture' and identify more fundamental issues when

they are not distracted by the minor details. The purpose is to create common understanding via prototypes (Bechky 2003), not to be focused on minutiae.

In addition, projective boundary objects help to bridge the division between the stakeholders and/or development teams who are responsible for the product's 'look-and-feel' and the developers who were responsible for the back-end functionalities (Strode et al. 2012). The richness of the information in the artefacts allowed business stakeholders, most of whom came from a non-technical, marketing or customer service background, or the developers, who were front-end specialists, to explore a more complete version of the system and identify whether there was a lack of expected requirement dependencies. By creating wireframes, prototypes or sharing the intermediate versions, case organisations were able to bridge all boundaries listed by Carlile (2002): pragmatic, syntactic and semantic boundaries. The prototypes and intermediate versions made abstract concepts more concrete, more plastic and more robust (Winkler et al. 2014).

Process boundary objects facilitate primary project processes: communication, collaboration and documentation (Subrahmanian et al. 2003; Strode et al. 2012). Containing different types of textual and visual information, process objects convey requirements information and customer feedback (Ciriello et al. 2014), as well convey abstract concepts such as project timelines and progress (Yakura 2002; Boell & Hoof 2015) The process objects in the three case studies include artefacts such as the different Agile walls, intranet wiki pages, document management tools, timelines or other plans, backlog tools, chat tools and feedback notes. In addition to communication benefits, certain process objects have other applications as well. For example, physical walls acted as territorial markers, announcing the areas of the office that belonged to each team and tribe (Sharp et al. 2009). The way the walls were organised and styled reflected the identity of the team, as discussed by Gal et al. (2008). Small paraphernalia that were related to the physical and virtual walls, such as team naming conventions, superhero figurines, avatars and colour schemes, all created a sense of team and a collaborative environment. Even a complete outsider could decipher that the walls were related to the projects or products, from the labels, timelines and user interface sketches posted on or next to the walls, creating common understanding of the project events (Strode et al. 2012). The benefits of applying boundary objects are summarised in ***Table 23***.

BO Categories	Artefacts	Boundary Object Functions
Infrastructural boundary objects	Contracts Environments	Inform project participants' shared understanding of what the project will encompass, shape their interactions, and guide their practices throughout the project (Koskinen & Mäkinen 2009). Link, shape and be shaped by the conventions of the communities of practice (Star & Ruhleder 1996). Facilitate collaboration before the commencement of the project, form the basis of the project's existence (Nicolini et al. 2012).
Projective boundary objects	Wireframes Visual design The system	Facilitate discussion of project goals (Carlile 2002; Poppendieck & Poppendieck 2003). Create common understanding (Bechky 2003) and share knowledge of designs (Di Marco et al. 2012) across stakeholder groups (Strode et al. 2012). Elicit feedback from users (Boujut & Blanco 2003; Ciriello et al. 2014). Communicate abstract concepts in more concrete, more plastic and more robust ways (Winkler et al. 2014).
Process boundary objects	Physical walls Virtual walls Backlog tools Chat tools Document storage Architectural plans	Mark territory and capture team and organisational identity (Sharp et al. 2009; Gal et al. 2008). Create visual representation of time and events (Yakura 2002; Boell & Hoof 2015). Create common understanding of the project events (Strode et al. 2012).

Table 23. Boundary Objects Categories and Functions

7.4.2. Application of Boundary Objects in the Case Studies

Figure 19 summarises which boundary objects were applied and how these objects were utilised in each of the case studies. The top part of the illustration shows the connections between the objects: the infrastructural objects enable (**Arrows 1**) the existence of the two other object types. The projective objects inform (**Arrow 2**) the process objects on what clarifications or changes the collaboration between the vendor and the customer has yielded. These changes should be stored in the process objects as user stories or more detailed technical requirements. The development team can then utilise these stories and requirements and capture this information into the process objects as new or amended functionalities (**Arrow 3**). The middle row of the table describes how these boundary objects were applied by each case study and the bottom row describes why each of these objects was applied in the cases.

In the next section, I will tie the perspectives of the members of the organisations towards Agile, the stakeholder configuration and the application of boundary objects together.

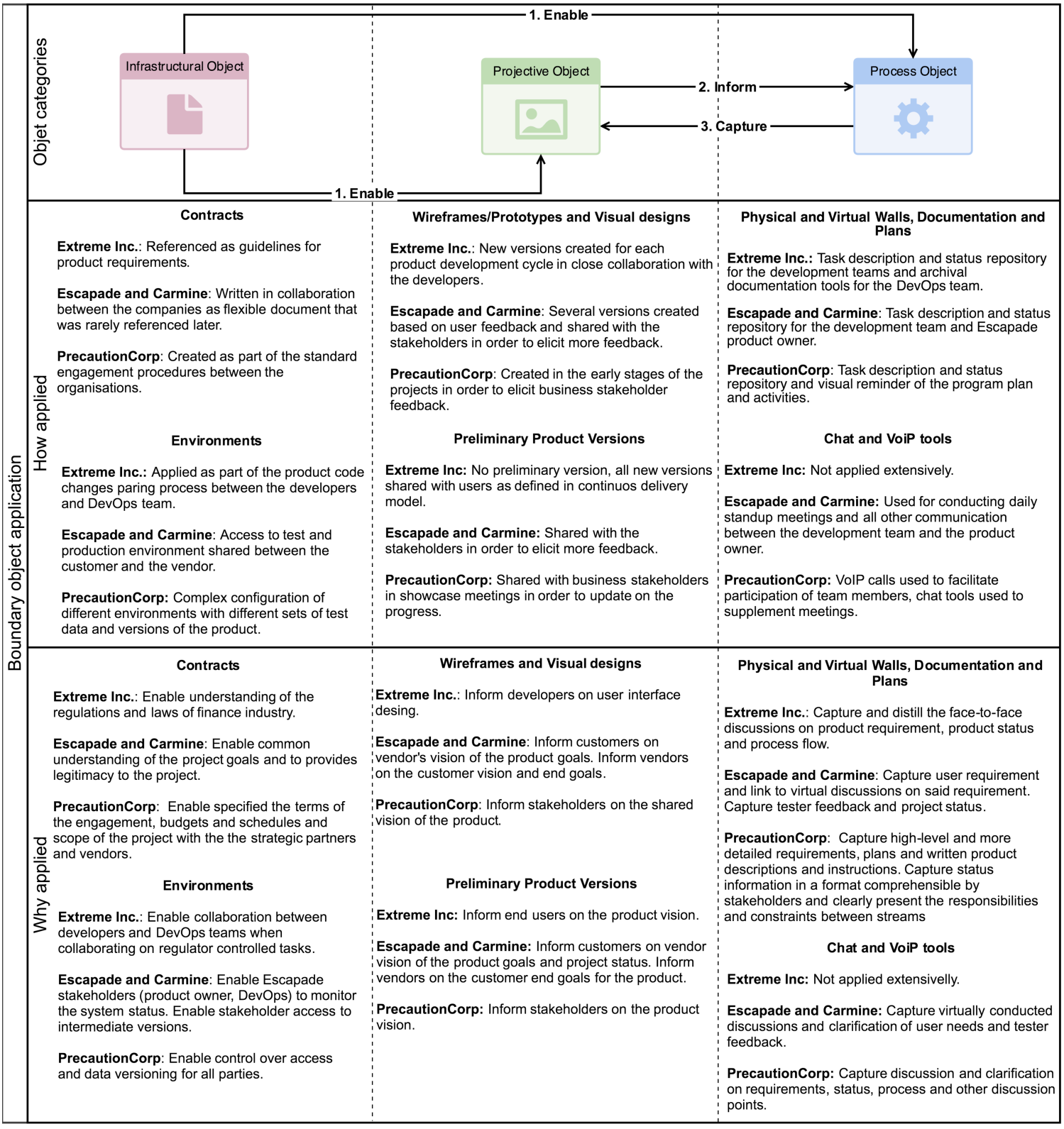

Figure 19. Boundary Objects Application

7.5. Framework of Agile Project Engagement

By analysing the three case studies, I have identified three main elements that impact Agile software development: the **perspective** the organisations have towards the Agile philosophy which impacts the **Agile process**, that is, how the Agile methods are applied in practice; the **configuration of the business stakeholder collaboration**; and the application of the **boundary objects**. In this section, I will analyse the role of the Agile elements across the three case studies and propose a theory of Agile engagement.

7.5.1. Elements of Agile Development

The three cases and the elements of the development projects provide three different permutations of an Agile development environment. The three types of approach towards Agile philosophy (Avid, Inclusive and Pragmatic) form the basis under which the organisations operate and how they interact with their stakeholders. These three ways can be classified into the by-the-book implementation, the flexible implementation and the complex implementation of the Agile development process.

The Avid Agile approach required commitment from the organisation and posed specific constraints on the other two elements. The application of the Extreme Programming method (Beck 2000), which requires extensive face-to-face collaboration and physical boundary objects (Sharp et al. 2009), would not be a good fit for an organisation which would not subscribe to the Avid approach but rather would look for control and business benefits (Goh et al. 2013) or troublesome external customers (Martin et al. 2010). However, the teams at Extreme Inc. ensured that the stakeholder collaboration and boundary object application was in alignment with their chosen methods, which their Agile process followed with by-the-book rigour.

The integrated collaboration environment, where the organisational boundaries were kept low, benefitted from the tangible physical objects that were ever present and provided easy access to information. For example, since the teams and stakeholders were located in the same premises and had low barriers to face-to-face interactions, the teams could also reap all the benefits of the physical Agile walls (Sharp et al. 2009; Strode et al. 2012). The application of the infrastructural objects focused on the collaboration between the development teams and the DevOps team, providing a platform for the code changes and object to work on (Nicolini et al. 2012). The projective objects (prototypes and sketches) were tools for clarifying the ideas of the designers and the end users, and applied for feedback as well (Di Marco et al. 2012).

The Inclusive perspective, exemplified by the project between Escapade and Carmine, ensured that stakeholders were included regardless of the specifics of the Agile methods they prescribed to. The Inclusive mindset meant that the both organisations had members who were willing to sacrifice some of the more stringent Agile principles, in order to facilitate good project results. Overall, everyone was flexible when it came to the daily Agile activities and collaboration.

A collaboration configuration where designated boundary spanners had a large role in the daily communication was supported by an extensive utilisation of boundary objects. The customer organisation was willing to do the best it could when it came to customer participation in the projects – one of the key elements of Agile development (Beck

et al. 2001). The members of Escapade knew that the customer had a large role in project success and their mindset had to be right for the project to succeed (Koskela & Abrahamsson 2004; Kautz 2010; Martin et al. 2010). They were willing to engage in the Agile activities and adopt boundary objects that would further enforce the Agile method application, such as the chat tool or the backlog tool (Gal et al. 2008). The Inclusive mindset of both companies also meant that they were willing to relinquish some degrees of control, in order to empower the boundary spanners to become boundary-spanners-in-practice (Levina & Vaast 2005).

Boundary object application, driven by the boundary spanners and their roles, included extensive application of objects from all three categories. Infrastructural objects, such as the contract, were applied to facilitate collaboration before the commencement of the project, and its articulation formed the basis of the project's existence (Nicolini et al. 2012). As the project progressed into the discovery and development phases, the contract receded into the background when a common understanding between the organisations was established (Star & Ruhleder 1996). Projective boundary objects, such as prototypes, were used to convey the project scope and end goals (Winkler et al. 2014). Process boundary objects were applied to foster communication (Subrahmanian et al. 2003) and to mitigate misunderstanding and conflict (Iorio & Taylor 2013).

The Pragmatic perspective was manifested in a willingness to compromise and foster an Agile environment where business benefits would be achieved, with appreciation and understanding of Agile methods in the organisation without hindering dogmatisms. PrecautionCorp had tailored its own Agile methods and the members of the organisation took heed of the advice given by the early advocates of the methods (e.g. Boehm 2002; Cohn & Ford 2003). It had successfully managed to transform a plan-driven organisation into an Agile one and strove to maintain the adherence to Agile. The members of the organisation – especially the nominated boundary spanners, Agile coaches – were actively encouraging the methods' application. They were aware that only applying a set of methods would not make the organisation Agile; there had to be an element of the paradigm shift as well (O'hEocha et al. 2010).

The Pragmatic perspective meant that PrecautionCorp was able to conduct complex, distributed projects, as suggested by some of the Agile advocates (Sutherland et al. 2008) without compromising the integrity of their 'agility'. Their perspective allowed the designated boundary spanners to both uphold the Agile methods and to maintain levels of control required for the chaotic and complex program (Goh et al. 2013; Martini et al. 2016). The motley collection of Agile and non-Agile processes, followed by the program streams, strategic partners and vendors, were kept in control by allowing liberties when it came to strict Agile method interpretation. The application of the boundary objects by the boundary spanners and the teams was also linked to the Pragmatic perspective. A member of an Avid Agile environment would not accept the levels of documentation that were applied in the PrecautionCorp organisation and would potentially scoff at the idea of Agile collaboration with a non-Agile vendor. However, the members of PrecautionCorp shared a perspective that values business benefits over stringent method application and were willing to engage and push the boundaries of plan-driven organisations with both the method application and boundary object use (Levina & Vaast 2008; Gal et al. 2008).

The discussion of elements that differentiated the three cases are summarised in ***Table 24***.

Element	Case 1. Extreme Inc.	Case 2. Escapade-Carmine	Case 3. PrecautionCorp
Agile perspective	**Avid** e.g. strong advocates of Extreme Programming practices	**Inclusive** e.g. mix and match, positive but non-religious approach	**Pragmatic** e.g. benefits are understood, strays when needed
Agile process influenced by different perspectives	**By-the-Book** Collaboration with team members and stakeholders collocated, daily meetings Product development teams are the focal point, pair programming method applied Feedback obtained from end users	**Flexible** Collaboration between the core members of the team, occasionally additional business stakeholders are engaged Project development team is the focal point, virtual development team, virtual collaboration Feedback obtained from the customer business stakeholders	**Motley** Collaboration between the members of the technical streams and between the different streams, collaboration between the business stream and tech streams Stream iteration managers are the focal point, virtual, globally distributed team collaboration Feedback obtained from business stakeholders, embedded product owners
Business stakeholder configuration	**Integrated** e.g. stakeholders are present, easy access	**Peripheral** e.g. stakeholders are always there but sporadically engaged	**Boundary spanning** e.g. stakeholders are proxies for more stakeholders
Boundary object application	**Enhancing use** **Infrastructural** – enabling the application of methods, e.g. the environments as boundary objects in continuous delivery **Projective** – clarifying ideas, e.g. prototypes and the products applied as boundary objects, reviewed by teams and with customers **Process** – enhancing presence, e.g. Agile walls to support face-to-face interactions, objects used to enhance presence	**Supporting use** **Infrastructural** – creating common goals, enabling sharing of vision, e.g. contractual agreements, shared product **Projective** – creating presence, e.g. prototypes as important boundary object used to create presence **Process** – supporting presence, e.g. communication and collaboration artefacts supporting the discussion around projective objects	**Controlling use** **Infrastructural** – creating strategic alliances, enforcing access restrictions, e.g. contractual agreements between parties, different test and development environments **Projective** – clarifying goals, e.g. prototypes and the product demonstrated to stakeholders **Process** – establishing control, e.g. walls/comms tools to support process, boundary objects used to control project

Table 24. The Elements of Agile in the Three Case Studies

However, it should be pointed out that amidst all these difference between the three case organisations, there were also significant similarities: in their own way, each organisation was dedicated to Agile software development and to their chosen methods. In each organisation, active engagement with the stakeholders was seen as one of the top

priorities in order to achieve the desired end results. There were differences in the detailed implementation of the Agile methods but overall, each organisation followed similar overarching practices to support their collaborative efforts: daily meetings, retrospective meetings, independent software development teams, active and easily accessible stakeholders. Similarly, the application of boundary objects followed similar pathways: the organisations generally used visual objects during the design, supported their collaboration with objects created for Agile working environment and defined their work in objects that would capture the outlines of the endeavours. Nevertheless, even if from a high level perspective, the organisations were similar, there are important differences when individual elements are investigate. Next, I will discuss how the elements influence the Agile development work itself.

7.5.2. Framework of Agile Engagement

The three elements of Agile and the three case studies have provided me with building blocks for a holistic framework of Agile engagement. Without neglecting either human or material aspects, the framework for Agile engagement combines the elements of an Agile environment in a novel way that does not exclude context or complexity.

The framework of Agile engagement is derived from the different views of Agile, boundary object application and stakeholder configuration models, as well as the Agile collaboration process. The three cases present three Agile environments, each of which is unique due to the diverse entanglement of the elements. The framework does not suggest that these are the only three ways to achieve an Agile environment. What it does suggest is that these elements are strongly linked together and influence or support each other. Observing these elements can help to understand the reasons why an Agile project environment is shaped a certain way.

These elements of Agile can be seen as constructs of a model: variables that are derived from both literature and the case studies. The Agile perspective is the overarching theme that resonates in all activities of each organisation. The stakeholder configuration and boundary object application form the Agile structures: the underlying ways the work has been organised in the organisations. The Agile process describes the extant actions the organisations are taking when conducting Agile projects. The process, in very simplified form, consists of the iterative work of development and product release. Together, these constructs form a model that illustrates how the perspective towards Agile development influences the Agile process and how the collaboration process is affected by the Agile structures: collaboration patterns between Agile teams and business stakeholder and boundary object application.

The framework for Agile engagement is illustrated in ***Figure 20***. The figure is divided into three parts. The top of the figure shows the simplified Agile process, consisting of the stakeholder collaboration, product development and feedback examination. This part of the illustration describes, in a very high level way, the activities of the Agile process. Underneath, the two constructs that form the Agile structures – stakeholder configuration and boundary object application – are presented. These constructs are linked to the Agile process with dashed arrows which indicate that they are not deterministic or causal. The dashed arrow describes a relationship between these constructs. On the right-hand side, the Agile perspective is linked with two arrows that indicate the influence of this construct on both the Agile process and Agile structures. The Agile perspective is also linked to the Agile process, as the Agile process can influence the Agile perspective.

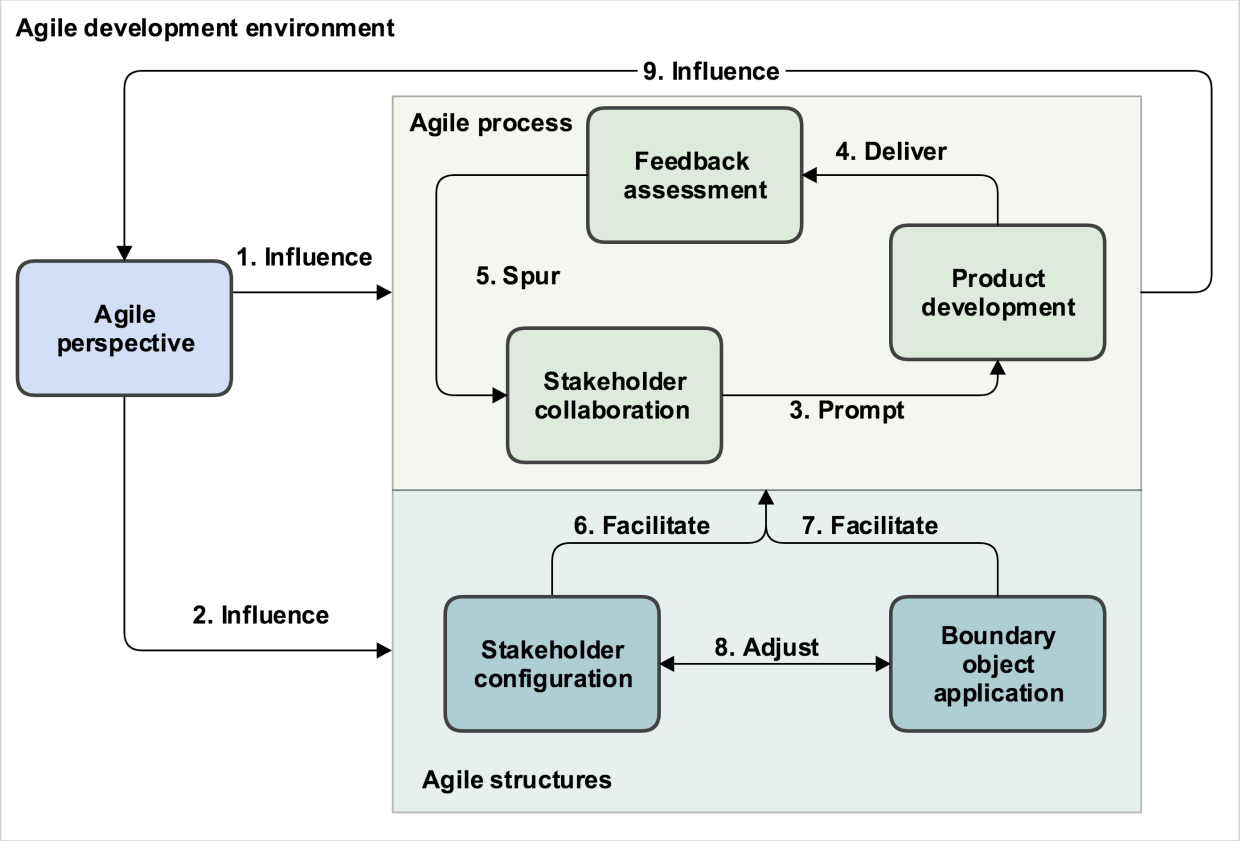

Figure 20. The Framework of Agile Engagement

The perspective, shared across the organisation, influences (**Arrow 1**) how the organisations conduct their Agile development process. The perspective also influences (**Arrow 2**) how the organisations form their stakeholder configurations and how they apply boundary objects. Influencing these aspects of the project has been the whole purpose of the Agile proponents. The Agile methods nudge organisations towards certain types of stakeholder configurations with Agile roles (e.g. Product Owner of Scrum (Schwaber 2004)), best practices (e.g. collocated customers of Extreme Programming (Beck 2001)) and iterative development accompanied with consistent releases (Highsmith 2002). The perspective towards Agile is developed by the Agile practitioners due to their views on the Agile Manifesto and the Agile methods detailed in different method collections, a perspective which is reflected in the overall Agile customs of the organisations. The hiring practices or other external and internal forces can – and sometimes are deliberately used to – shift the perspectives the individuals have towards Agile practices (as suggested, for example, by Beck (1999) and Cohn & Ford (2003)). There is a burgeoning industry of Agile consulting, which is focusing on shifting people's minds from plan-driven development to Agile and from one point on the Agile perspective spectrum to another point (Freedman 2016). However, this change, as with all organisational change, is not necessarily straightforward, nor can one expect that adding more people with a certain Agile view will change the organisation to share the view. The extent of literature discussing the challenges of Agile

adoption speaks volumes of this difficulty (Dybå & Dingsøyr 2008; Hummel 2014). However, organisations (for example, PrecautionCorp) have invested in the role of Agile coaches, who champion the methods, present cases of successful Agile projects, and attempt to increase trust in the methods and make people more accepting of the values and principles. The reflection of the perspective is seen both in the Agile process itself as well as in the Agile structures which facilitate (**Arrows 6** and **7**) this daily process of collaboration, development and feedback (for example, by boundary object application as discussed in Strode et al. (2012) or by suggesting stakeholder collaboration arrangements (Schwaber 2004)).

The daily process can be simplified into three parts: the collaboration process which prompts (**Arrow 3**) (Schwaber 2004; Beck 1999) the development process, which in turn leads to a delivery of an output (**Arrow 4**) (Beck 1999; Cockburn 2004) to parties who evaluate the outcome and provide feedback which is examined by the development, and which in turn spurs (**Arrow 5**) more collaboration and the next iteration cycle (Highsmith 2002). The outputs of the development range from prototypes to complete products, but can also refer to smaller deliverables, such as pieces of documentation, bug fixes or any part of the software development process that needs input from stakeholders (Cockburn 2004). These outcomes are influenced by the Agile perspective, which is in turn influencing the process, as was intended by the proponents of the Manifesto. For example, organisations with Avid Agile perspective, such as Extreme Inc. with Extreme Programming (Beck 1999), would focus on delivering outcomes often and at a steady pace, following their continuous delivery method and other chosen methods as closely as possible to the original instructions, with a by-the-book approach. If the organisation had adopted a Pragmatic approach, as discussed in the Escapade-Carmine case, the outcomes could potentially include more documentation and intermediary planning artefacts: boundary objects that holders of other perspectives might frown upon. The process consists of more variations of different Agile and, possibly, non-Agile methods and forms a motley of methods, boundary objects and boundary-spanning channels. Similarly, research discussing method adoption suggests that organisations modify their approach to better suit their needs (e.g. Paasivaara et al. 2012; Bass et al. 2016). These differences were manifested in the case studies clearly: the employees of Extreme Inc. relied solely on the minimal set of artefacts, mainly the different Agile walls whereas the PrecautionCorp process produced a lengthier list of boundary objects that were applied to ensure the complex program was progressing. Lastly, the Escapade-Carmine process was flexible and accommodating to small deviations from the strict Agile methods, but mostly strived to conform to the ethos of Agile method instruction guidebooks, such as Scrum Guidelines (Schwaber 2004).

The process of collaboration is facilitated (**Arrow 6**) by the stakeholder configuration and the boundary object application (**Arrow 7**). The facilitation of the collaboration by these two constructs is embedded in the numerous Agile case study examples, Paasivaara et al. 2008, Strode et al. 2012 and Bass et al. 2016, to name a few, as well as the many Agile guidebooks (e.g. Beck 2002; Highsmith 2002; Schwaber 2004). The way the stakeholder collaboration is organised allows different interactions between the parties and a constant adjustment between the application of the boundary objects and the stakeholder collaboration configuration (**Arrow 8**). However, the linkage between these two constructs is not explicitly discussed in literature, but discovered while conducting this research. I suspect this is due to the scarcity of Agile research which focuses on boundary objects, as discussed in the literature review section 2.4.

Where the stakeholder configuration facilitates the Agile development process (**Arrow 6**) by providing everyone with access to the right people, the boundary object application facilitates the Agile development process (**Arrow 7**) by providing the infrastructure and artefacts that capture the outcomes and feedback. For example, in Extreme Inc.'s integrated business stakeholder configuration, the boundary objects were applied to enhance the integration. Enhanced integration, in turn, facilitated even tighter integration. For Escapade and Carmine, the configuration of stakeholders was based on the peripheral, but accessible when needed, setup of business stakeholders. The boundary objects were applied to support the access when required. In the PrecautionCorp program, where the stakeholders were collaborating mainly via boundary spanners, the boundary object application was important in facilitating the work of the boundary spanners and allowed them to maintain control over the collaboration parties. By applying a set of objects for collaboration control, the boundary spanners were able to maintain their spanning capabilities, bridging the boundaries between all different parties and ensuring that all stakeholders were kept informed. The relationship between the two parts of the Agile structures is constantly adjusted: when the configuration of the stakeholders is changed, amendments are needed for the boundary object application as well and vice versa.

Finally, one should acknowledge that the Agile perspective is not an immutable construct, but changes over time. Thus, there might be a link between the Agile process and the Agile perspective, describing that projects which have applied Agile methods have an influence on the Agile perspective of the organisation (**Arrow 9**). This is also the underlying message of Agile proponents: well-conducted Agile projects are better than plan-driven projects and will yield better results, which will foster future Agile projects (Beck 2000; Highsmith 2002; Schwaber 2004; and other guidelines which advocate for Agile). Examples of deliberate attempts to change or maintain the perspective can be observed in all three cases. First, Extreme Inc. wished to maintain or enhance its Avid approach and used significant resources in the hiring efforts to ensure compatibility. Second, Escapade and Carmine were adamant about working together, only due to a mutually shared Inclusive Agile perspective, and the development process and Agile structures were in place to enforce and maintain the perspective and to extend it to the more peripheral stakeholders as well. Third, PrecautionCorp employed Agile coaches in order to expand its Pragmatic perceptive across its own organisation, and partnered with organisations who wished to engage in similar activities. All these activities were put in place to ensure successful Agile project implementations. Each successful program, project or product launch can be used, according to the Agile coaches, to further the Agile perspective in the organisation and disprove employees who doubt the methods.

The relatively short time I spent with the three cases did not allow me to observe significant organisational changes, as that was not the intention of the study. However, the testimonials of the interviewees on how they have championed Agile methods in workplaces and how they perceived that introduction and utilisation of Agile methods can change organisations for the better lead me to believe that successful implementations can create stronger following of Agile methods, but unsuccessful projects can sow doubt in the minds of the stakeholders and create a more challenging environment for employees who hold an Agile worldview. The influence of the Agile process on the perspective towards Agile forms a feedback loop that connects the influence that the Agile perspective has on both the Agile process and the Agile structures back as a self-reinforcing cycle.

The proposed framework of Agile engagement uncovers the linkages between the different elements of Agile – the perspective, process and structures – and provides a holistic view of Agile. It explains how the perspective of the

organisation towards Agile development affects every other aspect of the development and how the elements of the development are interconnected. It is not a taxonomy (Conboy 2009) or a checklist (Qumer & Henderson-Sellers 2008) that can be applied to any project to identify 'agility'; rather, the framework proposes that agility cannot be assessed without taking into account the context of the project, the difference in the collaboration, the perspective on Agile, and the application of boundary objects.

7.6. Conclusion

This chapter has presented a synthesis of the three case studies. By synthesising the data from the three case studies, I have come to conclusion that the three case studies exemplify Agile organisations across three dimensions. First, the Extreme Inc. case is an example of an organisation where the members hold an Avid perspective towards Agile methods. The organisation had arranged the collaboration to follow a tightly integrated model where boundary objects are applied to support the pair programming method and foster face-to-face collaboration.

The case of Escapade and Carmine presents an example of an Inclusive Agile perspective, where organisations strive to focus on collaboration and boundary mitigation. The organisations had set up a collaboration configuration, where boundary spanners and all boundary objects were designed to create a sense of presence and ease of collaboration.

The last case, the PrecautionCorp program, is a study of an organisation where the members of the organisation have chosen to observe Agile methods in a Pragmatic way. All collaboration between the stakeholders was organised via selected boundary spanners who mitigated the boundaries but also maintained a level of control over the chaos by applying a variety of boundary objects.

My analysis across these three case studies is distilled into the Framework for Agile Engagement: a novel theoretical model that ties together different elements of Agile development and how these elements relate to the way an organisation conducts its Agile projects or builds its products. The model provides answers to my research questions on Agile perspectives, collaboration and boundary objects. The last chapter will conclude the thesis by outlining the theoretical and practical contributions, limitations and future research.

8. Contribution and Conclusion

> *It is interesting, when you look at the places that do all of these things, the rituals, but are not really Agile. Only Agile in name. Extreme Inc.'s been really keen – trying to get back to those first principles.*
>
> *– Designer, Extreme Inc.*

This thesis has presented an analysis of the Agile development methods based on three research questions. I have analysed how organisations differ when their perspective towards Agile differs as well. I have also analysed how organisations can configure their collaboration between their stakeholders, both internally and externally. In addition, I have investigated how the organisations apply boundary objects in order to support their collaboration. This study has led me to following conclusion: Agile software development consists of different, interconnected elements that influence how the organisation applies Agile methods and conducts its work. These elements and their connections are captured in my Framework of Agile engagement, which holistically incorporates perspectives, collaboration activities and material elements of Agile software development.

This work is the first to present such a holistic framework and to discuss Agile methods from philosophical and material perspectives. I see this model as a useful tool for both practitioners and academics who are studying Agile development and deciphering why Agile organisations or teams behave in a certain way. My presumption is that the organisations that are aware of the Agile practices and what their usage entails, and who are actively trying to diminish intra-organisational boundaries, are also following the values of the Agile Manifesto (Beck at al. 2001) more strictly than organisations that are constantly having issues with the methods and stakeholder collaboration. These organisations understand what embracing change (Beck 1999) entails.

I suggest that that by analysing how the Agile practices and objects are interconnected, organisations can better self-analyse their ways of utilising Agile methods. This may also improve understanding of the various Agile methods and how organisations can better accommodate a wider variety of different stakeholders who have different information needs. We have seen that some practices are very important and foster the organisational identity (Gal et al. 2008), whereas other practices are more susceptible to compromises or workarounds. Better understanding of practices, from a practice theory perspective, can give us valuable insights into organisational change and resistance.

8.1. Theoretical Contributions

This thesis contributes to both theoretical research on Agile software development and theory of boundary objects. The main contribution of this thesis is the Framework of Agile engagement: a holistic description of how different elements of Agile are linked together and how, by analysing such elements, one can better understand how the Agile process is supported and what influences the actions taken in the Agile development environment. The study also contributes in analysis of literature and how the authors who contribute to the field view Agile. The literature review

provides an insight into how Agile development is viewed by the authors who contribute to the field of Agile development literature.

Contributions are also made in the analysis of boundary objects. The theoretical contributions to the concept of boundary objects comes from applying the lens of boundary objects into the Agile development environment and by linking stakeholder collaboration, boundary spanning and boundary objects in a novel environment – an environment that poses unique challenges for the application of the objects. This thesis also presents a breakdown of boundary object categories, tailored to clarify the object application for collaboration in a software development environment.

8.1.1. Theoretical Contributions to Agile Literature

By addressing the research question set forth at the beginning of this paper, this study makes a number of theoretical contributions. First and foremost, this thesis proposes a holistic Agile software development framework, the Framework for Agile engagement, which addresses the main elements of Agile software development – the Agile perspective, the stakeholder configuration and the application of the boundary objects – and how these elements impact the Agile process, collaboration and product delivery. This novel theoretical view is the first major attempt to link together the Agile perspectives, derived from extensive literature review and analysis on Agile development, as well as from empirical data analysis. The framework challenges reductionist models that attempt to describe Agile projects without larger context of the differences between and within Agile organisations. It is also an account based on extensive empirical data, collected from three mature Agile organisations: a study that is still a rarity amidst the literature of the field, not a single case study or a comparison between Agile and waterfall organisations.

The framework for Agile engagement provides and important theoretical contribution for the following reasons: the holistic view presented in the framework is more aligned with the holistic nature of the Agile development methods. The proponents of the development methods were not interested in very specific and detailed guidelines but provided a philosophical framework, applicable for the whole development process (Beck et al. 2001). To date, theoretical discussions of Agile software development have not engaged thoroughly with the Manifesto of Agile development or other original source material, as discussed in the literature review section 2.4.1, section which details the literature that sees Agile development from a philosophical perspective. Although there are papers, which do acknowledge the philosophical nature of the method, the extant literature has not been synthesised or discussed in comparison with other perspectives. This is evident from the other sections of the literature review, which shows that empirical studies on Agile software development often choose to not embrace Agile development as a philosophy but as a commodity to be adapted and tailored to fit organisational needs (see literature analysis from section 2.4.2 to section 2.4.4). On the contrary, the framework for Agile engagement acknowledges that the held Agile perspective is a factor that feeds the outcomes of Agile development method application process. Reductionist views on the project omit the Agile perspective as a factor and rarely impose any scrutiny on the 'agility' of their subject (O'hEocha et al. 2010).

In addition, this work contributes to the Agile literature with the analysis conducted in the literature review and the findings distilled from this review. The study has examined and analysed the academic literature on Agile

development methods and covered widely cited and also less-known authors from the dawn of the field to the latest outings, a body of knowledge that has evolved during the over twenty years of Agile development. The review of the literature across the selected body of knowledge of Agile development presents a variety of views and perspectives on the values and principles of the Manifesto, the methods themselves and their application. Based on the literature review and the case studies, I have synthesised the both literature on Agile development and organisations following the Agile methods commonly fall along three major different categories along the spectrum of views: the Avid perspective, the Inclusive perspective and the Pragmatic perspective.

8.1.2. Theoretical Contributions to Boundary Objects

Thirdly, this thesis is one of the earliest studies that examine the role of boundary objects in the context of Agile software development projects as part of the Agile structure, interconnected with the stakeholder collaboration configuration. The thesis contributes to better understanding of how the application of boundary objects is linked to and influences the organisational roles of boundary spanners, which have been created to ensure the collaboration between the different parties of the Agile development environment. The analysis of the case studies has provided examples of how organisations leverage the boundary objects to heighten their collocated collaboration or how the object can be applied to mitigate challenges posed by a virtual project environment. In particular, the framework for Agile engagement sheds light on the interconnectedness among the stakeholder collaboration and the underlying Agile structure, consisting both of boundary object application and stakeholder collaboration configuration. It extends the understanding of boundary object application by providing a comparative case study and by focusing on all boundary objects applied in the cases. Previous studies analysing boundary objects have focused on a single object or a single use of multiple objects (e.g. Henderson 1991; Boujut & Blanco 2003; Lee 2007). The interplay between the different boundary objects has been given less attention.

It is especially important to understand the interconnectedness of boundary objects and stakeholders in Agile software development because of the volatile nature of the Agile environment. This is because Agile software development methods stress flexibility and readiness for change, as well as rapid pace of development, and this turbulence can pose a threat to object integrity and clarity of the collaboration efforts. Boundary object erosion (Subrahmanian et al. 2003) can occur if the changes introduced to the multiple affected objects in response to the same stimuli are not synchronised. Practitioners have to maintain and update all three types of boundary objects in order to avoid project status confusion, or disparities in the messages that are being conveyed across the boundaries. Careful maintenance of the boundary objects will ensure the efficiency of the boundary objects as collaborative tools (Carlile 1997).

The fourth and final contribution of this study is the conceptual model of the application of the boundary objects in the Agile environment. From the empirical data, the study identified three boundary object types that were salient to Agile software development: the infrastructural objects, projective objects and process objects. More specifically, infrastructural objects provided the background against which processes, people and other objects function. Projective objects contained a vision of the final product and were applied to communicate the vision and the end goals across the boundaries of the collaboration parties. Projective objects offered a dynamic representation of the system and served as a reference point for discussions around how to achieve the development goals. In addition,

process objects reflected the information contained in the projective objects. However, while projective objects provided a vision of the finished system, the information in the process objects translated this vision into more practical language (e.g. status information, discussion data) that provided a more in-depth perspective of the mechanics of how to get there. These three categories complement the extant analysis of boundary objects, expanding the application of the concept in Agile software development environment and contributing to the understanding of the material aspects of the phenomenon (Cecez-Kecmanovic et al. 2014).

8.2. Practical Contributions

The practical contributions of this thesis are several: practitioners can apply the framework for Agile engagement when analysing their own positions, can benefit from better understanding of the relations between Agile process, Agile perspective and Agile structures, and can enhance their understanding of the best possible application of boundary objects.

8.2.1. Practical Contributions to Stakeholders and Agile Values

One of the goals for my thesis was to investigate the linkage between the objects, stakeholders and the Agile values and principles – the perspective towards Agile that the organisations hold. This linkage was presented in the framework for Agile engagement. My presumption is that the organisations that are aware of the Agile practices and what their usage entails and who are actively trying to diminish intra-organisational boundaries are also following the values of the Agile Manifesto (Beck at al. 2001) more strictly than organisations that are constantly having issues with the methods and stakeholder collaboration. These organisations understand what embracing change (Beck 1999) entails. I believe that analysing how the Agile methods are applied in principle and in practice may help organisations to self-analyse their extant Agile perspective and process.

By understanding what Agile perspective the organisation holds currently, the organisation may also improve understanding of how it can better accommodate a wider variety of different stakeholders who have different needs and views. The organisation can have more emphasis on the activities that foster organisational identity (Gal et al. 2008) and enhance organisation-wide understanding of the Agile perspective.

The analysis of the Agile perspectives can be a useful tool for organisations when they discuss introduction of Agile or expansion of their Agile practices beyond the development teams. No perspective is wrong or right: the factors that determine the perspective are linked to the context and the circumstances of the organisations, but also other factors might have a role that defines the perspective. Understanding that there are several ‘right’ ways to conduct Agile development can aid Agile coaches who are trying to convince sceptical organisations to test out the methods. These different perspectives are also useful tools for internal communication among the teams, as they can provide useful language and terminology, which is needed when the agility of the work is discussed.

8.2.2. Practical Contributions for Application of Boundary objects

The study also enhances understanding of how the boundary object types facilitate the Agile development process and how the stakeholders and boundary objects interact. Better understanding of the underlying Agile structures can

help organisations to design and implement structures that better align with their desired customer or stakeholder interactions. Customer collaboration, one of the main values of the Agile Manifesto, has been researched to some extent (Koskela & Abrahamsson 2004; Kautz 2009; Hoda et al. 2011), but not from the perspective of boundary objects combined with stakeholder engagement and perspective towards Agile. This study brings new insights into how the Agile practices that are linked to customer collaboration can be enhanced with boundary objects. The boundary object perspective provides a new way to investigate the mediation of information, specifically in the Agile project environment.

Practitioners can use these findings to better understand the functions of the different boundary objects types, as well as how each of them may be leveraged to enhance collaboration and foster Agile values. In particular, our study suggests that infrastructural objects should be crafted carefully, in order to ensure that they are not too rigid and prevent other object types from being effectively used for collaboration. If infrastructural boundary objects such as contracts are too rigid, they can hinder change and negatively impact other objects. Infrastructural objects can disappear from sight until something goes awry and the objects are invoked to resolve a conflict (Hoda et al. 2011). In Agile development, our study suggests that the collaborating parties should craft objects cooperatively and remain mindful of the role of infrastructural objects. Collaborative amendments of infrastructural objects according to the project progress could help to avoid unnecessary issues that might arise in project disagreements (Hoda et al. 2011).

The taxonomy of boundary objects developed in our study can also be used as a diagnostic tool to assess what types of objects exist and which of these are missing in an Agile software development environment. In particular, Agile practitioners would be able to analyse their current utilisation of objects and identify which object types could be introduced or leveraged more effectively. For example, projective objects can be powerful tools in ensuring common understanding, as seen in previous studies on prototypes (e.g. Subrahmanian et al. 2003; Gal et al. 2008; Winkler et al. 2014). As one of the novel aspects of Agile development is the sharing of the early versions of the final product as a form of a prototype, our study suggests that Agile practitioners should collaborate using early system versions. This is because they can elicit feedback from users who might be less familiar with how to use and comment on the unfinished, sometimes crude, prototypes (Da Silva et al. 2011; Kautz 2010). The feedback should then be recorded to either the projective objects themselves or to the process objects, in order to maximise the value of having a working piece of software that customers can evaluate.

Practitioners would also benefit from an understanding of the interconnectedness of the three object types and the roles of the boundary spanners. In our study, we have discussed the different ways in which the objects were applied by the boundary spanners in order to convey meaning and clarify goals. For instance, our study suggests that it is especially important to acknowledge the relationship between process boundary objects and projective boundary objects. The boundary spanners should maintain the integrity of both categories and ensure the information is not unsynchronised. Boundary objects that are not created for a flexible environment can quickly become obstacles for collaboration, as they are unable to adequately capture changes within the projective boundary objects. Understanding the application of the objects can help practitioners identify potential collaborative pitfalls and consequently, circumvent them to improve the effectiveness of collaboration.

8.3. Limitations of the Research

There are some limitations of this research, as there are in all case studies, qualitative or quantitative. The three cases present three different aspects of Agile development, but this does not mean that all Agile projects necessarily fall in line with these three categories. However, I believe that they provide a representative selection of organisations that people would generally agree to be exemplars of Agile software development and not just of practicing iterative development (O'hEocha et al. 2010). This notion is further enhanced by the background research I conducted by familiarising myself with the case organisations and by immersing myself into the local Agile community. In addition, the interviews and discussions with practitioners from other Agile organisations provided me with insights into and background information on the landscape of Agile development in Australia.

Nevertheless, this research is limited to the context of organisations primarily located in Australia. Agile development projects, collocated or distributed, are conducted all over the world, but Australia is a unique location when it comes to globally or even locally distributed development. The time differences between Australia, India, China and Canada are significant and require accommodation by all parties. In addition, remote working is commonplace for Australians and employees are familiar with best practices that enable remote work (Miles 2013).

Another potentially limiting aspect of the research is the industries that my case organisations operated in. Two of the organisations were from the finance industry, an industry where Agile development is very prominent, behind only software development industry (VersionOne 2017). Conversely, the two other organisations, Carmine and Escapade, represent eCommerce and Agile consulting and provide insights into other industries besides those of Extreme Inc. and PrecautionCorp – representing two very different organisations in the same industry. Even though the cases do not cover a wide variety of different industry sectors, I believe that when it comes to Agile development methods and product development, the differences in the Agile method application and the complexity of the stakeholder network are more impactful when it comes to the Agile development environment, rather than the influences of the industry culture.

When it comes to generalisability of case studies, Walsham cites a paper by Lee and Baskerville (2003). Lee and Baskerville describe a framework of four types of generalisability that, according to them, applies to both positivist and interpretive research, that is, concerned with empirical statements, for example, data, measurements, observations or descriptions, or theoretical statements. The four types are generalisations: from data to description; from description to theory; from theory to description; and from concept to theory. My study takes descriptions and observations and a set of empirical statements and generalises to theory. Lee and Baskerville provide an example of such a study when they state that (pp. 236, emphasis mine):

Empirical descriptions serving as inputs to the process of generalizing could specify, for example, the measurements of the effect of a treatment administered in a particular field experiment; ***the rich details in a case study of a particular corporate headquarters;*** *or the sample estimates of the population characteristics of workers in a particular geographic region. The resulting theoretical statements could comprise, respectively, a theory positing new variables and the relationships among them that would explain the experimental effect that was measured in the field experiment;* ***a theory explaining the corporate headquarters' social structure and culture that would account for the behaviors and actions noted in the thick description of the case study****; or a theory explaining the*

underlying labor market forces that would result in the levels of the population characteristics that the sample estimated.

In this thesis I have conducted a very similar exercise to that described in this example. I have provided rich details of Agile method application in the each of the three cases and proposed a theory based on the combined and recurring elements of Agile software development in these organisations. This study is generalising the theory from on the analysis of the routines that form the daily life in these organisations. By analysing the accounts the informants have given me on on the daily activities I have been able to form a picture of the organisational structures, understand the communication and collaboration between different stakeholders in the companies and compare these findings across the cases. However, analysis of the practices themselves is not the sole source of theorisation in this work. I draw on both the practices as well as the perspectives the interviewees have, expressed towards these practices. Where the practices are somewhat similar, the context and the perspectives differ. This has provided new variables, described in the framework for Agile engagement.

8.4. Future Directions

There are several avenues this research can be taken in the future. First, this thesis has proposed a framework of Agile engagement, a novel framework for analysing Agile environments. It has also proposed a framework of the application of the boundary objects in the context of Agile software development. Finally, the study has also expanded on Agile literature by providing an analysis of the different perspectives literature on Agile software development has towards the methods. Each of these theoretical contributions can be further analysed and expanded with new case studies or new literature analysis.

The most significant contribution of this thesis, the framework for Agile engagement, can be further investigated by applying the framework in new Agile software development case study context. I wish to explore how the framework of Agile engagement is accepted in the Agile communities and if Agile practitioners outside my case organisations find the framework helpful. I believe that the framework should be applicable in any Agile organisations but it would be beneficial to discuss and debate the framework with more practitioners who might have interesting perspectives on the subject matter. In order to discuss the framework with the practitioners, the framework needs to be first communicated to practitioners. This requires continuing active engagement with the Agile community which might initiate new research projects.

Similarly, the categorisation of boundary objects and their linkage to the stakeholder configuration could be applicable for organisations who struggle with collaboration and wish to understand how to best include their various stakeholder groups. The various applications of the boundary objects concept, discussed in 2.5 shows the versatility of the theory. Discussions on software development often allocate insufficient attention to the objects, a subject matter that could be one of the future directions of this study. The framework for boundary objects use in Agile software development environment could be potentially explored in other business contexts. If these boundary objects are beneficial for collaboration in Agile development, they might be applicable in for a variety of business stakeholders from marketing to communications, especially since Agile methods also being applied outside their traditional domain of software development (Rigby et al. 2010). Similarly, the framework of Agile engagement

could be applied in different Agile context, not just limit the application within software development. Agile methods, especially if seen as a holistic philosophy rather than a set of simple guidelines, is not necessarily bound to software development but with some Inclusive approach mentality, could be extended to a vide variety of activities. Future research of the Agile software development methods could be similarly extended to encompass future endeavours where Agile methods are going to be applied.

Finally, the main contribution of this work, presented in the forms of the frameworks, is the entanglement of Agile values and principles, the core ideas described in the Manifesto for Agile development, and the organisations who wish to reap the benefits of these ideas. The Manifesto has proven to be a salient document, which has inspired people to change their work environments and to strive for improvement against old-fashioned ways of working. Practices, terminology and organisational context might change, but the basic philosophy remains. Future research into Agile software development could strive to explain why the Manifesto has garnered such attention and following within the software development community and explore the philosophical aspects imbued into the daily lives of Agile practitioners.

References

Abrahamsson, P., Salo, O., Ronkainen, J., and Warsta, J. 2002. *Agile Software Development Methods: Review and Analysis*. VTT Finland.

Abrahamsson, P., Warsta, J., Siponen, M. T., and Ronkainen, J. 2003. "New Directions on Agile Methods: A Comparative Analysis", *Software Engineering, 2003. Proceedings. 25th International Conference on*: IEEE, pp. 244-254.

Anderson, D. J. 2010. *Kanban: Successful Evolutionary Change for Your Technology Business*. Blue Hole Press.

Azizyan, G., Magarian, M. K., and Kajko-Matsson, M. 2011. "Survey of Agile Tool Usage and Needs", *Agile Conference (AGILE), 2011*: IEEE, pp. 29-38.

Balijepally, V., Mahapatra, R., Nerur, S., and Price, K. H. 2009. "Are Two Heads Better Than One for Software Development? The Productivity Paradox of Pair Programming", *MIS Quarterly*, pp. 91-118.

Barrett, M., and Oborn, E. 2010. "Boundary Object Use in Cross-Cultural Software Development Teams", *Human Relations* (63:8), pp. 1199-1221.

Barrett, M., Oborn, E., Orlikowski, W. J., and Yates, J. 2012. "Reconfiguring Boundary Relations: Robotic Innovations in Pharmacy Work", *Organization Science* (23:5), pp. 1448-1466.

Baskerville, R., Pries-Heje, J., and Madsen, S. 2011. "Post-Agility: What Follows a Decade of Agility?", *Information and Software Technology* (53:5), pp. 543-555.

Bass, J. M. 2016. "Artefacts and Agile Method Tailoring in Large-Scale Offshore Software Development Programmes", *Information and Software Technology* (75), pp. 1-16.

Bechky, B. A. 2003. "Sharing Meaning across Occupational Communities: The Transformation of Understanding on a Production Floor", *Organization Science* (14:3), pp. 312-330.

Beck, K. 1999. "Embracing Change with Extreme Programming", *Computer* (32:10), pp. 70-77.

Beck, K. 2000. *Extreme Programming Explained: Embrace Change*. Addison-Wesley Professional.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., and Jeffries, R. 2001. "Agile Manifesto, 2001" from http://www.agilemanifesto.org.

Biernacki, P., and Waldorf, D. 1981. "Snowball Sampling: Problems and Techniques of Chain Referral Sampling", *Sociological Methods & Research* (10:2), pp. 141-163.

Boehm, B. 2002. "Get Ready for Agile Methods, with Care", Computer (35:1), pp. 64-69.

Boehm, B. W. 1988. "A spiral model of software development and enhancement." *Computer*, (21:5), pp. 61-72.

Boehm, B., & Turner, R. 2005. "Management challenges to implementing agile processes in traditional development organizations." *IEEE software*, (*22*:5), pp. 30-39.

Boell, S. K., & Cecez-Kecmanovic, D. (2014). A hermeneutic approach for conducting literature reviews and literature searches. CAIS, 34, 12.

Boell, S. K., and Hoof, F. 2016. "Using Heider's Epistemology of Thing and Medium for Unpacking the Conception of Documents: Gantt Charts and Boundary Objects", Proceedings from the Document Academy (2:1), p. 3.

Boujut, J.-F., and Blanco, E. 2003. "Intermediary Objects as a Means to Foster Co-Operation in Engineering Design", Computer-Supported Cooperative Work (CSCW) (12:2), pp. 205-219.

Callon, M., & Latour, B. (1981). "Unscrewing the big Leviathan: how actors macro-structure reality and how sociologists help them to do so. Advances in social theory and methodology: Toward an integration of micro-and macro-sociologies, 1.

Cao, L., Mohan, K., Ramesh, B., and Sarkar, S. 2013. "Adapting Funding Processes for Agile IT Projects: An Empirical Investigation", *European Journal of Information Systems* (22:2), pp. 191-205.

Cao, L., Mohan, K., Xu, P., and Ramesh, B. 2009. "A Framework for Adapting Agile Development Methodologies", *European Journal of Information Systems* (18:4), pp. 332-343.

Cao, L., Ramesh, B., and Abdel-Hamid, T. 2010. "Modeling Dynamics in Agile Software Development", *ACM Transactions on Management Information Systems (TMIS)* (1:1), p. 5.

Carlile, P. R. 1997. *Understanding Knowledge Transformation in Product Development: Making Knowledge Manifest through Boundary Objects*.

Carlile, P. R. 2002. "A Pragmatic View of Knowledge and Boundaries: Boundary Objects in New Product Development," *Organization Science* (13:4), pp. 442-455.

Cecez-Kecmanovic, D., Galliers, R. D., Henfridsson, O., Newell, S., and Vidgen, R. 2014. "The Sociomateriality of Information Systems: Current Status, Future Directions", *MIS Quarterly* (38:3), pp. 809-830.

Chan, F. K., and Thong, J. Y. 2009. "Acceptance of Agile Methodologies: A Critical Review and Conceptual Framework", *Decision Support Systems* (46:4), pp. 803-814.

Chong, J. 2005. "Social Behaviors on XP and Non-XP Teams: A Comparative Study", *Agile Conference, 2005. Proceedings*: IEEE, pp. 39-48.

Chow, T., and Cao, D.-B. 2008. "A Survey Study of Critical Success Factors in Agile Software Projects", *Journal of Systems and Software* (81:6), pp. 961-971.

Ciriello, R. F., Aschoff, R., Dolata, M., and Richter, A. 2014. *Communicating Ideas Purposefully – toward a Design Theory of Innovation Artifacts*.

Cockburn, A. 2004. Crystal Clear: A Human-Powered Methodology for Small Teams. Pearson Education.

Cockburn, A. (2006). Agile software development: the cooperative game: Pearson Education.

Cockburn, A. (2018). Crystal methodologies. Retrieved from http://alistair.cockburn.us/Crystal+methodologies

Cockburn, A., and Highsmith, J. 2001. "Agile Software Development, the People Factor", *Computer* (34:11), pp. 131-133.

Cohn, M. 2004. *User Stories Applied: For Agile Software Development*. Addison-Wesley Professional.

Cohn, M. 2010. *Succeeding with Agile: Software Development Using Scrum*. Pearson Education.

Cohn, M., and Ford, D. 2003. "Introducing an Agile Process to an Organization [Software Development]", *Computer* (36:6), pp. 74-78.

Conboy, K. 2009. "Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development", *Information Systems Research* (20:3), pp. 329-354.

Conboy, K., Coyle, S., Wang, X., and Pikkarainen, M. 2010. "People over Process: Key People Challenges in Agile Development", *IEEE Software* (99:1), pp. 47-57.

Coyne, R. D. 1996. *Designing Information Technology in the Postmodern Age: From Method to Metaphor*. MIT Press.

Cummings, T.G., 1978. "Self-regulating work groups: A socio-technical synthesis". *Academy of management Review,* (3:3), pp.625-634.

Da Silva, T. S., Martin, A., Maurer, F., and Silveira, M. 2011. "User-Centered Design and Agile Methods: A Systematic Review", *Agile Conference (AGILE), 2011*: IEEE, pp. 77-86.

de Lima Salge, C. A., and Berente, N. 2016. "Pair Programming Vs. Solo Programming: What Do We Know after 15 Years of Research?", *System Sciences (HICSS), 2016 49th Hawaii International Conference*: IEEE, pp. 5398-5406.

DeMarco, T., & Lister, T. 2013. Peopleware: productive projects and teams: Addison-Wesley

Derby, E., Larsen, D., & Schwaber, K. 2006. Agile retrospectives: Making good teams great.

Di Marco, M. K., Alin, P., and Taylor, J. E. 2012. "Exploring Negotiation through Boundary Objects in Global Design Project Networks", *Project Management Journal* (43:3), pp. 24-39.

Di Marco, M. K., Taylor, J. E., and Alin, P. 2010. "Emergence and Role of Cultural Boundary Spanners in Global Engineering Project Networks", *Journal of Management in Engineering* (26:3), pp. 123-132.

Dingsøyr, T., and Lassenius, C. 2016. "Emerging Themes in Agile Software Development: Introduction to the Special Section on Continuous Value Delivery", Information and Software Technology (77), pp. 56-60.

Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. 2012. A decade of agile methodologies: Towards explaining agile software development: Elsevier

Donaldson, S. E., & Siegel, S. G. (2001). Successful software development: Prentice Hall Professional.

Dowling, P., & McGrath, K. (2015). Using free and open source tools to manage software quality. Queue, 13(4), 20.

Drechsler, A., and Ahlemann, F. 2015. "Toward a General Theory of Agile Project Management – a Research Design", European Conference of Information Systems.

Drury, M., Conboy, K., & Power, K. (2012). Obstacles to decision making in Agile software development teams. Journal of Systems and Software, 85(6), 1239-1254.

Dybå, T., Arisholm, E., Sjøberg, D. I., Hannay, J. E., and Shull, F. 2007. "Are Two Heads Better Than One? On the Effectiveness of Pair Programming", IEEE Software (24:6), pp. 12-15.

Dybå, T., and Dingsøyr, T. 2008. "Empirical Studies of Agile Software Development: A Systematic Review", *Information and Software Technology* (50:9), pp. 833-859.

Dybå, T., and Dingsøyr, T. 2009. "What Do We Know About Agile Software Development?", *IEEE Software* (26:5), pp. 6-9.

Eisenhardt, K. M. 1989. "Building Theories from Case Study Research", *Academy of Management Review* (14:4), pp. 532-550.

Eveleens, J. L., and Verhoef, C. 2010. "The Rise and Fall of the Chaos Report Figures", *IEEE Software* (27:1), pp. 30-36.

Fitzgerald, B., Hartnett, G., and Conboy, K. 2006. "Customising Agile Methods to Software Practices at Intel Shannon", *European Journal of Information Systems* (15:2), pp. 200-213.

Freedman, R. 2016. "The Agile Consulting Model", in *The Agile Consultant*. Springer, pp. 167-176.

Gal, U., Lyytinen, K., and Yoo, Y. 2008. "The Dynamics of IT Boundary Objects, Information Infrastructures, and Organisational Identities: The Introduction of 3D Modelling Technologies into the Architecture, Engineering, and Construction Industry", *European Journal of Information Systems* (17:3), pp. 290-304.

Geertz, C., 1973. The interpretation of cultures (Vol. 5019). Basic books.

Gerwin, D. and Moffat, L., 1997. Authorizing processes changing team autonomy during new product development. Journal of Engineering and Technology Management, 14(3-4), pp.291-313.

Ghobadi, S., and Mathiassen, L. 2016. "Perceived Barriers to Effective Knowledge Sharing in Agile Software Teams", *Information Systems Journal* (26:2), pp. 95-125.

Goh, J. C.-L., Pan, S. L., and Zuo, M. 2013. "Developing the Agile IS Development Practices in Large-Scale IT Projects: The Trust-Mediated Organizational Controls and IT Project Team Capabilities Perspectives", *Journal of the Association for Information Systems* (14:12), p. 722.

Guzzo, R. A., and Dickson, M. W. 1996. "Teams in Organizations: Recent Research on Performance and Effectiveness", *Annual Review of Psychology* (47:1), pp. 307-338.

Hafermalz, E., and Riemer, K. 2015. "The Question of Materiality: Mattering in the Network Society", *European Conference of Information Systems*.

Henderson, K. 1991. "Flexible Sketches and Inflexible Data Bases: Visual Communication, Conscription Devices, and Boundary Objects in Design Engineering", *Science, Technology, & Human Values* (16:4), pp. 448-473.

Highsmith, J., and Cockburn, A. 2001. "Agile Software Development: The Business of Innovation", *Computer* (34:9), pp. 120-127.

Highsmith, J. 2002. *Agile Software Development Ecosystems*. Addison-Wesley Professional.

Hirschheim, R., and Klein, H. K. 1989. "Four Paradigms of Information Systems Development", *Communications of the ACM* (32:10), pp. 1199-1216.

Hoda, R., Noble, J., and Marshall, S. 2011. "The Impact of Inadequate Customer Collaboration on Self-Organizing Agile Teams", *Information and Software Technology* (53:5), pp. 521-534.

Holmström, H., Fitzgerald, B., Ågerfalk, P. J., and Conchúir, E. Ó. 2006. "Agile Practices Reduce Distance in Global Software Development", *Information Systems Management* (23:3), pp. 7-18.

Humble, J., and Farley, D. 2010. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (Adobe Reader)*. Pearson Education.

Hummel, M. 2014. "State-of-the-Art: A Systematic Literature Review on Agile Information Systems Development", *System Sciences (HICSS), 2014 47th Hawaii International Conference on*: IEEE, pp. 4712-4721.

Hunt, A. 2015 "The Failure of Agile" URL: https://toolshed.com/2015/05/the-failure-of-agile.html, accessed 2.3.2018

Hüttermann, M. 2012. *DevOps for Developers*. Apress.

PMI, Project Management Institute,. 1987. *Project Management Body of Knowledge (PMBOK)*, Project Management Institute.

Iorio, J., and Taylor, J. E. 2014. "Boundary Object Efficacy: The Mediating Role of Boundary Objects on Task Conflict in Global Virtual Project Networks", *International Journal of Project Management* (32:1), pp. 7-17.

Jonsson, K., Holmström, J., and Lyytinen, K. 2009. "Turn to the Material: Remote Diagnostics Systems and New Forms of Boundary-Spanning", *Information and Organization* (19:4), pp. 233-252.

Kabbedijk, J., and Jansen, S. 2011. "Steering Insight: An Exploration of the Ruby Software Ecosystem", *International Conference of Software Business*: Springer, pp. 44-55.

Kautz, K. (2009). Customer and user involvement in agile software development Agile Processes in Software Engineering and Extreme Programming (pp. 168-173): Springer.

Kautz, K., 2010, "Participatory design activities and agile software development." *In IFIP Working Conference on Human Benefit through the Diffusion of Information Systems Design Science Research* (pp. 303-316). Springer, Berlin, Heidelberg.

Kerzner, H. 2013. *Project Management: A Systems Approach to Planning, Scheduling, and Controlling*. John Wiley & Sons.

Kirkman, B.L. and Rosen, B., 1999. "Beyond self-management: Antecedents and consequences of team empowerment." *Academy of Management journal*, (42:1), pp.58-74.

Klein, H. K., and Myers, M. D. 1999. "A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems", *MIS Quarterly*, pp. 67-93.

Kniberg, H., and Ivarsson, A. 2012. "Scaling Agile @ Spotify", *URL: wordpress. com/2012/11/113617905-scaling-Agile-spotify-11. pdf*).

Koch, S. (2004). "Agile principles and open source software development: A theoretical and empirical discussion." *Paper presented at the International Conference on Extreme Programming and Agile Processes in Software Engineering.*

Korkala, M., and Abrahamsson, P. 2007. "Communication in Distributed Agile Development: A Case Study", *Software Engineering and Advanced Applications, 2007. 33rd EUROMICRO Conference on*: IEEE, pp. 203-210.

Koskela, J., and Abrahamsson, P. 2004. "On-Site Customer in an XP Project: Empirical Results from a Case Study", *European Conference on Software Process Improvement*: Springer, pp. 1-11.

Koskinen, K. U., and Mäkinen, S. 2009. "Role of Boundary Objects in Negotiations of Project Contracts", *International Journal of Project Management* (27:1), pp. 31-38.

Lakoff, G., and Johnson, M. 1999. *Philosophy in the Flesh*. New York: Basic books.

Langley, A. 1999. "Strategies for Theorizing from Process Data", *Academy of Management Review* (24:4), pp. 691-710.

Larman, C., and Basili, V. R. 2003. "Iterative and Incremental Developments: A Brief History", *Computer* (36:6), pp. 47-56.

Latour, B., 2005. "Reassembling the social: An introduction to actor-network-theory." Oxford university press.

Latour, B., and Woolgar, S. 2013. *Laboratory Life: The Construction of Scientific Facts*. Princeton University Press.

Lee, A. S., and Baskerville, R. L. 2003. "Generalizing Generalizability in Information Systems Research", *Information Systems Research* (14:3), pp. 221-243.

Lee, C. P. 2007. "Boundary Negotiating Artifacts: Unbinding the Routine of Boundary Objects and Embracing Chaos in Collaborative Work", *Computer-Supported Cooperative Work (CSCW)* (16:3), pp. 307-339.

Lee, G., and Xia, W. 2010. "Toward Agile: An Integrated Analysis of Quantitative and Qualitative Field Data on Software Development Agility", *MIS Quarterly* (34:1), pp. 87-114.

Lehtinen, T. O., Mäntylä, M. V., Vanhanen, J., Itkonen, J., and Lassenius, C. 2014. "Perceived Causes of Software Project Failures – an Analysis of Their Relationships", *Information and Software Technology* (56:6), pp. 623-643.

Levina, N., and Vaast, E. 2005. "The Emergence of Boundary Spanning Competence in Practice: Implications for Implementation and Use of Information Systems", *MIS Quarterly*, pp. 335-363.

Levina, N., and Vaast, E. 2008. "Innovating or Doing as Told? Status Differences and Overlapping Boundaries in Offshore Collaboration", *MIS Quarterly*, pp. 307-332.

MacCormack, A., Verganti, R., and Iansiti, M. 2001. "Developing Products on 'Internet Time': The Anatomy of a Flexible Development Process", *Management Science* (47:1), pp. 133-150.

Manz, C. C., and Sims Jr., H. P. 1987. "Leading Workers to Lead Themselves: The External Leadership of Self-Managing Work Teams", *Administrative Science Quarterly*, pp. 106-129.

Marchenko, A., and Abrahamsson, P. 2008. "Scrum in a Multiproject Environment: An Ethnographically-Inspired Case Study on the Adoption Challenges", *Agile, 2008.. Conference*: IEEE, pp. 15-26.

Marheineke, M., Velamuri, V. K., and Möslein, K. M. 2016. "On the Importance of Boundary Objects for Virtual Collaboration: A Review of the Literature", *Technology Analysis & Strategic Management* (28:9), pp. 1108-1122.

Martin, R, 2016 "The Scribe's Oath presentation", *YOW! Conference 2016,* Sydney, Australia, 8th - 9th of December 2016

Martin, A., Biddle, R., and Noble, J. 2010. "An Ideal Customer: A Grounded Theory of Requirements Elicitation, Communication and Acceptance on Agile Projects", *Agile Software Development*. Springer, pp. 111-141.

Martini, A., Pareto, L., and Bosch, J. 2016. "A Multiple Case Study on the Inter-Group Interaction Speed in Large, Embedded Software Companies Employing Agile", *Journal of Software: Evolution and Process* (28:1), pp. 4-26.

Maruping, L.M., Zhang, X. and Venkatesh, V., 2009. "Role of collective ownership and coding standards in coordinating expertise in software project teams." *European Journal of Information Systems*, (18:4), pp.355-371.

McHugh, O., Conboy, K., and Lang, M. 2012. "Agile Practices: The Impact on Trust in Software Project Teams", *IEEE Software* (29:3), pp. 71-76.

Miles, E. 2013. "Teleworking in Australia", retrieved 24.10.2017, from http://www.mccrindle.com.au/ResearchSummaries/2013/Teleworking_in_Australia.pdf.

Modi, S., Abbott, P., and Counsell, S. 2013. "Negotiating Common Ground in Distributed Agile Development: A Case Study Perspective", *Global Software Engineering (ICGSE), 2013 IEEE 8th International Conference:* IEEE, pp. 80-89.

Myers, M. D., and Newman, M. 2007. "The Qualitative Interview in IS Research: Examining the Craft", *Information and Organization* (17:1), pp. 2-26.

Nerur, S., and Balijepally, V. 2007. "Theoretical Reflections on Agile Development Methodologies", *Communications of the ACM* (50:3), pp. 79-83.

Nerur, S., Cannon, A., Balijepally, V., and Bond, P. 2010. "Towards an Understanding of the Conceptual Underpinnings of Agile Development Methodologies", *Agile Software Development*: Springer, pp. 15-29.

Nerur, S., Mahapatra, R., and Mangalaraj, G. 2005. "Challenges of Migrating to Agile Methodologies", *Communications of the ACM* (48:5), pp. 72-78.

Nicolini, D., Mengis, J., and Swan, J. 2012. "Understanding the Role of Objects in Cross-Disciplinary Collaboration", *Organization Science* (23:3), pp. 612-629.

O'hEocha, C., Conboy, K., and Wang, X. 2010. "So You Think You're Agile?", *International Conference on Agile Software Development*: Springer, pp. 315-324.

Orlikowski, W. J. 2007. "Sociomaterial Practices: Exploring Technology at Work", *Organization Studies* (28:9), pp. 1435-1448.

Paasivaara, M., Durasiewicz, S., and Lassenius, C. 2008. "Distributed Agile Development: Using Scrum in a Large Project", *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conferencen*: IEEE, pp. 87-95.

Paasivaara, M., Lassenius, C., and Heikkilä, V. T. 2012. "Inter-Team Coordination in Large-Scale Globally Distributed Scrum: Do Scrum-of-Scrums Really Work?", Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement: ACM, pp. 235-238.

Paetsch, F., Eberlein, A. and Maurer, F., 2003. "Requirements engineering and agile software development." *In Enabling Technologies: Infrastructure for Collaborative Enterprises. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops*, pp. 308-313

Pan, S. L., and Tan, B. 2011. "Demystifying Case Research: A Structured–Pragmatic–Situational (SPS) Approach to Conducting Case Studies", *Information and Organization* (21:3), pp. 161-176.

Parmiggiani, E., and Mikalsen, M. 2013. "The Facets of Sociomateriality: A Systematic Mapping of Emerging Concepts and Definitions", *Scandinavian Conference on Information Systems*: Springer, pp. 87-103.

Pickering, A. 2010. *The Mangle of Practice: Time, Agency, and Science*. University of Chicago Press.

Poppendieck, M., and Poppendieck, T. 2003. *Lean Software Development: An Agile Toolkit*. Addison-Wesley.

Qumer, A., and Henderson-Sellers, B. 2008. "A Framework to Support the Evaluation, Adoption and Improvement of Agile Methods in Practice", *Journal of Systems and Software* (81:11), pp. 1899-1919.

Ragin, C. C. 1992. "Casing and the Process of Social Inquiry", *What is a Case?*, pp. 217-226.

Ramesh, B., Cao, L., and Baskerville, R. 2010. "Agile Requirements Engineering Practices and Challenges: An Empirical Study", *Information Systems Journal* (20:5), pp. 449-480.

Raymond, E. S. 2001. *The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. O'Reilly Media, Inc.

Ries, E. 2011. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Crown Business.

Rigby, D. K., Sutherland, J., and Takeuchi, H. 2016. "Embracing Agile", *Harvard Business Review* (94:5), pp. 40-50.

Rowe, P.G., 1991. Design thinking. MIT press

Sarker, S., and Sarker, S. 2009. "Exploring Agility in Distributed Information Systems Development Teams: An Interpretive Study in an Offshoring Context", Information Systems Research (20:3), pp. 440-461.

Schultze, U., 2000. A confessional account of an ethnography about knowledge work. *MIS Quarterly*, pp.3-41.

Schlauderer, S. and Overhage, S., 2013. Exploring the customer perspective of agile development: Acceptance factors and on-site customer perceptions in scrum projects

Schwaber, K. 2004. *Agile Project Management with Scrum*. Microsoft Press.

Scrum Alliance, T. R. 2015. "The 2015 State of Scrum Report", from www.scrumalliance.org/why-scrum/state-of-scrum-report/2015-state-of-scrum.

Seymour, M., and Coyle, S. 2016. *Towards a Research Agenda for Adopting Agile Project Management in Creative Industries*.

Sharp, H., and Robinson, H. 2004. "An Ethnographic Study of XP Practice", *Empirical Software Engineering* (9:4), pp. 353-375.

Sharp, H., Robinson, H., and Petre, M. 2009. "The Role of Physical Artefacts in Agile Software Development: Two Complementary Perspectives", *Interacting with Computers* (21:1), pp. 108-116.

Smith, D. M. 2011. "Hype Cycle for Cloud Computing, 2011", *Gartner Inc., Stamford* (71).

Star, S. L. 2010. "This Is Not a Boundary Object: Reflections on the Origin of a Concept," *Science, Technology, & Human Values* (35:5), pp. 601-617.

Star, S. L., and Griesemer, J. R. 1989. "Institutional Ecology, Translations and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39", *Social Studies of Science* (19:3), pp. 387-420.

Star, S. L., and Ruhleder, K. 1996. "Steps toward an Ecology of Infrastructure: Design and Access for Large Information Spaces", *Information Systems Research* (7:1), pp. 111-134.

Strauss, A., and Corbin, J. 1998. *Basics of Qualitative Research: Procedures and Techniques for Developing Grounded Theory*, Thousand Oaks, CA: Sage.

Stray, V., Fægri, T.E. and Moe, N.B., 2016. "Exploring norms in agile software teams." *In International Conference on Product-Focused Software Process Improvement,* pp. 458-467. Springer, Cham

Strode, D. E., Huff, S. L., Hope, B., and Link, S. 2012. "Coordination in Co-Located Agile Software Development Projects", *Journal of Systems and Software* (85:6), pp. 1222-1238.

Subrahmanian, E., Monarch, I., Konda, S., Granger, H., Milliken, R., and Westerberg, A. 2003. "Boundary Objects and Prototypes at the Interfaces of Engineering Design"*, Computer-Supported Cooperative Work (CSCW)* (12:2), pp. 185-203.

Sutherland, J., Schoonheim, G., Rustenburg, E., and Rijk, M. 2008. "Fully Distributed Scrum: The Secret Sauce for Hyperproductive Offshored Development Teams", *Agile, 2008. Conference*: IEEE, pp. 339-344.

Takeuchi, H., and Nonaka, I. 1986. "The New New Product Development Game", *Harvard Business Review*.

Thomas. D. 2014 "Agile is Dead (Long Live Agility) URL: https://pragdave.me/blog/2014/03/04/time-to-kill-agile.html accessed 2.3.2018

Van Maanen, J. 1979. "The Fact of Fiction in Organizational Ethnography", *Administrative Science Quarterly*, pp. 539-550.

Van Maanen, J., & Barley, S. R. 1982. "Occupational communities: Culture and control in organizations"

VersionOne, T.R. 2016. "10th Annual State of Agile Survey", *Technical report*, Version One

VersionOne, T. R. 2017. "11th Annual State of Agile Survey", *Technical report*, Version One.

Vidgen, R., and Wang, X. 2009. "Coevolving Systems and the Organization of Agile Software Development", *Information Systems Research* (20:3), pp. 355-376.

Vlaar, P. W., van Fenema, P. C., and Tiwari, V. 2008. "Cocreating Understanding and Value in Distributed Work: How Members of Onsite and Offshore Vendor Teams Give, Make, Demand, and Break Sense", *MIS Quarterly* (32:2), pp. 227-255.

Walsham, G. 1995. “Interpretive Case Studies in IS Research: Nature and Method”, *European Journal of Information Systems* (4:2), p. 74.

Walsham, G. 1997. “Actor-Network Theory and IS Research: Current Status and Future Prospects”, in *Information Systems and Qualitative Research*: Springer, pp. 466-480.

Walsham, G. 2006. “Doing Interpretive Research”, *European journal of Information Systems* (15:3), pp. 320-330.

Wang, X., Conboy, K., & Cawley, O. 2012. “’Leagile’ software development: An experience report analysis of the application of lean approaches in agile software development.” *Journal of Systems and Software*, (85:6), pp. 1287-1299.

Weber, S. 2004. The success of open source: Harvard University Press.

Weick, K. E. 2007. “The Generative Properties of Richness”, *Academy of Management Journal* (50:1), pp. 14.

West, D., Grant, T., Gerush, M., and D’Silva, D. 2010. “Agile Development: Mainstream Adoption Has Changed Agility”, *Forrester Research* (2:1), p. 41.

Winkler, M., Huber, T., and Dibbern, J. 2014. *The Software Prototype as Digital Boundary Object – a Revelatory Longitudinal Innovation Case*.

Yakura, E. K. 2002. “Charting Time: Timelines as Temporal Boundary Object,” *Academy of Management Journal* (45:5), pp. 956-970.

Yin, R. K. 2013. *Case Study Research: Design and Methods*. Sage Publications.

Zaitsev, A., Gal, U., and Tan, B. 2014. “Boundary Objects and Change in Agile Projects”, *Proceedings of 25th Australasian Conference on Information Systems, Auckland, New Zealand, 8th-10th Dec. 2014.*, Auckland: ACIS.

Zaitsev, A., and Tan, B. 2016. “Agile Project Done Right: Lessons from Project Betterproduct”, *Pacific Asia Conference on Information Systems.* AIS, Chiayi. Taiwan

Zaitsev, A., Tan, B., and Gal, U. 2016. “Collaboration Amidst Volatilty: The Evolving Nature of Boundary Objects in Agile Software Development”, *Istanbul, Turkey: Twenty-Fourth European Conference on Information Systems (ECIS)*.

Appendix A: The Interview Protocol

Basic background information

1. Name and role in this organisation?
2. How long have you worked for this organisation?
3. What kind of other roles/jobs have you had in the past?
4. Please describe the industry you are currently working in.
5. How long have you been working in the industry?
6. Please describe the organisation you are currently working for.

Software project methods

7. What kinds of software project management methods are used or have been used in this project/program/organisation?
8. How long have you been practicing these methods?
9. What is your personal opinion on the methods?
10. What other methods have you used in the past?

Software projects/program

11. What is your role in the project/program/product development? (ask if not already specified)
12. How long have you worked for this specific project/program/product?
13. How would you describe the project in your own words?
14. What are the goals of the project?
15. What kind of end results do you expect from the project?
16. What kind of challenges do you expect from the project?

The team

17. Who are the people you most communicate daily?
18. Who are the people you communicate only weekly or monthly?
19. Are there any challenges in the communication with the people you communicate the most/least?
20. What kind of formal communication rules or policies there are?
    a. Are there daily, weekly, monthly meetings etc?

21. Who do you think you should communicate more with? Less?
22. Who do you think should communicate with each other from other team members, excluding yourself?

Tools and documents

23. What kinds of communication tools are you using in your organisation?
24. Which situations are these different tools used for?
25. Which features of them do you prefer?
26. Have you used similar tools before? If yes, please describe.
27. How do the tools you are using now compare to them?
28. Do you think the tools support the methods used in the project?
29. Are you aware of any details on the contracts the project/program has with the vendors/partners?
30. If yes, please describe the contract.

Follow-up interview questions/Industry specific

Please describe the events that have taken place since we last discussed the project/program/product.

a. Are there any new practices/tools introduced?
b. Have there been any changes to the team/organisation?

31. Last time we discussed X, please tell me what happened with X during the last months?
32. Last time you told me Y. Did thing Y happen?