FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Agile Software Development in Distributed Teams with Low Spatial Distance: Challenges, Benefits and Recommendations

Diplomarbeit

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering and Internet Computing

eingereicht von

Manuel Stadler
Matrikelnummer 0926556

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuer: Thomas Grechenig
Mitwirkung: Raoul Vallon

Wien, 22. August 2016

(Unterschrift Verfasser) (Unterschrift Betreuer)

Agile Software Development in Distributed Teams with Low Spatial Distance: Challenges, Benefits and Recommendations

Master's Thesis

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering and Internet Computing

by

Manuel Stadler
Registration Number 0926556

elaborate at the
Institut of Computer Aided Automation
Research Group for Industrial Software
to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Thomas Grechenig
Assistance: Raoul Vallon

Vienna, August 22, 2016

Statement by Author

Manuel Stadler
Grundsteingasse 22 / 110, 1160 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

I hereby declare that I am the sole author of this thesis, that I have completely indicated all sources and help used, and that all parts of this work – including tables, maps and figures – if taken from other works or from the internet, whether copied literally or by sense, have been labelled including a citation of the source.

(Place, Date) (Signature of Author)

Acknowledgements

First I want to thank my parents for all their continuous support during my studies. I also want to thank the employees of the INSO institute for the guidance and supervision during this thesis. Last but not least, I want to thank all the experts who participated in the interviews and provided valuable input for this research.

Kurzfassung

In der Softwareentwicklung werden verstärkt agile Vorgehensweisen zur Strukturierung und Koordination der täglichen Arbeit eingesetzt. Methoden wie Scrum oder Extreme Programming empfehlen, alle Teammitglieder am selben Ort bzw. sogar im selben Raum zu positionieren. Gleichzeitig gibt es aber auch einen klaren Trend in Richtung verteilten Arbeitens sowie verteilter Teams. Trotz dieser beiden Gegensätzlichkeiten werden beide Aspekte immer öfter kombiniert und agile Methoden vermehrt in verteilten Teams eingesetzt. Diese Diplomarbeit untersucht, wie agile Prozessmodelle in verteilten Teams angewendet werden, welche Vor- und Nachteile diese mit sich bringen und die Herausforderungen, die dabei entstehen. Als Einschränkung dieses komplexen und weitläufigen Themengebietes wurde der Fokus explizit auf Teams, welche innerhalb eines Landes oder in Nachbarländern verteilt sind, gelegt.

Im Zuge einer Fallstudie wurden verteilte Teams mit Standorten in Österreich und Deutschland untersucht. Die empirische Datenerhebung erfolgte mittels qualitativer, semi-strukturierter Interviews mit Experten sowie Teamleitern mit praktischer Erfahrung auf diesem Gebiet. Die Analyse der gesammelten Daten zeigt klar, dass agile Methoden erfolgreich in verteilten Teams mit geringer Distanz angewendet werden können und diese sogar Vorteile mit sich bringen. Moderne Kommunikationstechnologien ermöglichen es Teams, ohne bedeutende Hindernisse über Entfernungen zu kommunizieren und erleichtern außerdem eine enge Zusammenarbeit über Standortgrenzen hinaus. Typische agile Praktiken und Kennzeichen wie kurze Iterationen, Pair Programming, tägliche Standup-Meetings, Code-Reviews oder Retrospektiven werden von verteilten Teams erfolgreich eingesetzt und bieten neben einer Verbesserung im Softwareentwicklungsprozess gleichzeitig weitere Kommunikationskanäle, welche zur Stärkung der Teamstruktur beitragen.

Häufige informelle sowie formelle Kommunikation, kurze Iterationen und regelmäßiger persönlicher Kontakt vor Ort helfen dabei, die größten Schwierigkeiten verteilten Arbeitens, insbesondere Herausforderungen in den Bereichen Koordination und Kontrolle, zu überwinden. Als Ergebnis der Forschungsarbeit werden sechs grundsätzliche Herausforderungen sowie fünf Vorteile von agilen Methoden in verteilten Teams herausgearbeitet. Abschließend werden elf Empfehlungen für Teams in ähnlichen Situationen aus den gesammelten Daten abgeleitet.

Schlüsselwörter

agil, Scrum, Kanban, eXtreme Programming, verteilte Softwarentwicklung, verteilte Teams

Abstract

Agile methodologies are facing increased adoption in software engineering teams. Established practices like Scrum or extreme programming furthermore strongly suggest to co-locate all team members in one office. Simultaneously there is a rising trend of remote working and distributed teams in today's software development community. Despite this alleged contradiction there is an increasing number of distributed teams applying agile methods. This thesis examines how agile process models can be applied in distributed teams, which challenges have to be faced and which benefits such process models can entail. To delimit the scope this thesis focuses on teams with a limited spatial dispersion, explicitly investigating teams that are distributed within one or at most across neighboring countries.

In order to achieve that the author performed a case study analyzing multiple distributed teams located in Austria or Germany. Data collection was done through semi-structured interviews of team leaders and experts who have practical experience in this area. The content analysis clearly indicates that applying agile methods in distributed teams with low spatial distance poses no problem but instead even brings forth several benefits.

Modern technology enables teams to communicate remotely without serious obstacles and allow a close collaboration across geographical locations. Teams successfully mastered common agile practices and methods like short iterations, pair programming, daily meetings, code reviews or retrospective meetings in their distributed settings which not only improve the software engineering process but furthermore pose important communication channels which strengthen the team.

Frequent informal as well as formal communication, short iteration cycles and regular face-to-face contact help to overcome problems of distributed collaboration especially when it comes to the problem areas of coordination and control. As a result this thesis brings up six challenges as well as five benefits of agile methods in distributed teams. Conclusively eleven recommendations derived from the analyzed data are presented which aim at improving the application of agile methods in such environments.

Keywords

agile, Scrum, Kanban, eXtreme Programming, distributed software development, virtual teams, distributed agile

Contents

List of Figures

List of Tables

List of Abbreviations

CI Continuous Integration

DPP Distributed Pair Programming

GSD Global Software Development

RUP Rational Unified Process

TDD Test Driven Development

WiP Work in Progress

XP eXtreme Programming

1 Introduction

This first chapter gives an introduction to the topic of this thesis, the motivation why it was chosen, and shortly introduces the reader to the structure of the thesis as well as the chosen methodology.

1.1 Problem Statement and Motivation

> „A few years ago, collocated teams were the norm, and it was unusual for a team to be geographically distributed. By now, the reverse must be true. Personally, I'm now surprised when someone tells me that everyone on the team works in the same building.“ [1, p. 355]

In today's software development world, agile processes have gained an increasing amount of attention and have faced an extensive adoption. But there is also a trend of developing software in distributed teams, using modern communication technologies which are intended to bridge challenges arising through the dispersion of teams.

The next step more and more teams are taking, is to adopt an agile way of working in distributed teams - but this undertaking also introduces new challenges, Kajko-Mattsson, Azizyan, and Magarian [2] are addressing some of those issues in their work:

> „Being in stark contrast with each other, Agile and Distributed Software Development (DSD) methods are regarded as partners in an impossible marriage. Despite this, many organizations consider them as practices worth striving for.“ [2, p. 51]

Working in a distributed team from a remote location is an increasing trend among software developers all over the world. Stack Overflow, a very well known community for software engineers surveyed this point and found that in the year 2016 around 30 percent of all software developers work at least part-time from remote locations, Figure 1.1 shows the exact numbers of that survey. This is the global number but the survey also states a number for Germany which claims that around 25 percent of German software engineers had experience with remote work. [3]

Their survey from the previous year stated in 2015 the number of remote software engineers was around 29 percent where in 2014 the percent was around 20 percent. This shows that especially in the last few years the number of remote collaboration has risen enormously. [4]

The motivation for this thesis is to shed light on how those seemingly contrasting factors - agility and team distribution - are combined in the scope of limited spatial distance between team member locations.

Furthermore, to the authors knowledge, there was no similar case study until now that focused on distributed agile software development in the Austrian and German area. Another personal motivation is the fact that the author himself is part of a software development team that is distributed over multiple locations in the Austrian area and applies agile methods and practices.

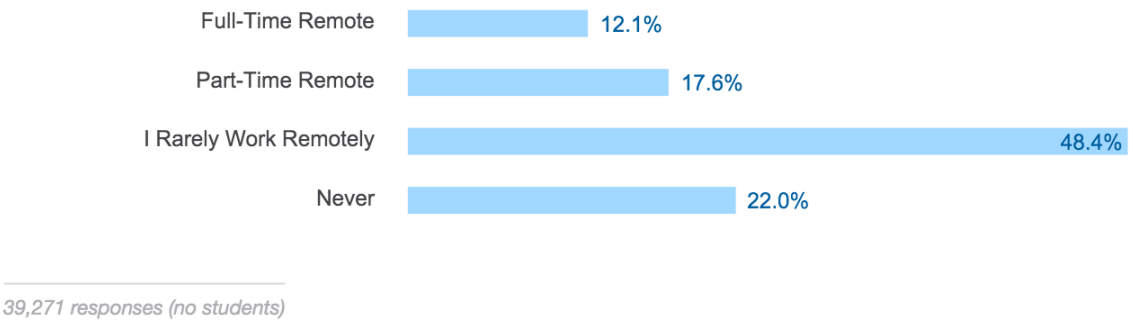

Figure 1.1: Software developers surveyed on remote work [3]

1.2 Related Work

Apart from the mere fact of teams distributing, also the adoption of agile development in distributed teams is becoming more and more popular, and there exist various case studies of different scopes analyzing the introduction, success and challenges arising. One example is the work from Pries-Heje [5] who analyzed a „project using the agile method Scrum with participants distributed across the two countries: Seven in a Scrum team in Denmark and eight in a Scrum team in India.“ [5, p. 21] They gathered qualitative data by conducting two rounds of interviews with project members in both locations.

Another case study which investigated the impact of an agile development process versus a 'classical' structured approach was performed by Estler et. al. [6]. They collected their data in two phases, using qualitative as well as quantitative methods.

> „The importance of choosing the right development process to ensure the successful and timely completion of distributed software projects cannot be understated... Or can it? This paper presents an extensive case study analyzing the impact of different development processes on the success of software projects carried out by globally distributed development teams.“ [6, p. 11]

Agile methods also entail a lot of different practices, one well known example is the practice of pair programming, which is traditionally by two individuals sharing one workstation. Flor [7] researched this practice in a distributed setting and stated that: „Remote programming pairs have the potential for cross-workspace audio, visual, and manual channels available via their computers.“ [7, p. 58]

Smite, Brede Moe and Agerfalk [8] are discussing agile development in their book *Agility Across Time and Space: Implementing Agile Methods in Global Software Projects*. This work addresses the introduction of agile development in different kinds of distributed teams and it provides insight and practical advice regarding this matter.

> „In contrast to other engineering disciplines, developing software is recognized as a significantly complex task that heavily relies on human interaction. Accordingly, distributed software projects with geographically, temporally and socio-culturally dispersed teams unavoidably experience unique pressures and challenges. There are major problems related to communication, coordination and collaboration caused by geographical, temporal and socio-cultural distance.“ [8, p. 4]

Regarding distributed teams there are several aspects that have to be considered like the different dimensions of distance (e.g. [9], [10] or [11]) or how to facilitate effective communication, discussed by [12] or [13]. The importance of communication in distributed agile teams is also highlighted by Dorairaj, Noble, and Malik [14] who performed a case study about effective communication in distributed agile teams.

Due to the fact that combining agile methods and distributed development is a rather young enterprise there is not a huge amount of basic literature available, but there is a lot of research going on as the case studies mentioned in this section indicate.

1.3 Objectives

This thesis researches agile software development in distributed teams. It investigates the different aspects, strategies and the application of agile methods in a distributed environment where a team of software developers is not gathered in one physical place but located in different sites. The main research questions this thesis tries to answer will be:

- RQ1: How can agile methods be used in distributed teams (limited to a low spatial and time dispersion)?
- RQ2: To what extent have the principles of agile methods be adopted to be applicable in such a distributed setting?
- RQ3: Which challenges have to be faced utilizing agile methods and how can those issues be handled?
- RQ4: Which benefits result from pursuing agile methods in such a distributed setting?

Approach

The first step to achieve this goal is by giving an overview and summary of current literature about agile development, especially with the emphasis on how they have to be adapted and extended to be applicable in a distributed environment with the limitations defined in the previous section.

Secondly, with these theoretical findings in mind the empirical gained knowledge is presented and examined to report the status quo in distributed software development teams. This results in an overview of how agile methods are applicable, which of its methods and practices are well suited or unqualified and furthermore give suggestions on how agile methods can be efficiently applied in distributed teams.

Focus

To delimit the scope, the thesis will focus on teams with a limited spatial dispersion. Smite et. al. [15] developed a taxonomy to classify distributed software development situations which will be used in the thesis, and therefore focus on teams that are either classified as *Onshore - Insourcing - Geographically Close* or *Offshore - Insourcing - Geographically Near*. This means that locations can be in different cities of the same country, or - if spanning multiple countries - such situations where the local time is not deviating significantly. Furthermore the author chose a further limitation by defining specific criteria that the units which are used as data source in the practical part of this thesis have to fulfill. Generally this research focuses on teams that have at least one office in Austria or Germany, the detailed constraints are presented in Section 5.3.4.

The scientific value of this thesis is to improve the understanding on how agile methods are applied in the specific area of distributed teams with low spatial distance. Which challenges arise in such particular situations and also which benefits agile methods and practices may bring. Furthermore the thesis uses the gathered knowledge to present recommendations for low-spatial distributed teams.

1.4 Structure of the Thesis

The main structure of the thesis is depicted in Figure 1.2 and is divided into a theoretical part as well as a section of empirical research. It starts with a review of literature and recent research in the fields of distributed software development teams and how agile methods can be applied in such situations.

Literature Review

The first part of the thesis will be dedicated to a literature research of relevant topics in the area of agile software development. It will analyze known agile processes and strategies. Furthermore literature and research focusing on distributed development will be examined. Chapter 2 investigates the agile software development approach as well as classic established methodologies like for example Scrum. The following Chapter 3 deals with the challenges and complexity introduced when facing a distributed team. Finally, Chapter 4 analyzes approaches and literature about distributed agile methods and research that examined virtual teams using methodologies like Scrum or eXtreme Programming (XP).

Case Study

This starts at Chapter 5 and represents the practical part of the thesis where a case study about different distributed teams which have adopted an agile development style will be conducted. The case study design and execution will follow the guidelines of Yin [16] and Runeson et. al. [17] and empirical data will be collected through semi-structured interviews. The case study is divided into the following parts:

After an introduction about case study research and the application of this method in the field of software engineering Section 5.2 describes the design of the case study. Following, Section 5.3 puts forward the data collection approach that was chosen and introduces the different units of analysis. After setting up the frame of the study and gathering the empirical data, Section 5.4 explains how the qualitative data was analyzed and presents the structure how the analyzed data is presented to the reader.

Chapter 6 then lists the analyses of all the reviewed cases, the contained sections illustrate the sorted and analyzed data from around one hundred pages of transcribed interview material that was gathered during the data collection phase.

Results

Finally, in Chapter 7, Section 7.1 summarizes those cases in a cross case analysis of the analyzed case study units. This is followed by Section 7.2 which uses the new gained insights to discuss the research propositions defined in Section 5.2.5. The following Section 7.3 answers the research questions defined in section 1.3. Finally, Section 7.5 lists recommendations that are derived from the collected insights to improve collaboration in distributed teams.

The thesis ends with the conclusion in Section 8 summarizing the performed research and providing an outlook of the future within this field of research.

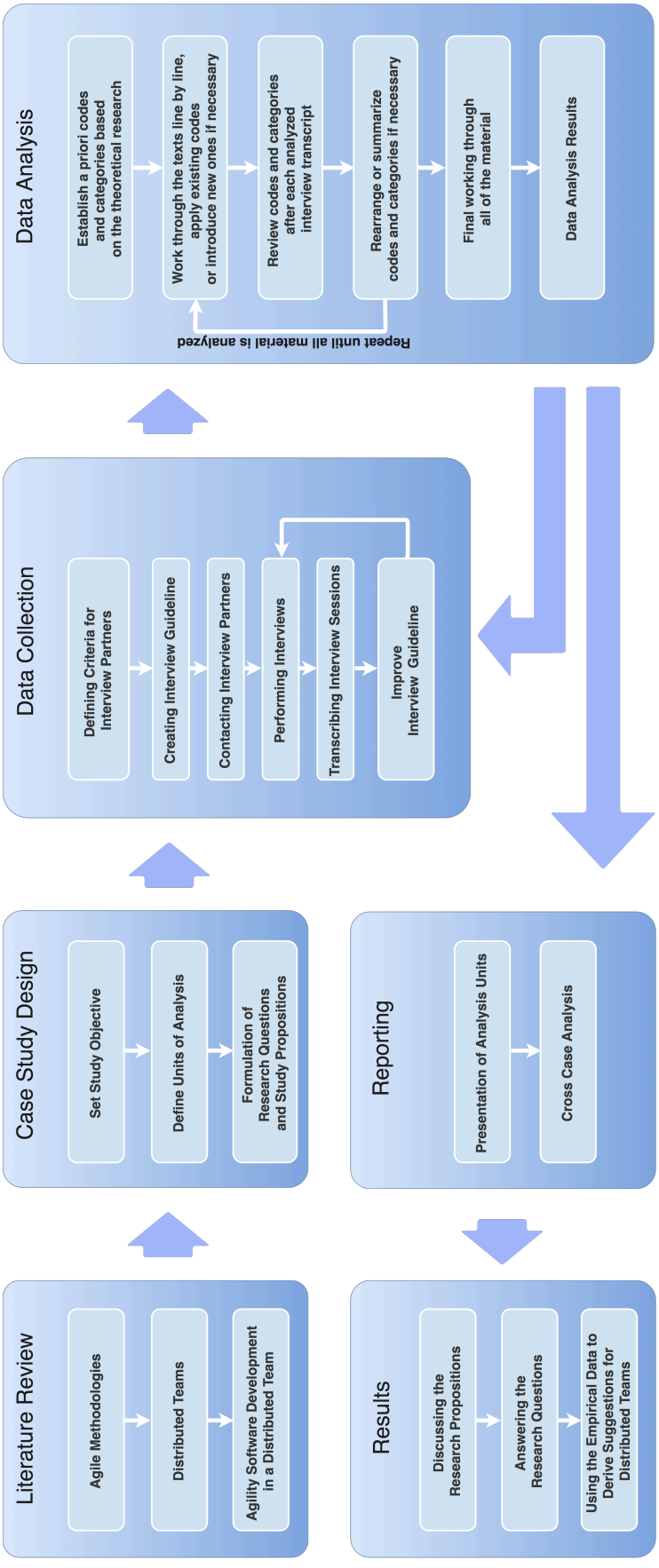

Figure 1.2: The structure of this thesis

2 Agile Software Development

2.1 Introduction

> „Agile development is a philosophy. It's a way of thinking about software development.“ [18, p. 9]

Traditional software development methods often begin with the determination and documentation of requirements. They usually follow a rigid plan and are often not flexible enough when it comes to dealing with shifting requirements and a constantly changing environment. That led to many developers becoming frustrated with this initial fixation on documentation tasks, but not only developers had problems dealing with the constantly changing setting, also customers often had troubles stating their exact need up front. [19, p. 1-7]

Those challenges motivated practitioners to develop new strategies, methodologies that do not reject but rather embrace a changing and evolving environment. Williams and Cockburn [20] explain that such approaches were simultaneously developed on three different continents - and while they have been authored independently they still share the same fundamental principles. Namely, those three approaches were „the Dynamic Systems Development Method in Europe; Feature-Driven Development in Australia; and Extreme Programming, Crystal, Adaptive Software Development, and Scrum in the US.“ [20, p. 39]

Finally, seventeen people of those who were working on the different approaches met in Utah - and what emerged from this conference was the Agile Manifesto, with the main component being four comparative values which represent the foundation of the agile position. [20, p. 39]

Since its introduction, the word *agile* has become a collective term describing different processes and management techniques used by software developers to coordinate and execute their work. Agile processes describe various aspects of software engineering and are applicable in a wide range of projects. Dingsøyr, Dybå, and Moe [21] are describing agile development methods as a reply to traditional software or plan based development methods which rely more on a formal approach. [21, p. 1]

Augustine [22] characterizes agility as a method to face the turbulent business environment most businesses are facing nowadays. He sees it as a way to encounter increasing project unpredictability and the ability to adapt to change while still being able to deliver a high amount of customer value.[22, p. 20-21]

2.2 The Agile Manifesto

Manifesto for Agile Software Development

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.

Kent Beck	James Grenning	Robert C. Martin
Mike Beedle	Jim Highsmith	Steve Mellor
Arie van Bennekum	Andrew Hunt	Ken Schwaber
Alistair Cockburn	Ron Jeffries	Jeff Sutherland
Ward Cunningham	Jon Kern	Dave Thomas
Martin Fowler	Brian Marick	

These statements from Beck et al. [23] were authored in 2001 and build the foundation of the agile movement widely known as the *agile manifesto*. Its main content are four statements as well as twelve principles which describe the idea of the agile process.

Individuals and Interactions over processes and tools. This first statement emphasizes the importance of individuals and their cooperation, in the end it is them who build software and products. While processes and tools sure are useful by providing support and improving efficiency, it is the knowledge and skills of the people which are needed to produce results. Furthermore, developers are often impeded and tied down with unnecessary procedures. In agile project management processes should not dictate a teams actions but should support them. Processes have to be adapted to suit the team and not vice versa. [24, p. 13-14]

Many agile principles support that idea of people first, before implementing new processes it is crucial to have the team accepting them. It is most important to understand people within a team, how they cooperate and how every individual's work may have an impact on other team members. [25, p. 34]

Working Software over comprehensive documentation. In software projects there is a very high amount of things that could be documented, but it is very uncertain what is really needed in the future - which documentation is actually read by someone and what will just get dusty. In an agile environment, working software is preferred over documentation. Frequently presenting and delivering versions of a real product is the essential idea of this argument, because not documentation or specification documents but real working software creates the most value for shareholders. [24, p. 12]

Stellman and Greene [25] argue this point in the following way: „To an agile practitioner, working software is software that adds value to the organization. It could be software that a company sells to make money, or it could be software that people who work at the company use to do their jobs more efficiently. For a project to add value, it needs to deliver or save more money than it cost to build.“ [25, p. 35]

But that does not mean that documentation is useless or should be avoided. The important factor is the nature of documentation. Highsmith [24] argues that there is often an essential flaw in the kind of documentation, he states that it is very important how documentation is created: On the one hand there is the situation where the documentation is a *by-product of the interaction* (for example a scenario where two people sit together and conjointly develop a specification). In the second case, the documentation is a *substitution for interaction* which may impede progress. An example for this case would be a a product manager creating a requirements document which then is sent to a development team. [24, p. 12]

Customer Collaboration over contract negotiation. The customer in agile development can be defined as that party that either generates business value out of the created product or - in case of a retail product - the person using it. The function of the customer is to define value, while other stakeholders are defining constraints. In a new, uncertain and complex product, the relation between customer and developer is very important to the products success and should be collaborative rather than be guided by contract disputes. [24, p. 13]

Responding to Change over following a plan. Projects always have aspects that are unknown or uncertain. Furthermore, today's software development takes place in a constantly and rapidly changing environment, it is not only scope creep that has to be dealt with, furthermore it is nearly everything that is subject to change may it be technology itself, features, scope or architectures. This constant progress is the reason why it is often hard to just follow a fixed, made up front plan. Highsmith [24] furthermore specifies this statement with the following points that should be pursued [24, p. 10-11]:

- Envision-Explore versus Plan-Do
- Exploration versus production
- Adapting versus anticipating

Agile methods accept that change is unavoidable and therefore pursue adaptive planning strategies and also include various strategies for feedback loops to improve quality. One example would be to create plans on a high level in the beginning which are reworked later into more particular plans once more information has become available. [26, p. 3]

That is, while there is value in the items on the right, we value the items on the left more. The goal of this statement is to clarify that there is a big difference between one thing being *unimportant* and one thing being *more important* than another one. The aspects on the right of those four values are not unimportant at all. The right tools play an essential role in the whole software development process and determine the speed of progress and therefore also the costs. Also contracts are important for stability and the relationship between customers and developers. [24, p. 10]

Therefore the aspects on the right are still rather important but: „the items on the left are the most critical. Without skilled individuals, working products, close interactions with customers and responsiveness to change, product delivery will be nearly impossible“ [24, p. 10]

2.2.1 Agile Principles

Beside the four values presented in the previous section, the agile manifesto furthermore lists twelve principles. They can be divided into four main topics they cover: delivery, communication, execution and improvement. [25, p. 52]

1. *Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.*

   The main ideas of this principle are „releasing software early, delivering value continuously, and satisfying the customer“ [25, p. 55] This principle is also one of the reasons why agile methods are iterative: Teams plan early releases to acquire feedback from their customers and use that feedback to improve their product.

2. *Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.*

   This practice often sounds easier in theory than when it comes to real world situations. When change requests that require a lot of additional work come up it can get emotionally inside a team, especially when additional requirements and changes are not respected in deadlines set by superiors. It is important to get rid of the culture to blame developers for occurring delays and that a team together with the customer owns the requirements collectively. [25, p. 58]

3. *Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.*

   This practice extends the previous one and helps a team embracing change by frequently delivering working software. Such iterations represent regular deadlines where a demo is shown to the customer and again feedback can be used to adapt the course of the project and include new or changing requirements early. [25, p. 61]

4. *Business people and developers must work together daily throughout the project.*

   It is important to involve the customers and business people into the development process. This principle comes back to the core agile value of customer collaboration over contract negotiation. To achieve a good result it is necessary to regularly discuss the product that is produced, and it is important that „the agile team collaborates with the customer (typically the product owner) as someone with an equal say in the way the project is run.“ [25, p. 70]

5. *Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.*

   When every team member understands the value of the software they are building projects run best. On the other hand when people are not rewarded for building the software or do not understand its value, projects may break down. It is for example counteractive to reward programmers if no bugs are found during code reviews and even worse to penalize found issues. [25, p. 70]

6. *The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.*

This principle states that face-to-face communication is one of the best ways to share ideas. But this does not mean that documentation is not necessary, instead documentation also represents a form of communication. „That's why agile communication practices focus most on individual people communicating with each other, and reserve documentation for those cases where complex information needs to be recalled in detail later." [25, p. 67]

7. *Working software is the primary measure of progress.*

   Instead of using status reports to communicate the status of a project, this principle suggests to use working software for this endeavor. „By delivering working software at the end of each iteration, and by doing a real product demonstration that shows everyone exactly what the team did, they keep everyone up to date on the progress of the software in a way that is almost impossible to misread." [25, p. 75]

8. *Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.*

   It is important to plan what is delivered at a sustainable pace, which is working better when the development is done in iterations, because it is easier to estimate the work that can be done in the next two or three weeks than it is to estimate the possible work that can be done in the next year. [25, p. 76]

9. *Continuous attention to technical excellence and good design enhances agility.*

   Wrong estimates are not the only thing that can bring a project out of balance, also bad design which turns a seemingly easy part of software code into very hard pieces of work can mess up schedules. Therefore it is advised to build up good coding habits and take enough time to write good source code. Taking a bit more time during the coding phase to avoid bugs saves a lot of time on the long run by saving much more time it would cost to fix issues during a later stage. [25, p. 77]

10. *Simplicity -the art of maximizing the amount of work not done- is essential.*

    Every additional code added to an existing project normally increases its complexity and dependencies between services and systems make code even more complex and increase the difficulty of changing it. This principle advertises to always just add the minimum amount of work to complete a task successfully. Avoiding dispensable features is often cheaper in the long run because the maintaining costs are often higher than the actual value. [25, p. 79]

11. *The best architectures, requirements, and designs emerge from self-organizing teams.*

    In an agile project there is no explicit requirements and design phase, instead the team continually meets and revises the project plan. Every team member is responsible for the architecture, and instead of creating a big design upfront an agile team incrementally designs their project. „In fact, when a team is building the software piece by piece, starting with the most valuable chunks, the architect's job becomes more challenging (but often more interesting)." [25, p. 80]

12. *At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.*

    Constantly improving the way of working is a key element of agile teams. Reviewing past projects or project phases and using those insights to improve in the future is essential. This requires honesty and the ability to identify and accept shortcomings because that is the only way to become more capable in the future. [25, p. 80-81]

2.3 The Agile Team

2.3.1 Self-Organization

Self-Organization is a fundamental requirement for agile teams. Self-organization means that a team can decide for itself how to best achieve the goals given to it. This means that every team will choose different ways of organizing itself, because the way that works best for the team is depending on multiple factors like the ability and experience of single team members, team size and of course the project goal. Furthermore, letting a team decide the internal structure instead of one manager giving the orders strengthens the commitment of the team to „fully own the problem“. [1, p. 189]

A Scrum team can also be characterized as a *complex adaptive system* which is: „A system with many entities interacting with each other in various ways, where these interactions are governed by simple, localized rules operating in a context of constant feedback.“ [27, p. 404] In this context, self-organization is a „bottom-up emergent property of a complex adaptive system whereby the organization of the system emerges over time as a response to its environment.“ [27, p. 416]

The importance of this aspect in real project teams is also reflected in the findings of a survey study from Chow and Cao [28] which identified the self-organization aspect as one of the critical success factors to agile teams.

One of the main aspects to enable self-organization within a team is frequent communication, a factor also emphasized by the following quote:

> „A self-organizing team, on the other hand, does a huge amount of communication—but not those endless, useless status meetings that developers often hate being forced to attend. Instead, team members decide for themselves what they need to talk about in order to do the project right.“ [25, p. 83]

2.3.2 Co-Location

Many agile frameworks strongly demand co-location of teams and while not being an inevitable requirement most authors strongly recommend to co-locate the development team if possible in any way, e.g. [22, p. 132], [29, p. 199-201], [18, p. 45], [30].

On a closer look, co-location is nowhere claimed explicitly in the agile manifesto itself nor in one of its 12 principles, but the reason why it is mentioned with such tenacity is after all originating from one of them:

> „The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.“ [23]

One very well known methodology that insists on co-location is XP, while not strictly enforced in one of the values or principles of XP it is listed as one of the primary practices called "Sit Together". Beck and Andres [30] there states his insight of how he learned „how important it is to sit together, to communicate with all our senses.“

Also Shore and Warden [18] name co-location as their third prerequisite (out of a total of six) that should be met if a team wants to apply XP as development methodology: „XP relies on fast, high-bandwidth communication for many of its practices. In order to achieve that communication, your team members needs to sit together in the same room.“ [18, p. 45]

Like a lot of other literature about agile methods, they have also dedicated a whole chapter on the importance of co-location and spatial arrangement of an agile team. The main argument states that all communication apart from a direct face-to-face conversation disturbs the process and flow of work. This goes as far as providing sample workspace arrangements with workstations close to each other accompanied by pairing stations in the same room. [18, p. 112-119]

Schwaber and Beedle [19] also suggest to use *open working environments* which allow easy communication. They state that „silence is a bad sign“ and the presence of communication is an indicator on how well the team is doing. [19, p. 39]

Another example is Crystal Clear - an agile software development methodology that is part of the Crystal family - where co-location is considered a critical element of success. Crystal Clear comprises a rule stating that all team members have to sit in the same location or in at least adjacent rooms. [31, p. 88]

Its author Alistair Cockburn furthermore promotes the concept of *osmotic communication* which means that the flow of information is not just direct between two individuals but also reaches other team members in the background „so that they pick up relevant information as though by osmosis.“ [32, p. 24] He furthermore argues that this concept is hard to achieve without having the team located in the same room. In his opinion modern technology may provide a certain level of close communication but he also states that he has never witnessed real osmotic communication in a physically dispersed team. [32, p. 24]

But the insistence for co-location is not a claim confined to the agile movement, Teasley et al. performed a field study about the effects of co-location of software development teams where they "radically collocated" the project teams, so that everybody was „located in a single physical room.“ [33, p. 671] Their findings suggest that co-location „brings interactive, continuous communication, which allows overhearing and awareness of teammates' activities which helps in clarification, problem solving, and learning. It also enhances team building.“ [33, p. 681] Furthermore they stated that within their co-located setting the groups „showed significantly improved productivity and high levels of satisfaction by everyone involved, from team member to customer. The significant improvements in productivity over the company baseline are most likely due to the tight fit between the development method (timeboxing) and the collaborative facilities.“ [33, p. 680 - 681]

2.4 Scrum

2.4.1 Introduction

> „Scrum is different. Work feels different. Management feels different. Under Scrum, work becomes straightforward, relevant, and productive.“ [19, p. 23]

Scrum - being one of the most famous agile principles - is a framework for software development whose main components are displayed in Figure 2.1. It is not a standardized process that can be followed step by step, but a collection of practices based on the agile values. And while there are defined roles and processes, each organization will add unique characteristics and adaptions. So when applying Scrum it is not possible to change the fundamental principles, values and practices, but the structure of single processes may be adapted to individual needs. [27, p. 14]

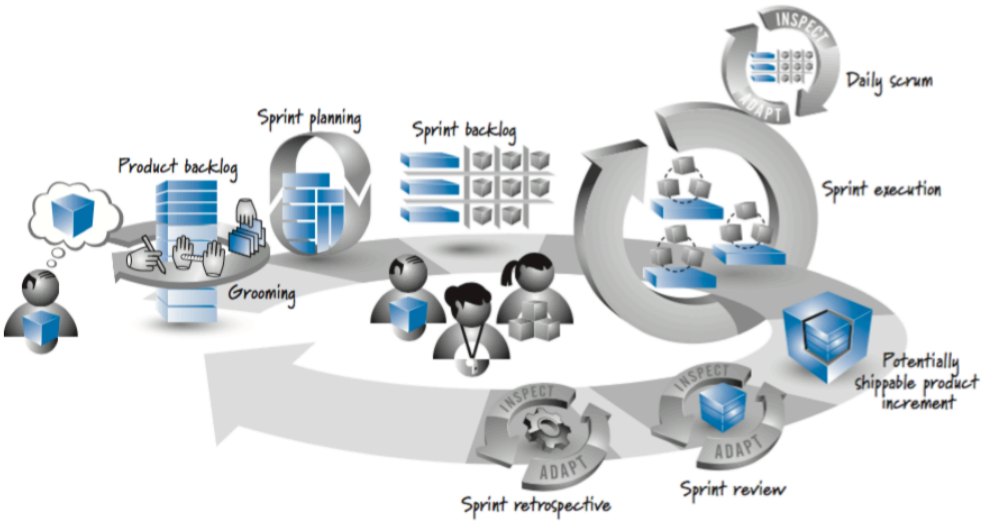

Figure 2.1: Scrum framework [27, p. 17]

2.4.2 Roles

Scrum distinguishes three different roles that together form a Scrum team:

The Product Owner is in charge of the Product Backlog, he is the authority who decides which features are built, ranks the priority of those work items and defines the approval conditions. Furthermore, the Product Owner constantly communicates a clear vision of the final product and the bigger picture and is in continual contact with the stakeholders of the project. [27, p. 15-16]

> „On a Scrum team, the Product Owner is the person who made the commitment to the company. He's the person who has to stand up and promise something specific that will be delivered at the end of the project." [25, p. 95]

The product owner is authorized to make decisions about the product, he answers questions about details of the product from the Development Team, and has the responsibility to communicate a clear vision of the product to the other team members. The Product Owner therefore meets with the other team members regularly in the Daily Scrum meeting and also is in charge of prioritizing the Backlog. He is the link between the external stakeholders and the development team. Normally the Product Owner does not have all the information because there usually are multiple stakeholders, so he has to be in constant contact with them to stay informed and be able to provide answers for the development team. [25, p. 95-96]

The Scrum Master acts as coach and guide through the whole Scrum process, he helps understand principles and practices and also is in charge to resolve issues and adjusts and improves processes. The Scrum Master takes a leading role when impediments that cannot be resolved by the Development Team alone have to be removed, he generally is a servant

rather than leading authority to the team. Generally the Scrum Master does not have authority to control the team, so he is not the same as a project manager but more of a leading figure. [27, p. 16]

> „How the Scrum Master does his job makes the biggest difference between traditional command-and-control project management and an agile Scrum team.“ [25, p. 94]

In Scrum there is no separate role which owns the plan and schedule for a project, while the development team just follows orders. Instead the Scrum Master assists the team creating a plan and schedule, this way all team members own the plan together and it prevents single team members feeling not responsible when obstacles occur. [25, p. 94]

The Development Team consists of - typically five to nine - cross-functional people who are responsible for designing the product, and also building and testing it. The Development Team is self-organizing and together as a collective should possess all the needed skills for completing the required tasks. Although it is possible to have larger teams, it is recommended to split those large team up into multiple smaller Scrum Teams. Furthermore the team has the full authority to self decide on how to accomplish the goals set by the Product Owner. [27, p. 16]

> „Teams as small as three can benefit, but the small size limits the amount of interaction that can occur and reduces productivity gains. Teams larger than eight don’t work out well. Team productivity decreases and the Scrum’s control mechanisms become cumbersome. (...) Most importantly, large teams generate too much complexity for an empirical process.“ [19, p. 37]

The survey of Ambler [34] in 2008 reported that around 30% of teams are sized between one and five people, while nearly 40% of teams have a size between 6 to 10 people. This shows that Scrum teams in practice follow the original recommendations.

2.4.3 Artifacts

Scrum uses the following artifacts in its process:

Product Backlog The Product Backlog is a prioritized list of work items (often in the form of user stories) for the whole project. The format of the contained items is not strictly defined:

> „The Product Backlog represents everything that anyone interested in the product or process has thought is needed or would be a good idea in the product. It is a list of all features, functions, technologies, enhancements, and bug fixes that constitute the changes that will be made to the product for future releases. Anything that represents work to be done on the product is included in Product Backlog.“ [19, p. 33]

An often chosen form to describe the elements of the Product Backlog are so called user stories. User stories are designed to be understandable for all stakeholders, software developers as well as business people. They are of a simple structure and can be formulated at various granularity levels. Often they follow the template: *"As a <user role> I want to <goal> so that <benefit>."*

User stories have the benefit of being very conversational, discussing a user story often enables a better form of information exchange than just relying on written requirements. This does not mean there is no written documentation, discussing user stories often may result in additional documentation like sketches or references to other documents. [27, p. 83]

The Product Backlog is a constantly changing and evolving artifact, work items can be added, removed and altered as conditions of the project change. Each item has to be estimated in its size and costs and then prioritized depending on multiple factors including costs, value and risks. As a size estimate teams often use relative measurements like Story Points. The person in charge of the Product Backlog is the Product Owner, it is his duty to constantly maintain and refine its items and keep the priority of the single items correct. [27, p. 19-20]

Sprint backlog At the beginning of each development iteration some of the work items from the Product Backlog are moved to a Sprint Backlog. Usually those work items with the highest priority are chosen, but which and especially the amount of items is a decision made in the beginning of each development iteration. Those items that are still not completed at the end of the iteration are moved back to the Product Backlog. [13, p. 5]

When a user story from the Product Backlog is selected for a Sprint, it is pulled to the Sprint Backlog and discussed with the Product Owner to ensure clarity on what it means and what are the acceptance criteria for it. User stories are often broken down into multiple smaller tasks which are estimated and prioritized as well. [25, p. 144]

Product The Product is the constantly evolving result from development work, and is presented at the end of each Sprint (Rubin phrases this as *potentially shippable product increment*). [27, p. 18]

2.4.4 Activities

Scrum knows multiple activities and routines that are part of the iterative process:

Sprint In Scrum, work is organized in iterations with a defined length, typically between one and four weeks. Sprints are time-boxed, which means they have a specific start and end date. Furthermore Sprints should all be of equal length, and normally any goal-altering changes or changes in personnel are permitted during a sprint. Within one Sprint there are several activities taking place, most importantly the Sprint planning, Sprint execution, Sprint review and the Sprint Retrospective. [27, p. 61-62]

Originally the proposed duration for a Sprint is thirty calendar days. Schwaber and Beedle [19] argues that this is the time a team needs to get hold of a problem and is able to bring forth a product increment. But he also states that while this thirty day period is a good compromises between competing pressures that „Adjustments can be made to the duration after everyone has more experience with Scrum.“ [19, p. 52] Other experts in Scrum do not name one fixed iteration length but state that Sprints „must also be short, somewhere between one week and a calendar month in length.“ [27, p. 62]

Sprint Planning The Sprint Planning is always conducted at the beginning of a new Sprint. It basically is a meeting where the Development Team, the Scrum Master and the Product Owner discuss the remaining items in the Product Backlog and define which work items should be accomplished in that upcoming iteration. There are several techniques to effectively analyze

a work item, typically they are broken down into multiple tasks which are then estimated on how much time their implementation would take. [27, p. 22]

There are two approaches for doing Sprint Plannings:

Two-Part Sprint Planning
In this approach the Planning is divided into two parts, the *what* phase and the *how* phase. In the first part the team determines its capacity of work it can do within the sprint and then together with the Product Owner selects the work items that will be worked on in that Sprint. During this selection the Product Owner also discusses the items and answers eventual questions to make sure everybody fully understands the task.

When the selection of work items is done the team breaks them up into multiple smaller individual tasks in the second phase. This can also be done with the Product Owner's help. Those smaller tasks are estimated and checked against the capacity the team estimated in the previous part. Depending on that outcome the forecast of the Sprint is adjusted. [27, p. 339]

One-Part Sprint Planning
Alternatively to the two phase approach, in this approach the team at the beginning determines its capacity of work it can complete within the Sprint. Based on that, an item from the Product Backlog is selected together with the Product Owner and the team checks if it can commit to the goals of the Sprint. This selection of items from the Product Backlog is repeated until the determined capacity is reached. [27, p. 340]

After finalizing the items for the Sprint Backlog and at the end of the Sprint Planning, the team finalizes its commitment to what it will deliver at the Sprints end. This commitment is embodied by the selected work items from the Product Backlog. [27, p. 346]

Sprint Execution This phase is where the actual work is done to complete the tasks of the current Sprint Backlog. There is no fixed rule in which way and order the tasks have to be completed, this organization is up to the team. [27, p. 22]

Daily Scrum The Daily Scrum is a meeting where the Development Team gathers every working day and it is preferably hold always at the same time. It is the Scrum version of the daily stand-up meeting from XP, and each team member essentially answers three questions: [27, p. 24]

- What did I accomplish since the last daily Scrum?
- What do I plan to work on by the next daily Scrum?
- What are the obstacles or impediments that are preventing me from making progress?

There are two main purposes for this meeting: First it gives feedback about the work a team is currently doing and the progress they make. Secondly it gives the team the opportunity to make adjustments and plannings on a very short notice, so a team can plan work for the upcoming day. This time to make decisions is sometimes also called the „last responsible moment“. [25, p. 110]

There are different opinions on who exactly should actively participate in the Daily Scrum, and who should just be in an observing role. Rubin [27] holds the opinion that everyone on the Scrum Team should participate in the Daily Scrum and should have the right to speak. [27, p. 24-25]

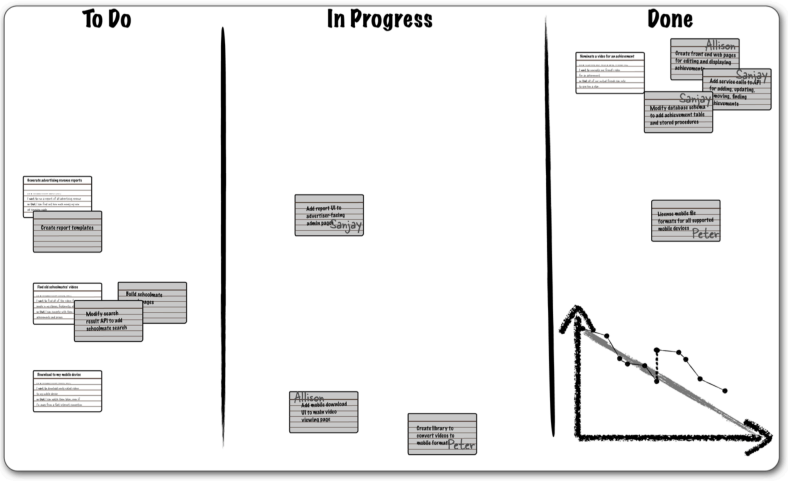

Figure 2.2: Task board, often used by agile teams [25, p. 37]

Sprint Review At the end of each Sprint there are two activities of inspective nature. The goal of the first one - called the Sprint Review - is the inspection and adaption of the built product. Participated by the whole Scrum Team as well as external stakeholders, the focus is on reviewing the completed work from the current Sprint. It should serve the external stakeholders who gain insight in the development process and progress of the product as well as the Scrum Team members who „gain a deeper appreciation for the business and marketing side of their product by getting frequent feedback on the convergence of the product toward delighted customers or users.“ [27, p. 26]

Retrospective The Retrospective is the second activity held at the end of each sprint. In contrast to the Sprint Review that is focusing on the product, this review is focusing on the process. The Scrum Team discusses what worked well in the Sprint and which aspects did not work well or need improvements. This results in a number of process improvements and changes that are applied in the next Sprint. [27, p. 26-27]

2.4.5 Tools

Scrum (and generally agile methods) also use various tools to improve communication and collaboration. The presented tools in this section are very common in agile environments and „most teams use a combination of a task board and a burndown and/or burnup chart as their principal information radiator.“ [27, p. 356]

Task Board To help keeping track of tasks and schedules teams often use task boards to visualize work items (depicted in Figure 2.2). Each element on such a task board typically represents a user story (the concept of user stories is described in Section 2.4.3) and is written on a sticky note or index card and attached to the board. Such a visual representation improves collaboration and gathering in front of the board, moving stories, talking and gesturing is a very intensive form of communication. [25, p. 37]

Burndown Chart Another common practices is the usage of burndown charts, an example depicted in Figure 2.3. Team members update the estimates of the amount of effort that remains for each yet uncompleted task once a day. The sum of the remaining estimated hours

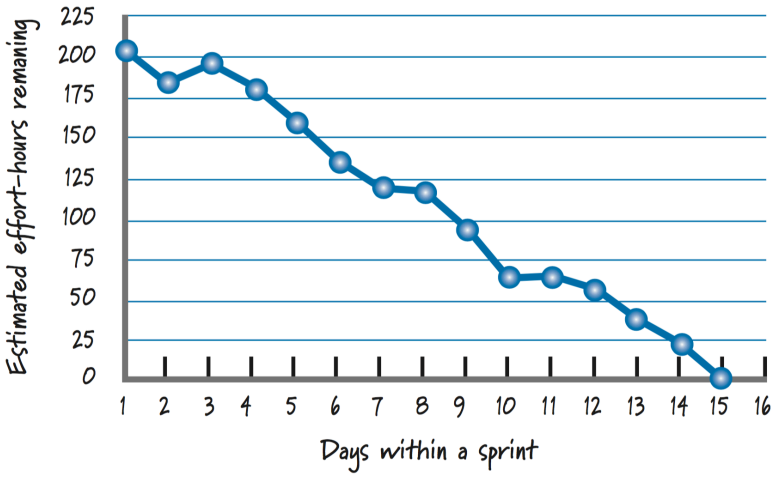

Figure 2.3: Sprint burndown chart [27, p. 358]

are then entered into the chart, and the resulting graph over time displays the progress during the Sprint. It is of course possible that some estimates need correction and the graph is not always strictly decreasing. In Sprint burndown charts the vertical axis are commonly annotated with the remaining hours of work, and there also exists a release burndown chart which commonly shows ideal working days or story points on the vertical axis. The horizontal axis shows the days of the Sprint in the case of a Sprint burndown chart and the Sprints in case of a release chart. [27, p. 358]

2.4.6 Team Organization

The ideal team size is suggested to be „seven people, plus or minus two“. [19, p. 36] Teams may also be smaller and a three people team can be beneficial but in the same time the small size tends to limit the number of interactions and may reduce productivity. Having a team larger than eight people also decreases productivity and makes organizing meetings more difficult. [19, p. 36-37]

But Scrum is not only fit for small projects requiring less than ten people but the way the process scales differs from other approaches. Instead of simply increasing development team size, Scrum favors creating multiple Scrum teams. However, when there are multiple Scrum teams it is required to have a way of effectively coordinate them. This coordination is the task of the so called Scrum of Scrums. [27, p. 218]

Scrum of Scrums

While the Daily Scrum is a meeting attended only by the members of that particular development team, the Scrum of Scrums is a meeting attended by individuals from different teams. Each development team chooses one member who represents it in that coordination meeting. Although some prefer to always have the same person on the Scrum of Scrums it is not a requirement and the representing individual may as well change depending on who is suited best to speak for the rest of the team. In addition, some teams send their Scrum Master in addition to the normal representative. Typically this meeting is not held every day but a few times a week, depending on

necessity and similar to the Daily Scrum the duration is often limited to not more than 15 minutes. Also the questions discussed are similar but on an other level [27, p. 218-219]:

- What has my team done since we last met that could affect other teams?
- What will my team do before we meet again that could affect other teams?
- What problems is my team having that it could use help from other teams to resolve?

2.5 Kanban

Kanban is a technique for project management that is based on a Just-In-Time and pull-driven production mechanism originally developed by Toyota and was adapted in a software engineering team in 2004. [35]

2.5.1 Principles

Kanban is a rather young methodology and therefore still evolving. Originally based on three principles, those have been extended to six core practices [36]:

Visualization
One of the main principles of Kanban is making information visible to the team. The foundation of this visualization and the whole process is the Kanban board.

Limit the Work in Progress (WiP)
Limiting the work that is done simultaneously is a further principle. While Scrum limits the WiP by defining time boxed iteration cycles, Kanban sets a limit to the number of work items that are allowed in each step.

Manage the Workflow
The first two practices also build the foundation for this aspect which aims at improving the way work items take while being processed. This is not a one time activity but more an everlasting task.

Make Process Policies Explicit
Having defined processes instead of implicit assumptions helps broaching such subjects within a team. Which in turn helps detecting problems, guard against confusion and improve workflows.

Implement Feedback Loops
Similar to the Retrospective meetings in Scrum, Kanban also includes practices for getting feedback on the process itself.

Improve Collaboratively, Evolve Experimentally
This practice encourages the team to try new practices as a team and improve and evolve the own process.

Figure 2.4: A Kanban board

2.5.2 Kanban Board

The Kanban Board is the basis for visualizing information and is also an information radiator. It can be a physical board, paper cards on a wall or also a digital board available from some process management software. Figure 2.4 shows how a typical Kanban board can look like.

> „The Kanban board provides visibility to the software process, because it shows assigned work of each developer, clearly communicates priorities and highlights bottlenecks. Additionally, its goal is to minimize WIP, i.e. develop only those items which are requested. This produces constant flow of released work items to the customers, as the developers focus only on those few items at given time.“ [35]

The board is divided into multiple vertical sections which represent the stages in the workflow. The colored items in it represent the work items for a project. The color can be used to either represent a prioritization or can also stand for a type of work. Those work items are then moved from left to right, depending on its state. The number in the header defines the limit of items that are allowed in each column, and some columns are divided into a *Doing* and a *Done* sub column to signal a more detailed state in that stage. Hammarberg and Sunden [36] also recommend to define acceptance criteria for each column, for example that a development task has to be reviewed by another developer before it is moved to the *Done* sub column.

2.6 Extreme Programming

> „Extreme Programming (XP) is about social change. It is about letting go of habits and patterns that were adaptive in the past, but now get in the way of us doing our

> best work. It is about giving up the defenses that protect us but interfere with our productivity. It may leave us feeling exposed.“ [30]

XP is a software development methodology that was designed to adapt to fast changing requirements and vague environments. It is based on short development cycles, teamwork and communication, shared values and various programming techniques. [30, p. 1]

2.6.1 Values

The philosophy of XP is based on some basic values which are the foundation of specific practices.

> „Values and practices are an ocean apart. Values are universal. Ideally, my values as I work are exactly the same as my values in the rest of my life. Practices, however, are intensely situated. If I want feedback about whether I'm doing a good job programming, continuously building and testing my software makes sense. If I want feedback when I'm changing a diaper, “continuously building and testing” is absurd. The forces involved in the two activities are just too different.“ [30, p. 14-15]

Communication is one of the most important aspects in XP teams. When problems occur it is often that case that somebody else already knows the solution, the essential part is to get that knowledge through to the location where it is required. Intense and frequent communication can also reduce the amount of needed documentation and prevent misunderstandings. [37, p. 5]

Simplicity means that XP strives to always find the most simple solution. The guiding question is „What is the most simple thing that could possibly work?“. This must not be confused with compulsory reducing complexity but it should always be considered in its context. Also this aspect is always connected with communication, Beck and Andres [30] state that: „Improving communication helps achieve simplicity by eliminating unneeded or deferrable requirements from today's concerns. Achieving simplicity gives you that much less to communicate about.“ [30, p. 19]

Feedback Getting feedback is an essential aspect of software development, it improves quality and is happening on multiple different levels. Getting new versions of a product frequent and fast to a customer or writing tests for software components are two examples which generate feedback that enables quick adaptions to flaws and errors. [37, p. 5]

> „XP teams strive to generate as much feedback as they can handle as quickly as possible. They try to shorten the feedback cycle to minutes or hours instead of weeks or months. The sooner you know, the sooner you can adapt.“ [30, p. 20]

Courage is taking action in the face of fear. It requires courage when striving for simple solutions since there are always things that have to be dropped. Feedback requires courage because giving or receiving negative feedback may be understood as personal criticism or attack. [37, p. 5]

> „If courage alone is dangerous, in concert with the other values it is powerful. The courage to speak truths, pleasant or unpleasant, fosters communication and trust. The

courage to discard failing solutions and seek new ones encourages simplicity. The courage to seek real, concrete answers creates feedback.“ [30, p. 21]

Respect is the point that is the foundation of the previous four values. XP will not work successfully if individuals in a team do not care for their team members or their project. To improve software development in terms of productivity as well as humanity each contribution of team members has to be respected. [30]

2.6.2 Principles

Since the distance between those values and concrete practices, there exist several principles[1] that build the bridge between those two. The principles of XP are „a set of domain-specific guidelines for finding practices in harmony with XP's values.“ [30, p. 22] Those principles are shortly listed below, and can be read in more detail in the original book about XP from Beck and Andres [30].

Humanity Software is developed by people. This principle states that individual needs are as well very important for a project's success.

Economics Making sure that actions have value and that they serve business needs.

Mutual Benefit Activities and solutions should not be at the expense of others and maintaining good relationships between individuals is important.

Self Similarity When a solution works try to use it again, also if the context differs try to copy and apply it at a different scale.

Improvement Nothing is perfect therefore it is necessary to continuously improve processes, designs or stories.

Diversity Alikeness is not effective, a team needs to be diverse and should unite individuals with various skills, perspectives and attitudes.

Reflection Do not just finish tasks but analyze why something failed or succeeded.

Flow XP favors a steady flow of activities over discrete, separated phases.

Opportunity Try to see problems as opportunities rather than as survival mission.

Redundancy Difficult problems that are critical to the development activity should be solved in several different ways. The additional costs of redundancy is exceedingly compensated by the savings from preventing disasters.

Failure Fail if you are having trouble in succeeding. This is not intended to use as an excuse if you knew better, but if you do not know what to do risking failure may be the best way to success.

Quality Giving up quality as a means of control is not an option, quality must not be used as a control variable.

[1] The principles, as well as all other references in this thesis, are taken from the 2nd edition of the book which differs a lot, especially in terms of naming, to the original 1st edition published in 1999. As stated in its foreword, the 2nd edition is more a complete rewrite than a minor update.

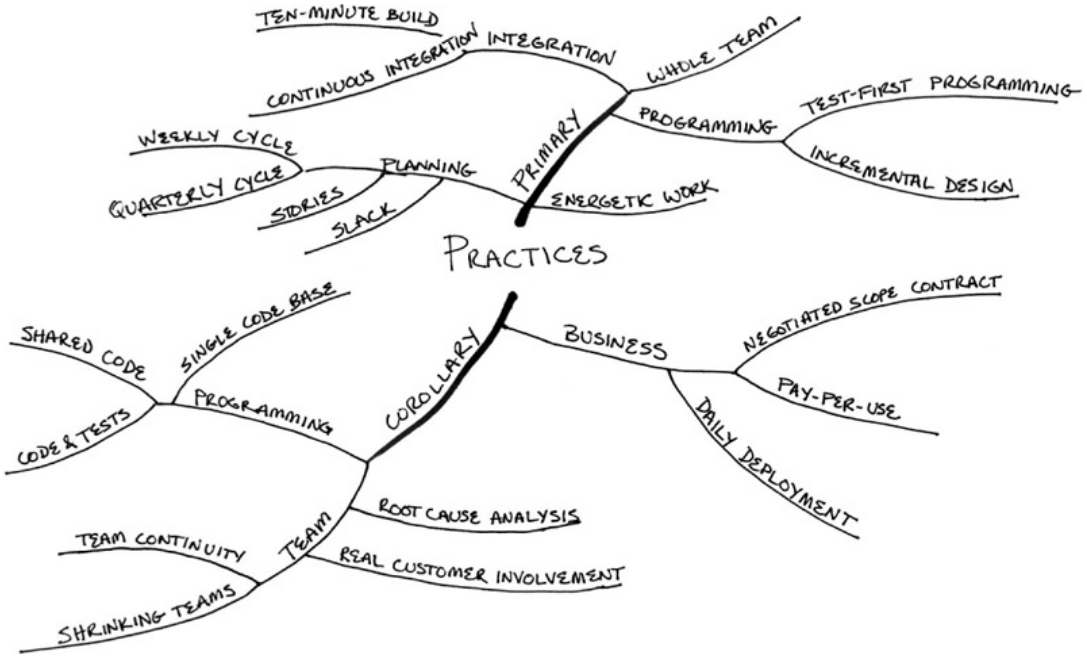

Figure 2.5: Summary of XP practices [30, Figure 3]

Baby Steps This principle states that taking many small steps towards a certain goal instead of making fewer big changes has a smaller overhead.

Accepted Responsibility States that it is not possible to assign responsibility, it rather has to be accepted.

2.6.3 Practices

In contrast to the values and principles, the practices of XP are defined actions and activities, they are depending on the situation. Applying the various practices is a choice, if a situation changes so does the practice that should be applied. Those practices are not the capstone of software development, they rather constitute a constant flow of improvement and are sometimes common ways of working together or improving products and working patterns. [30]

The nature of the practices is manifold, they include technical aspects that aim to improve quality of source code or test coverage, or team aspects like communication, trust, and collaboration. They also include business and management related techniques to enhance customer communication or process improvements. Figure 2.5 shows a summary of the common XP practices, some of them are discussed in detail in Section 4.4 which especially focuses on their application in a distributed environment.

2.7 Combination of Agile Methodologies

The methodologies presented in this chapter are often combined, and Scrum teams often apply practices from XP. „The Scrum framework is also flexible enough to embrace many other learning

loops. For example, although not specified by Scrum, technical practice feedback loops, such as pair programming (feedback in seconds) and test-driven development (feedback in minutes), are frequently used with Scrum development.“ [27, p. 46]

This comes at no surprise since there are various aspects that are shared across different agile methods. Shore and Warden [18, p. 26] for example cross-reference different practices across XP and Scrum and there are multiple practices that are explicitly defined by one but implied by the other practices.

There furthermore exist various form of hybrid process models where two or more different approaches are fused together. One example for this would be the so called *Scrumban* process, which is „a simplified version of Scrum, keeping the daily Scrum meeting and the Kanban board (hence the name), but eliminating the planning activities and velocity measurement.“ [38, p. 251]

In the Scrumban process, there are no time boxed Sprints but the work tasks are constantly flowing through a Kanban board. Its main focus is on the WiP and on the task flow instead of Backlog estimations. It is also argued that in a Scrumban process it is easier to include various XP practices than in a Scrum process. Generally Scrumban is suggested for teams that are working on simpler projects and face many interrupts during their work which. [38, p. 252]

> „Scrumban is not about using just a few elements of both Scrum and Kanban to create a software development process. Rather, it emphasizes applying kanban systems within a Scrum context, and layering the Kanban Method alongside Scrum as a vehicle for evolutionary change. Ultimately, it’s about aiding and amplifying the capabilities already inherent in Scrum, as well as providing new perspectives and capabilities.“ [39]

Another approach of a hybrid methodology is proposed by Nuevo, Piattini, and Pino [40], who combine the Rational Unified Process (RUP) and Scrum to a methodology they call *scRUmP*. That process is designed for globally distributed environments and its purpose is to „offer guidelines for the management and development of software in a distributed environment, using the advantages provided by the integration of agile methods such as Scrum and traditional methodologies such as the Rational Unified Process.“ [40, p. 66]

3 Distributed Software Development

3.1 Introduction

This thesis uses the definition of Ågerfalk et al. [10] for the term *development*, which states that development is „any software development lifecycle activity.“ [10, p. 48] Accordingly, development not merely includes just writing source code but rather also comprises various aspects of software development like planning and design activity, deployment or maintenance. Furthermore the term *activity* is broadly defined as „any individual or collective human action at any level of granularity that serves a particular purpose.“ [10, p. 48]

A software engineering team or project is accounted as distributed when team members are sited in different - geographically dispersed - locations. But it is not necessary that every subject is in a different place, a project can also be regarded as distributed when some of its sub-activities are not. [10, p. 48] This is the definition that is also chosen by O'leary and Cummings [41]: „Regardless of the units of measurement, geographically dispersed teamwork (by definition) requires that at least two members be separated by spatial distance. By defining geographically dispersed teams in this way, we allow for a continuum of dispersion from teams with one remote member to teams with no co-located members.“ [41, p. 14]

3.2 Taxonomy of Distributed Software Engineering

> „Since there are so many variations of the attributes associated with global software projects a large amount of new terms has been introduced. The diversity in sourcing jargon however has caused difficulties in determining which term to use in which situation, and thus causing further obstacles to searching and finding relevant research.“ [15, p. 105]

A distinction especially important for this thesis is the degree of distribution. This differentiation is quite complex since a lot of literature does not really apply a strict spatial measurement and do use different terminology for similar situations.

This problem is also stated by O'leary and Cummings [41] who argue that „Geographically dispersed teams are usually treated as an undifferentiated category, including everything from laboratory groups separated by temporary partitions to a team spread around the globe“ [41, p. 4] and furthermore concluded that „the majority of empirical research on geographically dispersed teams has defined dispersion loosely and usually in spatial terms. Even when the spatial dimension of dispersion has been defined explicitly, it has rarely been measured.“ [41, p. 9]

A rather coarse approach is presented by Woodward, Surdek, and Ganis [13, p. 8-12] and focuses a lot on the possibility to have synchronous communication and the amount of time a team is being co-located. It distinguishing four types of teams, ordered by their increasing level of distribution:

- Collocated
- Collocated Part-Time

- Distributed with Overlapping Work Hours
- Distributed with No Overlapping Work Hours

A recent study from Šmite et al. [15] attended this problem by analyzing 296 articles as well as conducting a survey with leading experts in the Global Software Development (GSD) field to ensure and strengthen their findings. They considered multiple factors including the legal entities between sites, whether teams are situated in different countries or within the same one, the geographical distance and the number of involved locations. Using those factors they created a glossary to characterize the different sourcing strategies [15, p. 122-123]:

Global software engineering Development of a software artifact across more than one location

Insourcing Leveraging company-internal human resources

Nearshoring Leveraging resources from a neighboring country

Offshore insourcing Leveraging company-internal resources situated in a different country

Offshore outsourcing Leveraging external third-party resources situated in a different country

Offshoring Leveraging resources from a different country

Onshore insourcing Leveraging company-internal resources situated in the same country

Onshore outsourcing Leveraging external third-party resources situated in the same country

Onshoring Leveraging resources from the same country

Outsourcing Leveraging external third-party resources

Sourcing Leveraging resources

Using this established terminology the following taxonomy provides relationships between those terms and creates a classification that can be used to identify branches that describe specific situations. It is segmented into five stages of distinction [15, p. 123-125]:

GSE Is the first section and determines whether there is any kind of external development at all.

Location The geographical location of the different sites can be either within the same country (onshore) or spanning over multiple countries (offshore).

Legal Entity Determines if the development is performed within the same company (insourcing) or in cooperation with other companies (outsourcing) for example by subcontracting.

Geographical Distance The definitions for geographical distance distinguish between offshoring and offshoring situations:

> **Onshore** Although if a team is distributed within the same country it still makes a difference whether two locations are within the same city, requiring a 20 minute drive to gather a face to face meeting or for example on the west coast and the east coast of the United States, which would require several hours of air traveling to cover that distance. Therefore the taxonomy distinguishes **close** and **distant** situations. In close situations there are no flights needed and it is therefore possible to have frequent face to face meetings if necessary. In distant situations on the contrary it is necessary to use air travel and therefore invest an substantial additional amount of time and money to meet personal.

Offshore The distance measurement in an offshore situation is different, the taxonomy differentiates between **near** (or *nearshoring*) situations on the one hand where air travel time is less than two hours and it is possible to have the arrival, meeting and departure within one day while still having time for a meeting of at least three hours. And on the other hand **far** (or *farshoring*) situations where the flying time is at least two hours such as in general makes an overnight stay necessary to conduct meetings.

Time Difference This last segment, similarly to the geographical distance section distinguishes between on- and offshore situations:

In onshore situations those constellations with a time difference of one hour or less are considered as **similar** while those with more than one hour are labeled **different**.

In offshore situations, a time difference of four hours or below is considered **small** based on the fact that in such constellations at least half of normal workday overlaps. a **large** time difference in an offshore setting is consequently defined as more four hours of time zone difference.

Utilizing this definitions, the final taxonomy is depicted in Figure 3.1. Some of the combinations are grayed out since they are not practicable. This created taxonomy is useful to exactly describe and compare situations but it is noteworthy that the authors point out that it does not include cultural differences and thus such factors may be considered in an additional step. Compared to other approaches this taxonomy provides a very detailed and structured classification and is very well suited for the goal and constraints of this thesis.

Following that taxonomy, this thesis will focus on teams that can either be classified as *Onshore - Insourcing - Geographically Close* or as *Offshore - Insourcing - Geographically Near*. This means that locations can be in different cities of the same country, or - if spanning multiple countries - such situations where the local time is not deviating significantly and the different locations are reachable within a reasonable amount of time.

3.2.1 Reasons to distribute Development

Although the reasons why a team ends up in a distributed setting are manifold and distributed working usually entails various challenges and problems, there are also several arguments supporting a distributed setting:

> „There are lots of reasons to run a project at multiple sites. Salary differential is only one of these reasons. The database people may be in Toronto and the telecom people in Denver. No matter the reason for considering multi-site development, it always comes down to a business decision: weighing whether the waste created by not sitting together is more than offset by other advantages.“ [30, p. 149]

The People

As argument on a global scale of distribution, Ebert [42, p. 17] states that many countries do not have enough personal resources to cover their demand of software engineers. He sees a global race going on between companies to get the best software developers. Outsourcing is one way to provide a company with the needed personal. Also van Solingen [43] argues in a similar but more general way, in his opinion a company which opens itself to distributed development can find more (well educated) people as employees. In his opinion this human resources argument is the main reason for using distributed development.

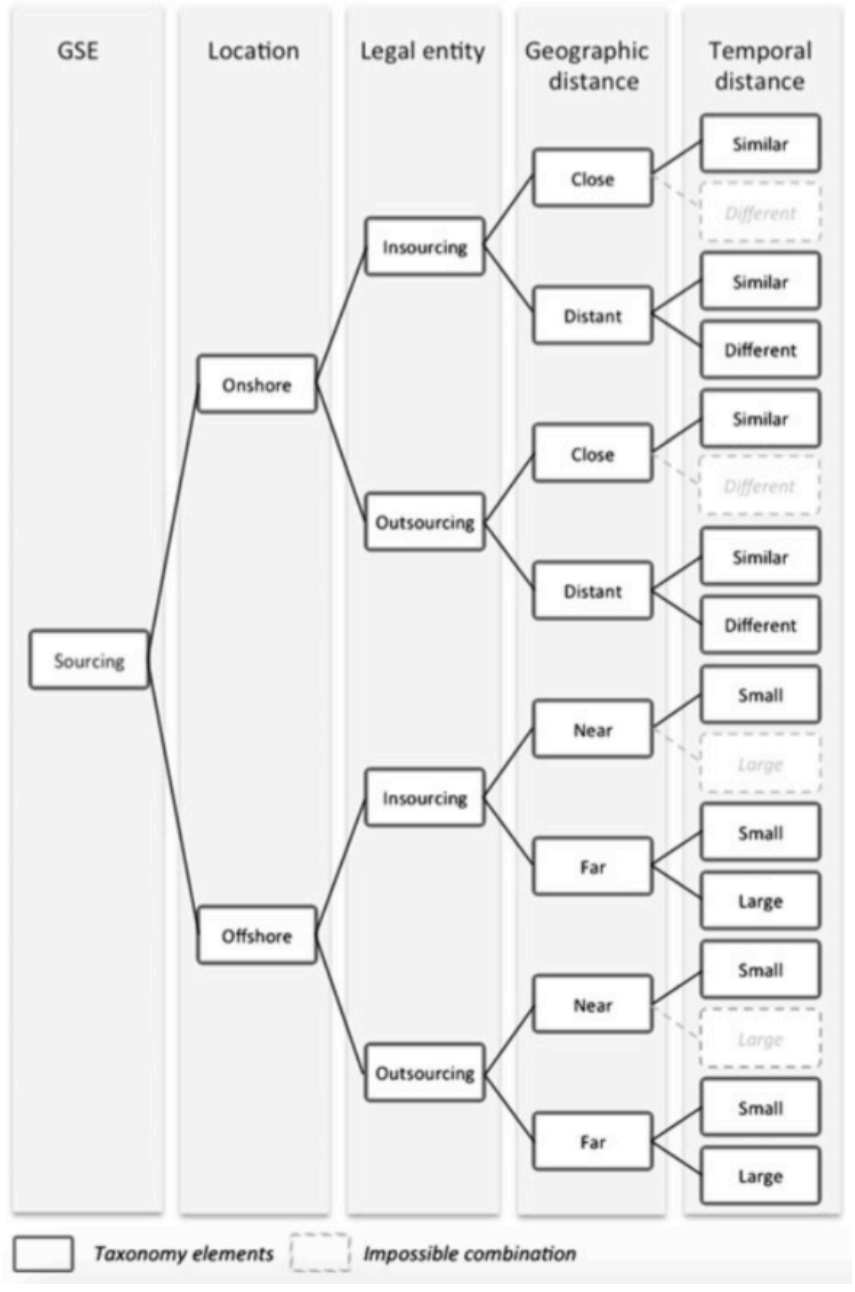

Figure 3.1: GSE taxonomy [15, p. 124]

Talent acquisition is but one part of this argument, in some points it is simply an argument of money and availability, and outsourcing of (at least a part of) a project may reduce expenses, as argued in [13, p. 6]: „With a well thought-out plan to best leverage the talent in multiple countries, it can be less expensive to develop a product. Working with distributed teams where the talent is available to do the work can sometimes reduce labor and business operations costs.“

Presence

Accessing new markets and also being near to customers, that is one very important point in most management strategies. For example, the IBM Scrum Community had members in their development teams allocated in more than 30 countries worldwide. [13, p. 67]

Showing presence in general is very important in a globalized world, and can also improve understanding of regional needs, not just in the matter of software development but also with provided services. [42, p. 16-17]

24 Hours Development

This argument is relevant in a distributed setting with no or little overlapping work hours. Where this difference in work time is normally regarded as a big challenge to face when working within a team, it also can be a strength. Also called the "Follow the Sun" model, one team can pass their work done to another team in a different timezone which is just starting their work time, so it is possible to work on a project 24 hours a day. [13, p. 6]

This can also in a smaller context be of use, like van Solingen [43] arguing that in some situations it can reduce coordination problems with single tasks: by passing an assignment on to those people working in another time zone, that task can be completed until the next morning and therefore reduce dependencies from teams that otherwise could delay the project progress. [43]

3.3 Dimensions of Distance

Regarding a team as distributed intuitively suggests that team members are not in the same spot but reside in different geographical locations. But this is just one dimension of distance. Additional to the mere geographical aspect there is also a cultural, temporal and a configurational dimension, which have different impacts on the challenges of distance portrayed in chapter 3.4. [44]

3.3.1 Geographical Distance

Geographical distance is measured directional and can be described as „the effort required for one actor to visit another at the latter's home site.“ [10] Therefore it is suggested to not use kilometers as indicator but rather the effort that is needed to relocate from one location to another. If for example two positions are well connected via air link they can be considered less distant than two locations that are initially closer but suffer from a bad transportation infrastructure. There exist several aspects that have an impact on that relocation effort, e.g. time and ease of travel, costs, or bureaucratic conditions like permits or visa. [10]

The Allen Curve

The Allen curve, named after its author Thomas Allen, is a graph (shown in Figure 3.2) that displays that communication between people drops exponentially with increasing physical distance.

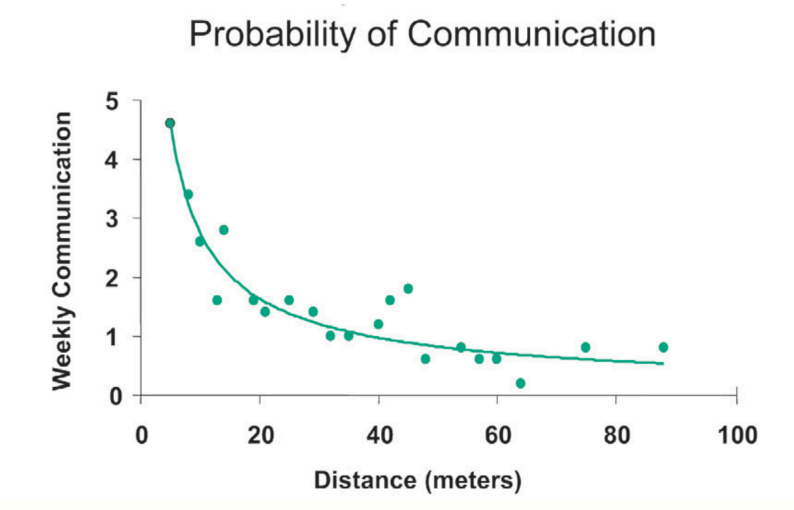

Figure 3.2: Probability of communication [45, p. 57]

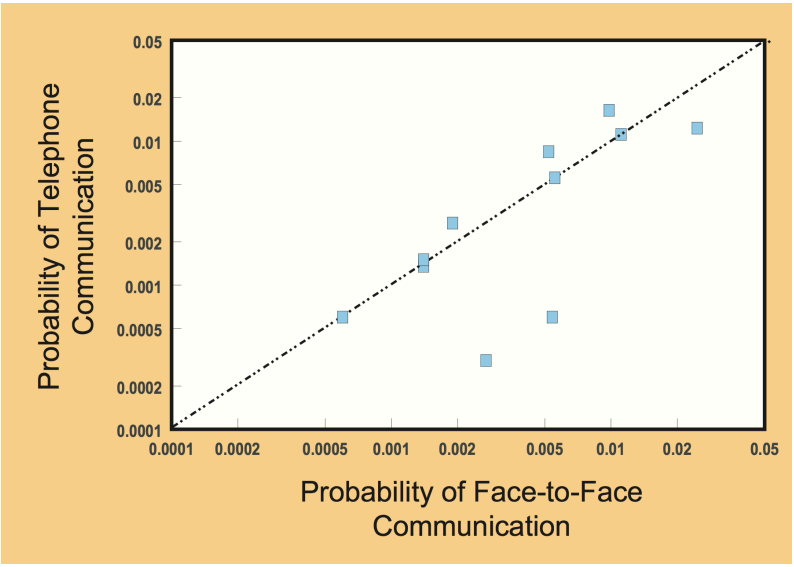

Figure 3.3: Correlation between face-to-face and telephone communication [45, p. 59]

„... the probability that people in a given organization will communicate with each other declines precipitously the farther away from each other they are situated and reaches an asymptotic level at about 50 meters.“ [45, p. 56]

Although this finding was already published in 1984 it apparently is still a valid concept, as Allen and Henn [45] show that their gathered data indicates a decline in the usage of all communication media with increasing distance. One reason for this is that different communication media (also including written communication like emails) correlate regarding their use. The more often two people see each other face-to-face, the more likely it is for them to also communicate through other mediums. This link is also shown in Figure 3.3 which depicts the correlation between face-to-face and telephone communication. [45, p. 58]

3.3.2 Cultural Distance

Cultural distance is generally a rather manifold dimension, Carmel and Agarwal describe it as „the degree of difference between the Center and the Foreign Entity.“ [9, p. 25].

The term *culture* is very complex and therefore is one of least understood dimension of distance. But it is argued that it is also a critical element of working in a distributed setting. [46]

When relating to distributed software development it is important to distinguish between the national culture, which is especially relevant when arguing in a GSD setting, and the organizational culture.

Carmel and Agarwal defines national culture as „an ethnic group's norms, values, and spoken language, often delineated by political boundaries of the nation-state“ [9, p. 25]. Organizational culture on the other hand are the shared beliefs and values within an organization, including for example the use of development methodologies or management practices.[9, p. 25] As a consequence, it is possible that two actors A and B are culturally closer although they are located in different countries, while on the other hand the cultural distance between A and an individual C - residing in the same country - may be far greater. [10]

This classification is also used by Dubé and Paré [47] who state that „When members of different organizations are united into a single team, two or more disparate socio-technical systems meet. For example, the culture of an organization will influence how team-related activities are valued. Faced with diverging evaluation and compensation systems, team members' motivation and behaviors may be affected“ [47, p. 19].

Furthermore they argue that the professional culture, especially present in cross-functional teams, can have a strong impact on a team: „When unfamiliar professional cultures are united in a team however, members may lack the shared meaning and language, patterns and routines to agree on a shared purpose, goals, and priorities. They may even have problems dividing tasks, coordinating work, handling conflict, and formulating rules“ [47, p. 20]. This shows that, while the national culture may not be of a big concern in teams distributed within the same country, aspects like the professional and company culture may very well be of importance.

The complexity of this dimension is also reflected in the different opinions of researchers, while a lot of authors agree on the importance and impact of different culture on the process of distributed software development (e.g. [9] [46] [10]), van Solingen argues that „culture in my experience is the best excuse for your own failure“ [43, min. 42]. In his opinion national culture is dominated by the company culture and again this shared company culture is dominated by a professional culture. He states that it is in fact necessary to learn and understand cultural differences within a team, but it is also not that big of a concern if the responsible leaders train their team members accordingly.

3.3.3 Temporal Distance

There are two major factors impacting temporal distance, the greater one being differences in time zones. The second aspect is the schedule of normal work hours of the different team members or locations. Scheduling work patterns therefore is critical since it is can either decrease that distance but can also make it worse. For example, if there is a one hour gap between two working sites caused by a time zone difference, different routines may increase that distance significantly. [10]

One of the biggest issues with a temporal distance is that it impedes synchronous communication between individuals and therefore having a big negative impact on communication. Small problems that could be easily resolved by direct communication may therefore turn into bigger

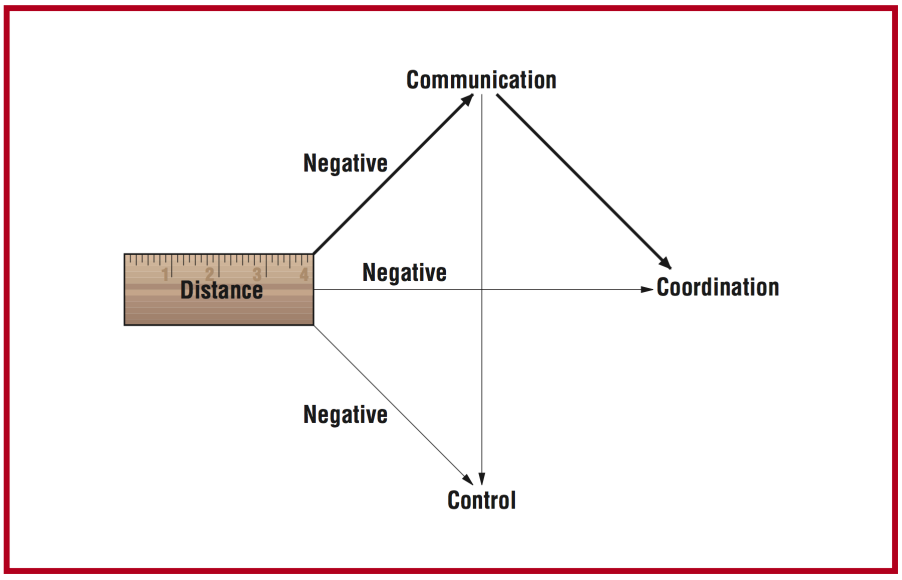

Figure 3.4: Impacts of distance [9, p. 24]

obstacles - asynchronous communication therefore frequently complicates and slows down problem resolution. [9, p. 27]

Due to the constraints of this thesis this aspect of distance will be not relevant and therefore regarded as no obstacle.

3.3.4 Configurational Dimension

Another very important aspect in a distributed setting is the configuration which is „the arrangement of members across sites independent of the spatial and temporal distances among them.“ [41, p. 16]

In difference to the previous mentioned dimensions, this dimension is dealing with the location of team members and not the distance between them. A team of eight people could be split into 21 different configurations like for example 4-2-2 or 3-3-2. This different configurations also include situations with a concentrated core team being on one site and isolated members being on an other site. Such isolation decreases the awareness from other team members regarding their actions. Furthermore, a larger number of sites increases the complexity for coordination and increases general conflict potential. [41, p. 17]

3.4 Challenges of Distance

There are various aspects of organization within software development and a common approach is that for an organization in order to function it is necessary to have *control* and *coordination*, both factors driven by *communication*. Various authors have identified those three aspects and focus on one or multiple of these aspects, e.g. [8] [48] [9] [10] [11] [49]. Figure 3.4 shows how distance impedes coordination and control directly (bold arrows), as well as indirectly through its negative effect on communication.

3.4.1 Coordination

From a general perspective, a definition of coordination is given by Malone and Crowston stating that: „Coordination is managing dependencies between activities.“ [50, p. 90]

In the context of software development, Wiredu [51] names *people, processes, information and technology* as the four core dimensions of distributed development, and states that success depends on the adequate coordination of those dimensions and the interactions between them. [51, p. 41] He furthermore defines coordination in distributed software engineering as „managing interdependencies, uncertainties and equivocalities, conflicts, technology representations, and their interrelations.“ [51, p. 38-39]

Viewed from a management perspective, coordination is the process of integrating tasks with organizational units, to enable those entities to contribute value to the all-up objective. This process of integration commonly requires steady and intense communication. [9, p. 23]

Traditional, co-located software development teams have built up ways of coordinating work, team members have often a shared view on the ongoing process. Frequent interactions, formal as well as informal, it is often clear who possesses certain expertise and where responsibilities are located. The flow of information is free and strengthened by many informal interactions, be it joint meals or random encounters and chats on the hallway. Based on prior collaborations there is a built up relationship between individuals which helps preventing misunderstandings and supports resolving arising problems. In a distributed environment, many of those mechanisms of coordination are disrupted or completely absent. Less (effective) communication, lack of awareness and incompatibilities (e.g. processes, tools or work habits) are very common aspects of distributed teams that interrupt coordination mechanisms. [52]

Ovaska, Rossi, and Marttiin [53] state that especially in a multi-site development environment coordination is vital to success but hard to achieve. It is not enough to coordinate activities but it is necessary to also coordinate the independencies between different activities. They propose that the participants in their studies coordinated development work by using interfaces meaning that the software architecture is used for coordination and generally they emphasize the importance of formal as well as informal communication to maintain a constant awareness and common understanding of a systems architecture and the current activity on other sites.

3.4.2 Control

Controls are generally defined as mechanisms to encourage individuals to act in ways that support objectives of organizations. They are forms of endeavors a controlling entity influences the behavior of a controlled individual, where both - the controller as well as the controlled - may either be single individuals or groups. Furthermore, control is grouped into two basic categories: formal and informal control. [54, p. 28]

Formal controls are clearly defined guidelines, in a project management environment examples are deadlines for deliverable, acceptable error rates, schedules or budgets. These formal specifications can also be used to measure the performance of individuals and the results can be used to reward individuals for meeting certain goals amplifying the level of control. [55, p. 140]

Informal control can be split into clan control which operates through dynamics of members within a team. The impact depends on the level of how strong individuals identify themselves with the team and share equal values and commitment. The second aspect of informal control is self-control which is generally based on the idea of intrinsic motivation. Using objectives that apply to specific individuals, allowing them to operate independent from each other and therefore rewarding them based on their individual performance. [55, p. 141]

When looking at the control aspect in the context of agile development teams, it can be seen that the self-organization (as described in Section 2.3.1) aspect taps into intrinsic motivation of team members. This means that within a self-organized team the commitment of individuals towards the project goal can rise due to individual motivations and allegiance towards a mutual goal. [1, p. 216]

3.4.3 Communication

Communication - serving as the coupling factor between coordination and control - is the exchange of information between a sender and a receiver with the goal of reaching a mutual understanding. [9, p. 23]

This can also be seen in the definition of Allen and Henn [45] who distinguish three types of communication, each serving a specific purpose.

Communication for coordination exists nearly everywhere since there has to be some form of communication to direct coordination and to exercise control. [45, p. 28]

Communication for information is responsible for transferring and transforming existing knowledge. The importance of these "keeping up-to-date"-factors is proportional to the pace of change within a given field.[45, p. 28]

Communication for inspiration differs from the previous type in the way that it actively creates knowledge and is very important in fields that require creativity for solving problems. This type of communication is often happening impromptu and between individuals working in different areas or projects and therefore often concedes new and uncommon mixtures of ideas. But that means that it also has to cross boundaries within an organizational structure which is often an obstacle. Due to these uncertain factors it is hard to predict and therefore the type that is most difficult to manage. [45, p. 28]

Ågerfalk et al. [10] put the first three dimensions of distance in relation to those challenges and created a matrix showing the impact on each other, shown as a summary in Figure 3.5.

Conway's Law

> „The basic thesis of this article is that organizations which design systems [...] are constrained to produce designs which are copies of the communication structures of these organizations.“ [56]

This statement from an article authored by Marvin Conway in 1968 became known as Conway's law and is one of the first recognitions that communication and coordination patterns of a project team have an impact on the resulting product. While sounding very logical it should be noted that although called "law" it is more of a hypothesis. There is some evidence supporting Conway's law, like a study from MacCormack, Baldwin, and Rusnak [57] stating that they found „strong evidence to support the mirroring hypothesis. In all of the pairs we examine, the product developed by the loosely-coupled organization is significantly more modular than the product from the tightly-coupled organization.“ [57, p. 2] But there are also those who doubt its general validity, for example van Solingen argues that it's logical that teams tend to organize the architecture of a system based on the teams situation and distribution, which he says is a bad thing to do, because

Process	*Dimension*		
	Temporal Distance	**Geographical Distance**	**Socio-Cultural Distance**
Communication	Reduced opportunities for synchronous communication, introducing delayed feedback. Improved record of communications.	Potential for closer proximity to market, and utilisation of remote skilled workforces. Increased cost and logistics of holding face to face meetings	Potential for stimulating innovation and sharing best practice, but also for misunderstandings.
Coordination	With appropriate division of work, coordination needs can be minimised. However, coordination costs typically increase with distance.	Increase in size and skills of labour pool can offer more flexible coordination planning. Reduced informal contact can lead to reduced trust and a lack of critical task awareness.	Potential for learning and access to richer skill set. Inconsistency in work practices can impinge on effective coordination, as can reduced cooperation through misunderstandings.
Control	Time zone effectiveness can be utilised for gaining efficient 24x7 working. Management of project artefacts may be subject to delays.	Difficult to convey vision and strategy. Communication channels often leave an audit trail, but can be threatened at key times.	Perceived threat from training low-cost 'rivals'. Different perceptions of authority/hierarchy can undermine morale. Managers must adapt to local regulations.

Figure 3.5: Framework of issues in distributed development [10]

„architecture is not there to serve the teams, architecture is there to serve the end user.“ [43, min. 25]

One repeatedly mentioned issue in the context of Conway's law in a distributed setting is, that teams tend to split up the software they build into separate parts to reduce communication and coordination needs while developing those components. The problems then arise in the integration phase when the built components need to be put together, one reason is for example incomplete specifications leading developers to make personal different assumptions about other components. [58, p. 88] This is also mentioned by Ovaska, Rossi, and Marttiin [53] who study the usage of architecture as a coordination tool and reported that in the multi-site development „there were several coordination problems in the implementation of the final system.“ [53, p. 243]

3.5 Communication in Distributed Teams

3.5.1 The Importance of Communication

It can be seen that communication is one of the most essential aspects of agile teams, *interaction* between team members is one of the first things that comes up in the agile manifesto (as described in Section 2.2). Furthermore the agile values (discussed in Section 2.2.1) are even more specific in this regard. They state that developers and business people have to collaborate on a daily basis (4th principle), that the most effective way of building a product is through face-to-face communication (6th principle), that teams have to be self organized (11th principle) - and as discussed in Section 2.3.1 self-organized teams strongly rely on frequent informal communication

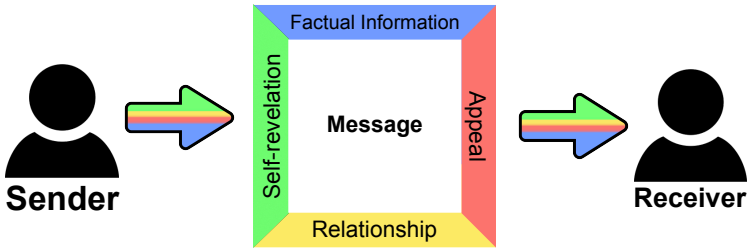

Figure 3.6: Four sides of a message, adapted graphic from [60, p. 33]

- and also on feedback (12th principle) to keep improving. In summary four of the 12 principles have a direct reference to communication aspects.

3.5.2 Communication Theory

Before investigating the different ways of distributed communication with their potential benefits and drawbacks, it is important to discuss some basic communication theory. „One cannot not communicate“ is the first axiom of communication from Watzlawick, Helmick-Beavin, and Jackson [59]. It states it states that every behavior is simultaneously communication, and as it is not possible to not act it is also not possible to not communicate. The second axiom says that every communication has a content as well as a relationship aspect, and that the latter classifies the former. This work, together with additional approaches of other psychologist led Schulz von Thun [60] to the creation of the Four-Sides model of communication. [60, p. 13-14]

Four-Sides Model

The Four-Sides model from Schulz von Thun is a very famous model of interpersonal communication and consists of three basic elements:

The *sender* is the entity that wants to communicate something, it encodes its concern in perceptible signals, called the *message*. On the other end is the *receiver* of that message who resides with the task to decode the message. The message itself is a very complex entity, it contains not just one directive but multiple different ones. The message is divided into four different aspects, that are all open to interpretation and may lead to differences between the sender and the receiver. To better explain the different aspects of the message, Schulz von Thun provides the example of two people within a car, where person A is the driver, and person B is the co-driver. B then says to A: *"The traffic light in front of us is green!"* [60, p. 27-33]

The factual information aspect contains facts and data and is normally rather clear. In the driving example given above this aspect contains information about the traffic light - it is showing green. [60, p. 28]

The self-revelation aspect says that in every message there is not just the raw factual information but it also contains information about the sender. This self-revelation consists of two sides, first the information that is willingly intended self-expression and as a second component it also contains unintended self-revealing. [60, p. 29]

The relationship aspect covers the relationship between the sender and the receiver and is often contained in things like phrasing, intonation and other nonverbal signals. This aspect is critical for the receiver because it determines on how the receiver feels treated. Sending a message always also expresses a certain relationship. The difference to the previous mentioned aspect is that in the relationship aspect the receiver is affected personal, while in the self-revelation aspect the receiver is rather parsing information about the sender. Furthermore this facet also contains two messages, first it tells something about how the sender experiences the receiver, what he thinks about him. Secondly it contains how the sender views its relationship with the receiver.

In the driving example the relationship aspect could express that B does not really trust the driver to drive safely without help. [60, p. 30-31]

The appeal aims to make the receiver feel, think or do certain things. This attempt to reach a certain goal can either be open or hidden, in the first case Schulz von Thun [60] calls it *advice* while in the latter case it is called *manipulation*. In the case of manipulation the sender does not refrain from using the other three sides intentionally to amplify the appeal effect. [60, p. 32-33]

3.5.3 Modalities of Communication

Communication has more to it than just the written or spoken words, a very important aspect is the nonverbal component of a message. Schulz von Thun [60] names the voice itself, intonation, pronunciation, facial expression and gestures as example for nonverbal components of communication. Those channels may give additional information on the original intentions of a message. [60, p. 37]

Cockburn [31] gives the example scenario of a discussion at a whiteboard and lists the following communication mechanisms that might be in play and have an influence on the participants:

Physical proximity is the mere fact that people are separated by a few meters from each other. This proximity allows to detect minimal visual cues like muscle tension or eye movement.

Three dimensionality means that people experience a scene three dimensional: „The parallax shift of the visual image is lost when the same people talk over a video link, even if they are similarly close to the camera and screen.“ [31]

Smell may be an unimportant sense to some people but can be very important to many people especially since it often is a subconscious perception.

Touch in terms of physical contact can have have an impact on communication and is part of a comprehensive manipulation of personal space and closeness

Sound is on the one hand spoken language where a speaker may for example use certain formulations or metaphors. Apart from the wording itself it are also aspects like the pitch, volume, pace, emphasis, intonation or pronunciation that can determine the meaning.

Visuals is the aspect of individuals seeing each other. Gestures and body language is a very important aspect of communication. But not only the image of persons is of importance, also the way people interact with their environment should be considered. The way a person draws something on a whiteboard provides information for others: while drawing obvious things swifter, slowing down or even pausing while sketching can increase awareness for important parts.

Crossmodality Timing refers to the timed correlation of the previous mentioned modalities. The argument is that the simultaneous presence of for example seeing someone sketching while at the same time hearing that person speak can enhance awareness and correct understanding.

Low latency is aiming at the time a message takes from the sender to the receiver. A low latency allows for real-time response so that it is possible to get immediate feedback within a conversation. This responses include question and answer in real-time to clear out misunderstandings right away and also the possibility to interrupt and ask for clarification or express own thoughts immediately.

3.5.4 Media Richness Theory

Media Richness Theory, first introduced in 1984 by Daft and Lengel [61], tried to evaluate different types of communication channels regarding „their ability to enable users to communicate and change understanding – their 'richness'." [62, p. 1] The basic argument was that depending on the uncertainty and the level of ambiguousness, certain media types are better suited than others. Cockburn [31] uses the terms "temperature" as synonym for richness and uses a scale from *cold* to *hot*: „Warmer indicates that more emotional and informational richness gets conveyed. E-mail is cooler than audio or videotape, and two people communicating face to face is the hottest channel." [31]

Dennis and Valacich reviewed the basic idea of media richness and surveyed different media types regarding the following dimension:

Immediacy of feedback is the degree to which a certain medium allows to get immediate feedback, and more generally it is the level of latency within a communication. [62, p. 2]

Symbol variety is the amount of different possibilities how certain information can be expressed and communicated. This aspect also includes nonverbal aspects that are mentioned in 3.5.3. [62, p. 2]

Parallelism is referencing to the amount of conversations that can exist simultaneous. A traditional medium like a telephone can only support one conversation effectively at a time. [62, p. 2]

Rehearsability is the ability to fine tune a message before sending it.[62, p. 2-3]

Reprocessability describes the extent to which it is possible repeat and review a message. [62, p. 3]

Figure 3.7 from Dennis and Valacich [62] examines selected media types regarding their capabilities in the listed richness dimensions. It is noticeable that one medium posses different levels of capabilities, depending on its usage and configuration.

It can be seen in this table that no medium has the best score in all dimension which means that the answer on the question which medium is the "best" is always depending on the situation and which of the dimensions is of most importance. Dennis and Valacich [62] conclude that in contrast to the original idea of Daft and Lengel [61] media cannot be simply ranked in order of richness and that such a classification is not practical. Furthermore, they argue that „choosing one single medium for any task may prove less effective than choosing a medium or set of media which the group uses at different times in performing the task, depending on the current communication process." [62, p. 9]

	Feedback	Symbol Variety	Parallelism	Rehearsability	Reprocessability
Face-to-face	high	low-high	low	low	low
Video conference	medium-high	low-high	low	low	low
Telephone	medium	low	low	low	low
Written mail	low	low-medium	high	high	high
Voice mail	low	low	low	low-medium	high
Electronic mail	low-medium	low-high	medium	high	high
Electronic phone ("chat")	medium	low-medium	medium	low-medium	low-medium
Asynchronous groupware	low	low-high	high	high	high
Synchronous groupware	low-medium	low-high	high	medium-high	high

Figure 3.7: Capabilities of selected media in Richness dimensions [62, p. 3]

3.5.5 Remote Communication

When distributed, a team faces several communication obstacles but as discussed in the introduction of this Section, communication is a vital part of agile teams. Within recent years the amount and quality of possibilities for communicate over distance have rapidly increased, and it is absolutely necessary for distributed team to find and use suitable ways of communication. To be successful requires the utilization of the right tools for the specific situation and team. Thissen et al. analyzed what tools different teams used and summarized different types with examples, shown in Figure 3.8. [12, p. 29]

Video Conference

Though it may seem that seeing each other on a video link may be nearly the same as a real face to face conversation, it still has not the same effect. First, it is the simple removal of several communication modalities listed in Section 3.5.3, namely: Physical proximity, touch, smell and also the perception of three-dimensionality. [31]

Further challenges regarding video conferencing are mentioned by Woodward, Surdek, and Ganis [13, p. 104] :

> „This approach needs added hardware as each location joining the video conference will need a webcam. To be able to conference multiple video streams at the same time, the team may also need extra software. There may be software and bandwidth limitations as well, which could cause problems with the video feed. The other challenge is when there are multiple participants in one of the video streams— where should the focus of the webcam be? When only focusing on the current speaker, the remote participants will lose any nonverbal reactions of other participants at that location. When focusing on a larger group at a location, it may be difficult to see everyone.“

Audio Conference

Removing visuals simultaneously removes a lot of cross-modality timing. Furthermore, the removal of visibility within a conversation eliminates a lot of non verbal communication aspects.[31] Also actions happening in one place cannot be noticed by remote participants being it gestures or actions like handing out papers or passing objects. [27, p. 30]

A general problem with audio transmission in conferences is the aspect of properly hearing and being heard. A conference where multiple people sit gathered around a table with a speaker and microphone in the center it can be difficult to clearly understand speakers. Although sounding like

a minor aspect, audibility is quite a relevant factor and a lot of meetings turn out to be ineffective due to poor perceptibility. Another common obstacle, especially for fresh formed teams is to identify who is currently speaking, due to the absence of a lot of communication aspects individuals have to put a lot of focus on the sheer voice of others. This problem can be reduced by either everyone identifying themselves before speaking or using technology that helps with identification of speakers. [27, p. 30]

Cohn [1] also mentions this issue of identifying the speaker and suggests to speak one's name always when starting to talk but he also states that with such a method „calls seem to take longer with all the "This is Mike..." prefacing that occurs. Also, during a rapid or heated discussion, it is very hard to remember to start each statement that way.“ [1, p. 378] Therefore he reports another approach he learned:

> „One team I worked with found an interesting nuance that improved upon this speak-your-name-before-you-speak approach. Team members called the technique "low fidelity videoconferencing" and often preferred it to regular videoconferencing because of the inevitable problems and delays with that equipment. Low-fidelity videoconferencing involved one person in each city who had a good ear for different voices holding up a photo of whoever was speaking at the remote location.When Sonali starts talking, someone holds up her photo. When she finishes and Manish starts talking, his picture is held up instead. Photos of each person had been taken in advance and were taped to rulers, making it easy to quickly hold up the right picture.“ [1, p. 378]

Side conversations are a fundamental nuisance in remote communication situations, limiting and preventing them is therefore a very important aspect. Apart from the previous mentioned problems, side conversations are further distracting people both the co-located people talking to each other and missing out information as well as remote participants who get distracted. A responsible organizer has to watch out for such occurrences and limit side conversations or postpone items that are unrelated and to make sure that everyone on the team feels included and that there is no location dependent grouping that excludes individuals from other sites. Getting mutual agreement is also different in an audio based conversation, asking if everyone has understood a certain aspect or is supporting a decision will often result in a chorus of "yes" answers, maybe drowning a single "no" answer. Therefore it is even more important to pay attention to effective phrasing to avoid missing out single responses. [27, p. 33]

Text based Communication

By removing voice from a communication „you lose vocal inflection, the ability to pause for effect, to check for interruptions, to speed up or slow down to make a point, to raise your tone or volume to indicate surprise, boredom, or the obviousness of the transmitted idea“ [31, Chapter 2]

Relying strongly on text based communication also may have an impact on leadership as suggested by Tyran, Tyran, and Shepherd [63]: In co-located teams where face to face meetings are the standard way of interaction, vocal and assertive team members tend to dominate a group whereas virtual teams are „more likely to have emergent leaders who are skilled at facilitating and motivating through the written word rather than those who command the leadership role through a dominating vocal or physical presence“ [63, p. 189]

Tool	Examples	Uses and Advantages	Bandwidth Needs, Immediacy	Sensory Modes
Instant Messaging and Chat	• Yahoo Messenger • MSN Messenger • AOL Instant Messenger • Internet Relay Chat	• Instant interaction • Less intrusive than a phone call • View who is available • Low cost • Low setup effort	• Low bandwidth, can use dial-up • Immediate • Synchronous or asynchronous	• Visual • Text and limited graphics
Groupware / Shared Services	• Lotus Notes • Microsoft Exchange • Novell Groupwise	• Calendars • Contact Lists • Arrange meetings • Cost and setup effort vary	• High bandwidth is preferable • Low bandwidth may be enough • Asynchronous	• Visual
Remote Access and Control	• NetMeeting • WebEx • Remote Desktop • pcAnywhere	• User controls a PC without being onsite • Cost varies • Setup varies	• High bandwidth • Immediate • Synchronous	• Visual • Audio • Tactile
Web Conferencing	• NetMeeting • WebEx • Citrix GoToMeeting	• Live audio • Dynamic video • Whiteboard • Application sharing • Moderate cost and setup effort	• High bandwidth • Immediate • Synchronous	• Visual • Unlimited graphics • Optional audio
File Transfer	• File Transfer Protocol (FTP) • Collaborative Websites • Intranets	• Share files of any type • Cost varies • Moderate setup effort	• Low bandwidth for small files • High bandwidth for large files • Asynchronous	• Varies with file content
Email	Numerous vendors and free applications	• Send messages or files • Cost and setup effort vary	• Low bandwidth, can use dial-up • Asynchronous	• Visual • Audio in attached files
Telephone	• "Plain Old Telephone Service" (POTS) Voice Over Internet Protocol (VOIP)	• Direct calls • Conference calls • Cost varies • Low setup effort	• VOIP requires high bandwidth • Immediate • Synchronous • Asynchronous for voice mail	• Audio

Figure 3.8: Tools for distance communication and information exchange [12, p. 30]

4 Agility in a Distributed Environment

> „To encourage software development as a craft and a business through the next fifty years, I hope that programmers everywhere will accept the challenge of producing much more valuable software. I believe the expanding market will more than make up for losses in any one location because of increased efficiency and multi-site development.“ [30]

4.1 Agility in a Distributed Setting

While, as discussed in Section 2.3.2, co-location is important in agile teams with the increased numbers of distributed teams and advances in technology, a lot of agile literature states that, while being not an ideal situation, distributed teams can also benefit from using agile methods.

> „The values of XP are just as suited to multi-site development as they are to teams that sit together. Embrace feedback more tightly because of the natural isolation created by distance. Nurture communication more because of the unavailability of face-to-face, full-spectrum interaction. You'll have to work harder to achieve simplicity because you won't have as many chances for serendipitous discovery of excess complexity. Courage is just as important as it is in any other setting. Respecting everyone on a distributed team is even more important because of differences in culture and lifestyle.“ [30, p. 149]

But Beck and Andres [30] also states that that some practices will have to be adapted for distributed teams. Some aspects have to be increased in frequency or intensity to make up for the distance between the team members.

As already mentioned in Section 3.4, Ambler [34] did a survey on agile development in 2008 where he differenced between *co-located* teams, *near-located* teams and *far-located* teams. The results showed that while the success rate of agile teams that are co-located is at 83%, the success rate of not co-located teams is still at 72%. Another substantial drop in success can be seen in the far-away teams which just reported a success rate of 60%.

4.1.1 Impact of the Team Size

Traditionally, agile methods encourage to have small, cross functional teams. This aspect is also an important aspect of distributed teams, smaller teams are better at coordinating themselves without the need of formal coordination mechanisms [64]:

> „Compared to members of larger teams, we found that members of smaller teams participated more actively on the team, were more aware of the goals of the team, were better acquainted with other team members' personalities, work roles and willingness to communicate and reported higher levels of rapport.“ [64]

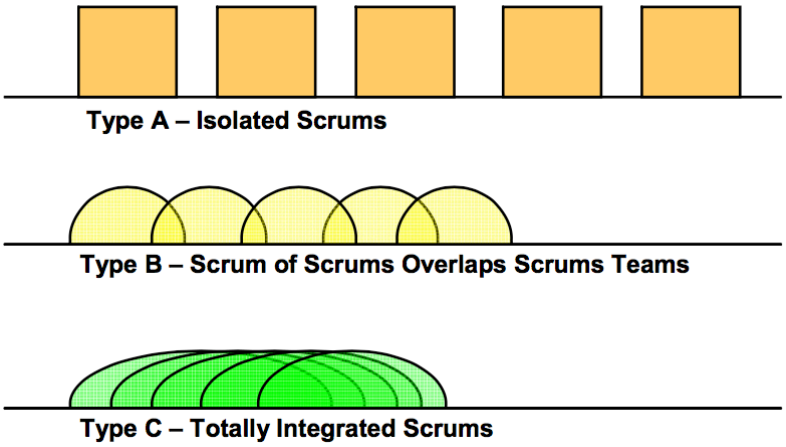

Figure 4.1: Types of distributed team organization [65]

4.2 Reasons to use Agile Methods

> „A common misconception is that Scrum is not a good fit for a geographically distributed team. Scrum's preference for face-to-face communication, the argument goes, makes it a poor choice for distributed teams. Fortunately, this argument is false.“ [1, p. 355]

Van Solingen [43, min. 6] argues that the main problems with distributed development are issues in coordination, control and communication, as discussed in Section 3.4. He further brings forward that for example Scrum, or agile methods in general are used to fix those problems because agile methods are especially strong in arranging those three dimensions. So Agility is used to solve the problems of distribution because it is firmly building on informal communication and informal ways of coordination and control. He furthermore argues: „I see lots of advantages with using Scrum in a distributed setting because Scrum is exactly solving those points which are weak in a distributed setting.“ [43, min. 7]

4.3 Distributed Scrum

4.3.1 Distributed Organization

Depending on the team situation, distribution can be handled in different ways, Sutherland et al. [65] list three possible constellations, depicted in Figure 4.1. Starting with a completely isolated model the overlap and interaction between teams increases with each model.

Type A: Isolated Scrums In this model, the team in each location is independent and teams are working independent from each other without cross location collaboration. This approach is basically trying to solve the problem of distribution by avoiding it - creating teams that are co-located to apply Scrum in a way it is originally intended. But separating the local teams as much as possible simultaneously also dismisses the benefits of having people in different

locations. If the expertise of one person is needed in another location it is still necessary to find a compromise and a way of distributed collaboration. [13, p. 12]

Type B: Distributed Scrum of Scrums This second model also focuses on creating separate teams on each location but with the difference that those teams use a regular Scrum of Scrums for coordination between the different locations. This would be applicable for a situation where different teams integrate create deliverables that are then integrated into an overall product. The Scrum of Scrums meeting should be held daily (in contrast to the recommendation mentioned in Section). Another difference is, that there is a fourth question added as a is proposed as addition to account for the distribution: „What blockers might you be throwing into another team's way?“ [13, p. 12-13]

Type C: Totally Integrated Scrums In this third model a Scrum team has members from multiple locations. Such a setting also utilizes a Scrum of Scrums meeting to coordinate between different teams and in generally needs a high amount of remote communication and collaboration. [13, p. 13-14] Furthermore it is strongly relying on the Daily Scrum meetings which help reducing cultural barriers and discrepancies in work styles. So, while seemingly creating coordination and communication burdens, Sutherland et al. [65] state that: „The virtual nature of this approach provides location transparency, which creates performance characteristics similar to a small co-located team.“

Choosing the right Organization

Selecting the right organization for a new project is a challenging task and always depending on the situation. Woodward, Surdek, and Ganis [13] recommend that „Individual Scrum Teams should aim to have the lowest distribution level possible.“ [13, p. 51] They argue that whenever possible it should be strived to create co-located teams who work on independent features.

On the other hand, it is mentioned that IBM uses the approach of totally integrated scrum teams successfully in a globally distributed setting. [13, p. 13-14] Also van Solingen [43] favors this variant, he argues that this is the best practice when multiple teams are working on the same system. He states that this way „you bring the distance within the team. And when you put the distance in the team people will work together. And they are trying to solve problems because let's say they are dealing with the total problem.“ [43, min. 28] But he also points out that in situations where it is possible to co-locate team members that should be the way to go for.

Also Sutherland et al. [65] point to similar direction, they state that instead of creating a complete isolation between teams it is better to at least have some overlap between the teams as in a type B organization: „This makes teams feel more co-equal and encourages communication, cooperation, and cross- fertilization.“ [65]

Team size is especially in a distributed setting a very important factor, and it is strongly endorsed to keep the amount of team members below ten members, teams greater than nine people should contemplate splitting into smaller teams. Smaller teams in general reduce the amount of communication lines needed to keep everybody updated and integrated. [13, p. 51]

Proxies

In distributed teams there are sometimes individuals or organizations that partially take the role of somebody else. This may sometimes be unavoidable but generally the use of proxy roles should be avoided as argued by Smite, Moe, and Gerfalk [8]: „This has in most cases we have discussed turned out to be a bad practice; an anti-pattern which we strongly discourage. It leads to reduced

empowerment of the offshored resources and too much information being lost in the communication.“[8, p. 308]

4.3.2 Starting a Distributed Scrum Project

The very first step when starting a new Scrum project is to set up the team and artifacts. The way the Scrum team and the artifacts are set up, influences all further processes.

Backlogs

The Product Backlog is an essential artifact in Scrum and as already mentioned the basic rule is that there is just one single Product Backlog. But having multiple teams across different locations sometimes leads to a modified administration.

Single Product Backlog If the distribution and team size allows it, the best way would be to have all members work together as one single Scrum team having one single Product Backlog as well as one single Sprint Backlog. Since the Sprint Planning meeting requires everyone on the team being present this can become challenging depending on the level of distribution. If there are multiple Scrum teams working on the same project, each team will have its own Sprint backlog but it is still strongly recommended to have one single Product Backlog if possible. [13, p. 52-53]

Single Sectioned Product Backlog Another approach in a situation with two or more teams would be the Product Owner dividing the Product Backlog into multiple sections that outline which work items are assigned to each team. Within this section the items are then worked on by the different teams according to their priority. [13, p. 53-54]

Separated Product Backlogs This approach is working for products that have multiple separated components which are appointed to different teams. Here each team has an own Product Backlog assigned which are derived from one overall Product Backlog picturing the overall product. This procedure allows more freedom and independence within the teams but also increases risks when it comes to the integration of the different pieces. The interdependencies have to be planned ahead and be thoroughly discussed in the Scrum of Scrums meeting. [13, p. 54-55]

Independent from the structure of the Backlogs, in a distributed environment it is inevitable to use a digital tool for managing and maintaining the Backlogs. In Practice there are multiple tools available, on the one hand there are solutions specific for managing Backlogs like Scrumworks [1] or Agilefant [2], or full project management suits that include such functionality, like Jira [3]. [8, p. 273]

Product Owner

The Product Owner role stays the same in distributed teams, but may face additional challenges regarding coordination of the different team members work. [13, p. 12-13]

[1] http://www.collab.net/products/scrumworks
[2] http://www.agilefant.com/
[3] https://www.atlassian.com/software/jira

Also van Solingen [43] does not recommend to have multiple product owners, he argues that there can just be one product owner in a Scrum team and that putting product owner proxies is more a way of avoiding problems rather than solving them since it is postponing feedback which slows down the whole development process. [43, min. 32-33] This argument is similar to the general argument on proxies from Section 4.3.1, but especially discouraged when it comes to the role of the Product Owner.

Sprints

Generally the application of Sprints in a distributed team does not differ in many ways from the application in a co-located team. But in contrast to a co-located Scrum team, where a typical Sprint may be as long as four weeks, in a distributed team the Sprint length sometimes tends to be shorter. But since setting up meetings is naturally more complex a to short Sprint time may also be tedious. A practical tip from Smite, Moe, and Gerfalk [8] is to not have Sprints shorter than two weeks and also to arrange co-located sprints at the start of a project. [8, p. 266-268]

Paasivaara, Durasiewicz, and Lassenius [66] performed a case study on three distributed development projects utilizing Scrum. Regarding the Sprint they stated that: „Due to short sprints, clear deadlines and goals it was quite clear for all the team members in our case projects what was supposed to be done during the next sprint“ [66, p. 199] Short Sprints in general increase transparency within a distributed project, particularly off-site team members benefit a lot since through the frequent interactions and updates they are kept up to date and generally get more involved. [66, p. 199]

Another case study of a very successful Scrum project reported a two weeks Sprint length. [65]

4.3.3 Daily Scrum

In a distributed team the principle of the Daily Scrum meeting does not differ from its application in a co-located team. The whole team gets together once a day to discuss the three Scrum questions (already mentioned in Section 2.4.4) in a roughly 15 minutes lasting meeting. But discussing these questions in a distributed meeting can be very challenging. One very important thing, especially relevant in a distributed setting, is to make sure every team member understands the reason of the meeting and understands the three questions. One common mistake is team members giving a status report which is not the basic intention of the Daily Scrum, it should not be mistaken as a status meeting. Its purpose is rather to provide a setting where the whole team gets together and team members mutually try to help identify and solve issues. The key point is to communicate value adding information which helps the team as a whole to make progress. [13, p. 98-99]

Another very important factor is to keep every member committed to team which is also one of they effects of the Daily Scrum:

> „The Daily Scrum meeting is where team members make a verbal commitment to the team. When they state what they are going to do today, they are making a verbal commitment to the rest of the team. The next day, when they state what they did yesterday, it is an opportunity for the rest of the team to confirm they met their commitments.“ [13, p. 100]

This aspect creates peer pressure within the team on two fronts: First it encourages individuals to complete work that is impeding progress of others and secondly it creates a liability towards the team when one continually fails delivering promised tasks. [13, p. 100] This aspect is important

in the context of the control challenge in distributed teams (discussed in Section 3.4) because the created liability poses a form of self-control within a team.

Logistics

The Daily Scrum is a short meeting that does not need much accessories, accordingly it can be conducted via different media. Most recommended is of course a face-to-face meeting if possible. In a distributed team this is normally not possible, therefore it is usually applied as mixture: Those people that are within range meet personally in the same location and the remote members are joined via text, audio or video communication. But using remote communication techniques often introduces a lot of issues and disturbances as discussed in Section 3.5.5.

But there are some adaptions and arrangements that can be made to alleviate those issues. Since the Daily Scrum is a recurring event it can be very helpful to create a steady arrangement for joining and conducting the meeting. One very important point is to always use the same telephone line or meeting infrastructure, this creates a routine for each team member and therefore makes it a lot easier to dial into the meeting. [13, p. 105]

One way of conducting the Daily Scrum is using a group instant messaging software where all team members join a group chat and answer the three Daily Scrum questions. Beside the obvious disadvantages like the further loss of communication channels, this approach also brings some interesting options. The most evident benefit derives from the used medium and the fact that all communication is written: The discussed points are automatically conserved and it is very easy to create transcripts from that chat session which can be sent to each team member by the Scrum Master. Such notes assist in keeping track of individual commitments and furthermore provide absent team members the possibility to review discussed tasks and issues.

The conduction of the chat session itself can become chaotic if every participants just post answers when they feel ready. The Scrum Master then has to sort through all messages and has to organize the different answers. Furthermore it is quite difficult to follow and hard to ask subsequent questions. A better approach would be to execute it similar to the co-located or audio based meetings, where there is a designated order for each participant and the Scrum Master sequentially asks everyone for his answers. [13, p. 105]

Another aspect is the tooling, physical boards do not work in distributed teams since remote team members do not have access to them. In distributed agile teams it is therefore necessary that the applied tools (see Section 2.4.5) are accessible by all team members. Operating the right tools is a vital success factor for distributed teams: „Tools enable team members to sketch their ideas on a virtual whiteboard. And today's development tools enable anyone anywhere to view work requests, view status of work, identify defects, and more.“ [13, p. 7]

4.3.4 Effective Collaboration

> „Remember that agile teams value working software over comprehensive documentation. Being a distributed team is not a good reason to invest significant cycles in comprehensive documentation throughout the Sprint. For collocated and distributed teams, the best way to transfer information to a new member is software code and team discussions; however, distributed teams are likely to need to rely more on documentation to communicate.“ [13, p. 122]

The *documentation* mentioned here is meant in terms of strictly defined documents that have to be created but is rather referring to the increased use of email and other written communication

systems that provide ah higher level of communication possibilities. This kind of informal documentation allows the building and preservation of mutual understanding.

Many teams use online chat systems as a basic communication technology that is used for communication but also has a documentation characteristic. Although agile values put individuals and interactions over processes and tools, using the right tools is a vital aspect of successful collaboration in distributed teams, since they often are the intermediate enables useful communication, Woodward, Surdek, and Ganis [13] entitle this as *documentation to overcome distance.* [13, p. 123]

4.4 XP

4.4.1 Introduction

Beck and Andres [30] states that XP is also suited for distributed teams, but such a situation poses the challenge of „applying XP's values, principles, and practices outside their "sweet spot," the small team sitting together.“ [30]

> „The values of XP are just as suited to multi-site development as they are to teams that sit together. Embrace feedback more tightly because of the natural isolation created by distance. Nurture communication more because of the unavailability of face-to-face, full-spectrum interaction. You'll have to work harder to achieve simplicity because you won't have as many chances for serendipitous discovery of excess complexity. Courage is just as important as it is in any other setting. Respecting everyone on a distributed team is even more important because of differences in culture and lifestyle.“ [30, p. 149]

Regarding the practices Beck and Andres [30] argues that they may have to be adopted in a multi-site project but teams should not abandon practices just for the reason that they seem difficult. Also some aspects become even more important as for example maintaining a single code base as a connection point between the different locations. The following sections will discuss various practices that are deemed especially important in a distributed team and how they are applied in such situations. [30]

4.4.2 Distributed Pair Programming

> „Write all production programs with two people sitting at one machine. Set up the machine so the partners can sit comfortably side-by-side. Move the keyboard and mouse back and forth so you are comfortable while you are typing. Pair programming is a dialog between two people simultaneously programming (and analyzing and designing and testing) and trying to program better.“ [30, p. 42]

This is how Beck and Andres [30] describe pair programming, which is a very well known practice in agile teams. Pair programming requires good communication and collaboration between two individuals, normally one person has the role of the driver who is in charge of the keyboard and implements code. The second person is called the observer or navigator and is continually reviewing the code and watches out for logical or syntactic errors. [67, p. 2]

There exist various scientific literature about the effectiveness of pair programming activities, and while they vary in their result, a recent meta-analysis on pair programming stated in their conclusion that:

> „However, with respect to the central factors expertise and task complexity, the current state of knowledge suggest that pair programming is beneficial for achieving correctness on highly complex programming tasks. Pair programming may also have a time gain on simpler tasks. By cooperating programmers may complete tasks and attain goals that would be difficult or impossible if they worked individually. Junior pair programmers, for example, seem able to achieve approximately the same level of correctness in about the same amount of time (duration) as senior individuals.“ [68, p. 1120]

If two people are geographically separated they obviously can not sit together on the same physical machine, they instead have to use tools to collaborate virtually. This Distributed Pair Programming (DPP) as Stotts et al. [69] defined it is: „that two members of the team (which may consist solely of these two people) synchronously collaborate on the same design or code from different locations. This means that both must view a copy of the same screen, and at least one of them should have the capability to change the contents on the screen.“ [69, p. 130]

Tools

The authors of a study from 2002 [69] suggest that an infrastructure for distributed pair programming should support (or at least are desirable to have) the following functions and characteristics: [69, p. 133]

- Remote desktop sharing
- Program Sharing
- File Transfer
- Session Security
- Audio Conferencing
- Whiteboard
- Textchat

Beside the basic need like a shared screen and audio communication Silva Estácio and Prikladnicki [67] suggest that the following requirements should be fulfilled by tools used for DPP [67, p. 7]:

Shared Repository: Distributed team members should access and manipulate the same files.

Support specific roles To improve the coordination between the pairs, tools should support the different roles (e.g. observer and driver) and their interactions

Gesturing Screen sharing or video conferencing tools often do not support any possibility to point out specific things or aspects, the only way to advert to something is verbally (e.g. "here in the first line"). Non verbal ways of pointing to certain information are remote cursors (also in [70, p. 532]).

Which tools to use specifically varies strongly, in general there are two different approaches: Using special purpose tools that are tailored for DPP or use general tools for screen sharing and text/audio/video communication.

Special purpose tools offer a better user interface, especially concentrating on the aspect of collaboration. Such tools may provide multiple view options depending on the role of the user, or allowing simultaneous editing of documents while also keeping it consistent. But software development is a very complex task and often requires a lot of different tools and setups. Furthermore, most developers have personal preferences of tools they like to use and limiting to using just those tools that support remote collaboration limits that variety a lot. Another drawback is that due to the complexity and variety in modern programming languages and code writing characteristics such tools may fail in providing all the necessary features that are needed to solve particular problems efficiently. [70, p. 532-533]

Examples of such tools can be found in [70] or [67], some selected examples are listed below:

- Moomba was a collaborative environment developed for distributed XP and included an editor that allowed simultaneous editing of a shared file. [71]
- COPPER was a collaborative editor designed for DPP [72].
- XecliP was a plugin for Eclipse[4] that aimed at providing DPP support. [73]

Shared desktops on the other hand do not limit that selection variety. By replicating the screen (or just certain applications) to another remote team member it is possible to use any single user software. The main disadvantage of such tools is on the other hand that they inherently are not built for the specific purpose of DPP and therefore commonly do not offer specific functionality for such utilization. [70, p. 532]

The selection of the right setup is a very important aspect and critical to the success of DPP, Canfora et al. state that „one major reason of the dismissal of the pair and, consequently, of the deterioration of the pair programming effectiveness, is the lack of an appropriate platform.“ [74, p. 21]

Potential Benefits

A study [69] performed in 2002 used an infrastructure consisting of several tools and based around shared desktop software like NetMeeting (Microsoft) and pcAnywhere (Symantec). They examined four groups each consisting of two people. The team members were separated by 30 miles apart from each other and two of those groups used DPP while the other two did not. Their results reported that the teams using DPP created 70% more unit test cases compared to the other groups, while also completing the given tasks in a lesser amount of time. The authors of the study finally concluded that they „have further suggestive evidence that the synchronous paired teams performed better than the non-paired teams.“ [69, p. 134]

A case study [75] performed with students at an American university compared four types of pairs: co-located pairs that used pair programming, co-located pairs without pair programming, distributed pairs using pair programming and distributed pairs without pair programming. The

[4] Eclipse is a wide spread open source IDE, see https://eclipse.org/

study concluded that „software development involving distributed pair programming is comparable to that developed using collocated pair programming or virtual teams without distributed pair programming. The two metrics used for this comparison were productivity (in terms of lines of code per hour) and quality (in terms of the grades obtained).“ [75]

Furthermore they reported that co-located teams did not accomplish significantly better results and that the utilization of DPP had a positive impact on teamwork and communication between the distributed team members.

Another study from Canfora, Cimitile, and Visaggio [76] used screen sharing and text messaging for communication. The lack of voice communication was noted by the test subjects and hence it is not surprising that 75% of the participants reported that they found finding an agreement during DPP more difficult than in a co-located scenario. But yet still 100% agreed that DPP improves communication and teamwork within a distributed team. The conclusion of the study suggested that is is important to make individuals familiar with each other and also to establish a behavioral protocol. [76]

4.4.3 Continuous Integration

In a software project integrating different pieces of work to a bigger project is a very important aspect and when this is postponed until the end of a project it can lead to various sorts of quality problem which often lead to delays and increase costs. Continuous Integration (CI) is a practice that addresses those risks by breaking the issue down into small increments. [77, p. 24]

> „Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly.“ [78]

Values of CI

Reducing Risks

Duvall, Matyas, and Glover [77] argues that making assumptions in software development is one of the principal problems as it increases the overall risk within a project. For example, assuming that configuration files don't have changed, that parameters of a method are always right or that third party libraries are still working. CI helps reducing this by rebuilding software every time there occurs a change in the source code which gives fast feedback to everyone on the team about what is going on in the project. Since CI integrates and performs tests several times a day, defects are detected much sooner and therefore can be fixed as soon as they were introduced.

Furthermore it is possible to track the health of a product over time, by constantly testing and inspecting the source code it is feasible to collect different health attributes over time. [77, p. 24]

Reduce Repetitive Processes

Reducing repetitiveness saves costs, time and effort. Be it code compilation, integration of databases, tests and inspections or deployment - all those aspects of software development can be automated and thus saving a lot of effort. [77, p. 30]

Deployable Software

This is one of the most obvious benefits of using CI, it facilitates the possibility to release a deployable product at any time. Even if there are just small changes in the source code, they are integrated with the code base regularly and it is not necessary to wait for any larger deployments which may cause delays. [77, p. 31]

Increase Project Visibility

This aspect is especially important in a distributed setting. A CI systems gathers and provides up to date information on build status and quality metrics or even provide reports on defect rates and completion statuses of components. Making this data visible to all team members raises the courage to bring in improvements and raises the overall involvement. [77, p. 31]

Cost Reduction

Miller [79] reported the utilization of CI in a project (with a duration of 108 working days and a total of 551 check-ins to the code repository) executed in a distributed team. The estimated costs for using CI in this project: „associated with checking in and fixing build breaks was approximately 267 hours, 7% of the total effort.“ [79, p. 291]

Furthermore they calculated the costs a hypothetical alternative would have produced. The definition for that alternative process was:

> „For each check-in a developer is required to compile, run all unit tests, the installation tests, and static analysis tools on a clean build machine. In other words developers do all the work the CI server is doing before each check-in. This is really the only alternative directly equivalent to CI in terms of ensuring the same quality of the code base and product under development.“ [79, p. 291]

Their alternative, very optimistic estimated process would have been a total project overhead of 464 hours. This 200 working hours needed in addition which would be another 5% of the total effort. In conclusion the report stated:

> „The actual cost of using the CI approach on this project was at least 40% less that the hypothetical cost of a check-in process that doesn't leverage CI but still maintains the same level of code base quality. Given the relatively small size of the product being developed and the low cost of doing a complete build this study actually gives us a number for the smallest saving likely from deploying a CI process.“ [79, p. 292]

Continuous Feedback

> „Getting the right information to the right people at the right time and in the right way - CI is the best tool for making this feedback automated, targeted, and real-time (continuous).“ [77, p. 205]

Feedback is one of the major factors in agile teams, it is essential for self organization of agile teams (as discussed in Section 2.3.1) and agile methods like Scrum use various tools to radiate information to team members (see Section 2.4.5). Therefore it is quite comprehensible that agile teams embrace CI as yet another source of feedback and a welcome information radiator.

Feedback is also a very important aspect of using CI, without getting fast feedback all the other aspects of CI become less useful. If a commit breaks a build and the report of this fact is delayed by hours or even days this prevents taking immediate action that again may prevent further damage and failures. A relevant factor is to get the right information to communicate, the necessary information always depends on the recipient and may include data about the build status, inspection and test reports and deployment results. Furthermore, the aspect of who receives what information has to be considered as well, it often is not necessary that every team member receives every available information all the time. Sending out to many messages can result in a decreased perception and people tend to miss those notices that are relevant to them. The right information to deliver to a team member depends on the role he is holding. Project managers tend to be more interested in information about resource allocation, costs and time. Technical architects on the other hand may be more interested in quality metrics and developers often are most interested in information about the code they recently checked into a code repository. [77, p. 205-208] The *right time* is often tantamount to *as fast as possible* since „the heart of continuous feedback is reducing the time between when a defect is introduced, discovered, and fixed.“ [77, p. 209]

The last thing to consider is the right way to transmit the feedback, depending on the nature of information, certain mechanisms are better suited than others.

E-mail is the major form of giving feedback in a CI system and sending e-mails can be automated within most available systems very easily. Also this feedback mechanism is easy deployable in a distributed team, information can be pushed asynchronously to the right people immediately. The disadvantage of e-mail is the risk of inundating individuals and the information sent is regarded more as annoying spam than as helpful information. [77, p. 210-201]

Ambient Devices are a great way of acquiring some physical representation.Duvall, Matyas, and Glover name Ambient Orbs[5] or more generally the use of devices that can be connected via home automation protocols. Integrating such non digital systems can also enhance an office environment and is very convenient since everyone in the room can just glance at it and gets the status of the latest metrics. [77, p. 214]

Large Monitors can be used to display automated real-time data. [77, p. 217-218] A lot of modern CI systems include extensions for displaying information on an external screen, for example Jenkins[6] with the "Wall Display Plugin" [7] or Bamboo[8] providing a built in "wallboard" feature.

Software that is running on the local machines of team members, that displays information. On a Microsoft Windows based operating system this could be an icon in the Windows task bar that displays the current build status. There also exists further software like web browser plugins or various widgets for different operating systems. [77, p. 217-221]

[5] See www.ambientdevices.com
[6] https://jenkins-ci.org/
[7] https://wiki.jenkins-ci.org/display/JENKINS/Wall+Display+Plugin
[8] https://www.atlassian.com/software/bamboo

5 Case Study

5.1 Introduction

The previous chapters summarized agile methodologies and practices and examined their application in a distributed setting. They presented different research and utilization possibilities from multiple scientific studies and papers as well as fundamental literature from the authors of the original agile manifest. Following this literature review, a case study is conducted. The goal is to gather information from teams that apply agile practices in their daily software development routine and compare those insights with the examined literature. Generally a case study can be defined as „an empirical inquiry that investigates a contemporary phenomenon (the "case") in depth and within its real-world context“ [16, p. 16]

This case study will follow the guidelines and suggestions from Runeson et al. [17] which focuses on conducting case studies in a software engineering environment. While there is a lot of literature available regarding case studies in social sciences and also in the context of information systems (IS) that methodology is in some aspects different to the field of software engineering. Case studies in IS tend to focus more on the usage context and to a lesser extent on the development and evolution of the system. Generally the term *case study* is applied in a wide variety of studies and due to the broad range of activities within software engineering processes it is adjuvant to have a specific guideline, as stated by Runeson et al. [17] :

> „There are clear overlaps with other disciplines, such as psychology, management, business, and engineering, but software engineering brings these other disciplines together in a unique way, a way that needs to be studied with research methods tailored to the specifics of the discipline.“ [17, p. 7]

Therefore, Runeson et al. [17] derived a definition for case study research in the field of computer science:

> „Case study in software engineering is an empirical enquiry that draws on multiple sources of evidence to investigate one instance (or a small number of instances) of a contemporary software engineering phenomenon within its real-life context, especially when the boundary between phenomenon and context cannot be clearly specified.“ [17, p. 12]

In this thesis the case study is on the one hand of a *descriptive* nature aiming at „portraying the current status of a situation or phenomenon“ but also of an *exploratory* nature, by investigating how teams tend to adapt certain processes. [17, p. 13-14] Its execution will follow the process depicted in Figure 5.1 and is divided into five parts.

The first step is to design the case study, determine its structure and define the objectives as well as the courses of actions that should answer those objectives. This is done in the following Chapter 5.2.

After setting the fundamental objective and the scope of the case study, the next Chapter 5.3 describes the common strategies that are used to collect data, which strategy was chosen in this

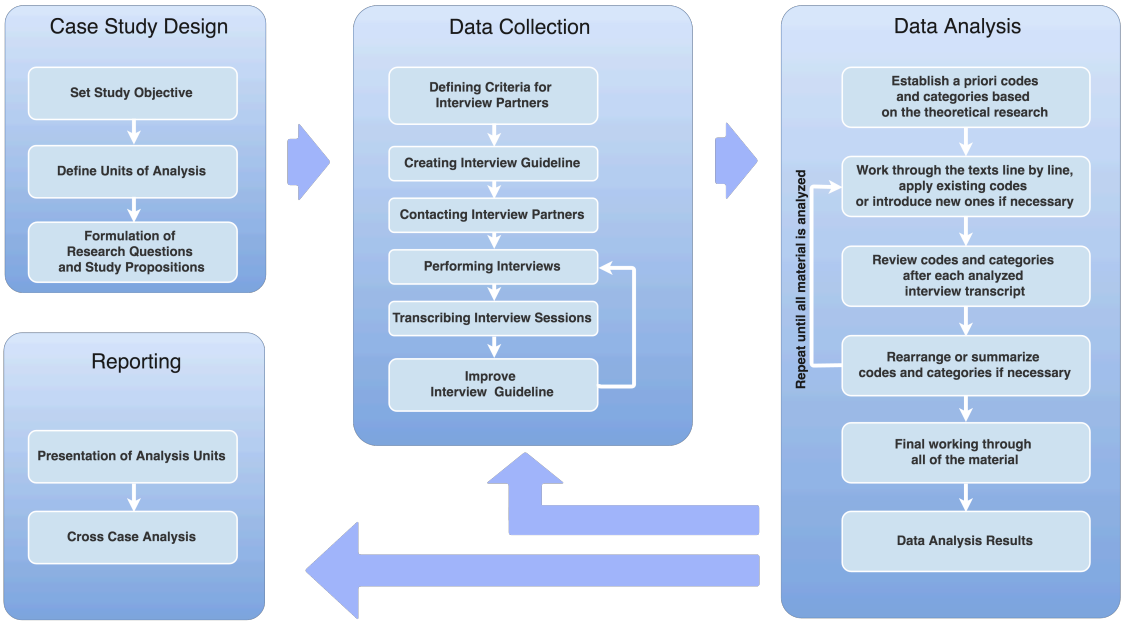

Figure 5.1: Structure of the case study

thesis and how it was applied. Additionally it introduces all the interviewed experts which are the units of analysis in the case study.

After the data has been collected it is analyzed using an empirical data analysis procedure that adapted from Mayring [80], a known expert in qualitative data analysis. As implied in Figure 5.1 it is an iterative process from data collection and data analysis because new gained insights from analyzing the first pieces of information can be used to improve the data collection process.

Chapter 6 presents the outcome of the data analysis and reports the different approaches experts in distributed teams have on the topic of agile software development. Finally Chapter 7 then summarizes and compares this analyzed and presented data.

5.2 Design of the Case Study

> „Software engineering case studies examine software engineering phenomena in their real-life settings and it is because the phenomena and setting will change during the study that such case studies require a flexible design, in contrast to for example the fixed designs of classic experiments.“ [17, p. 23]

The first step is to set up the design of the case study, the main components for a case study according to Yin [16, p. 29] are:

- Questions of the case study
- Research propositions (if any)

- Units of analysis
- Linking of data to propositions
- Interpretation criteria of the findings

Those components are discussed in the now following sections. One very important note about defined design choices is that they can be modified when new information gets available during the data collection process. Revelations during the collection phase can be sometimes very important and may alter some research design and shift focus of different questions and propositions. Yin [16] states as an example that: „in a single-case study, what was thought to be a critical or unusual case might have turned out not to be so, after initial data collection had started, ditto a multiple case-study, where what was thought to be parallel cases for literal replication turn out not to be so. With these revelations, you have every right to conclude that your initial design needs to be modified.“ [16, p. 65]

5.2.1 Objective

The objective of a case study defines what the stakeholders - normally the researcher and perhaps other industrial participants - expect to get as a result from undertaking it.

The intention for this case study is to gather insight on how teams that are distributed with low spatial geographical distance and use an agile methodology do apply (and if necessary adapt) the various agile methods. This gathered information will then be compared to the data of the literature. Due to the fact that communication technology improved rapidly over the last 10 years it may also be interesting to see if there are differences between the analyzed literature and the data gathered within this study.

5.2.2 Units of Analysis

Depending on the formulated research questions that should be answered it is necessary to define the units of analysis, and the type of the case study. A possible classification is to distinct between cases and the units of analysis within them. Runeson et al. [17] names „holistic case studies, where the case is studied as a whole, and embedded case studies where multiple units of analysis are studied within a case.“ [17, p. 26] Holistic case studies are suited for situations where there are no different logical units within a context, in a software development environment situations suited for this type would be the study of individual developers or unit testing. With more complex situations it is recommended to perform an embedded case study because „such a design anticipates the need to collect, analyze, and report on complex detail in the case.“ [17, p. 27] Those two types can in turn be performed in a single- or multiple-case context, resulting in four different types as shown in Figure 5.2.

For this thesis, the case study is constructed as an embedded single-case study, where the context is defined as *distributed development teams using agile methods* and the units of analysis are different development teams, this constellation is shown in Figure 5.3.

5.2.3 Theoretical Framework

This thesis follows the suggestion from Verner et al. [81] to perform an intensive review of literature to define a theoretical frame for the case study. The literature review may also provide additional input for the research questions and the propositions as well as insights on how other

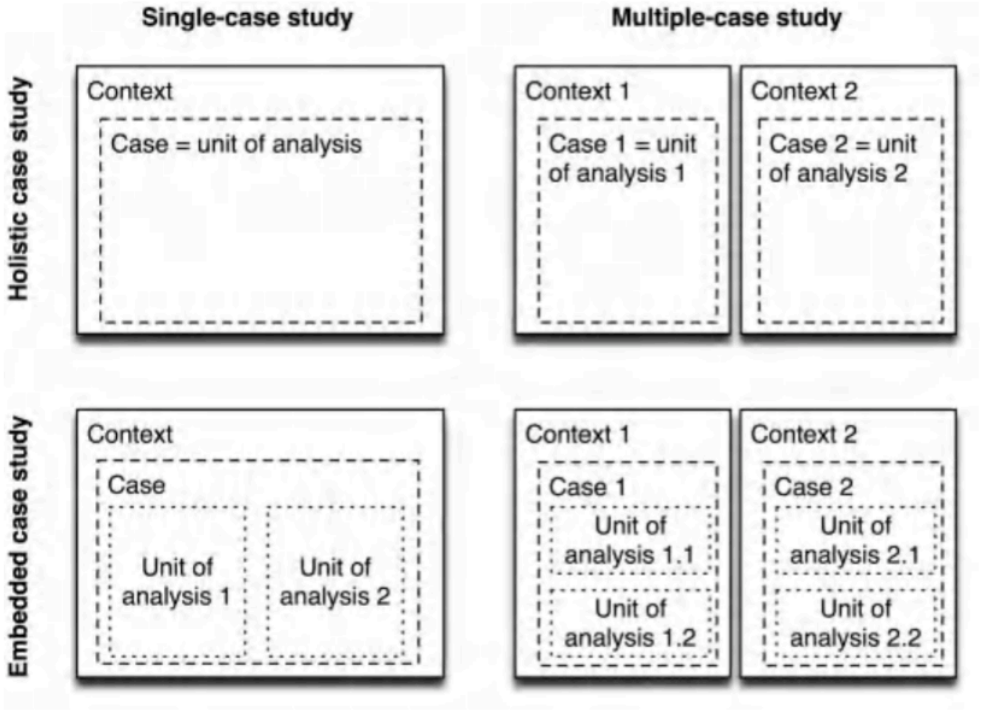

Figure 5.2: Forms of case studies [17, p. 27]

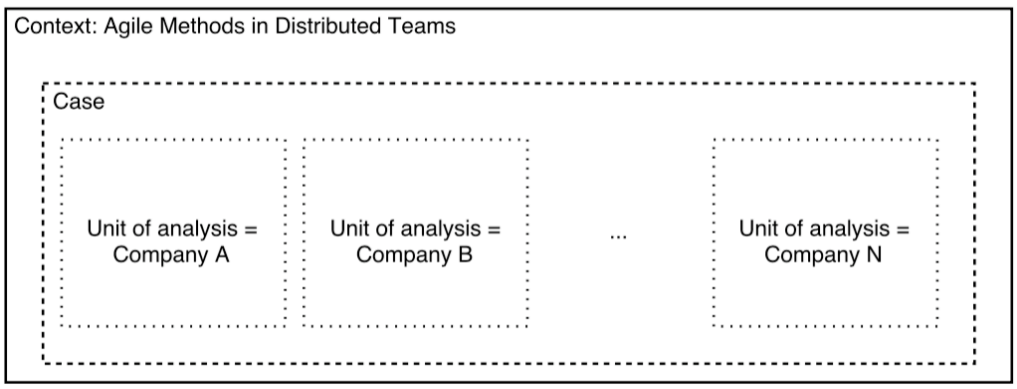

Figure 5.3: Embedded case study: Cases and units of analysis (adapted from [17, p. 27])

research was designed and conducted in that field as well as different measurement approaches. Therefore it considers previous significant work in that area of study and is used as a foundation for the following case study:

> „A comprehensive literature review and analysis with sufficient breadth and depth is used to form a solid foundation for the research. Only once an extensive prior art study has been completed can a researcher successfully identify where additional contributions are possible.“ [81]

The Chapters 2, 3 and 4 build the foundation for this case study and give an overview of established literature as well as exploring recent trends and research in the areas of agile software development, distributed teams and using agility in a distributed setting.

5.2.4 Research Questions

The research questions will be the same as stated in the introduction chapter. They are the central theme and starting point to investigate the different aspects of agile strategies and their application in a distributed environment where a team of software developers is not gathered in one physical place but located in different sites. To be able to investigate this fundamental question the following four research questions where formulated:

- RQ1: How can agile methods be used in distributed teams (limited to a low spatial and time dispersion)?
- RQ2: To what extent have the principles of agile methods be adopted to be applicable in such a distributed setting?
- RQ3: Which challenges have to be faced utilizing agile methods and how can those issues be handled?
- RQ4: Which benefits result from pursuing agile methods in such a distributed setting?

5.2.5 Study Propositions

Propositions are derived from the research questions, and „are more specific 'implementations' of research questions, providing further detail and structure to the inquiry.“ [17, p. 31] Propositions also have the benefit of better defining the measures and concepts that are used in the study, they direct „attention to something that should be examined within the scope of study.“ [16, p. 30]

The following propositions are derived from the formulated research questions and are intended to highlight different aspects of distributed agile teams:

1. The duration of iterations in distributed agile teams is similar to the duration of iterations in co-located teams.

   Section 3.4 states that coordination and control are two major issues procured by distribution. Short and frequent iteration cycles which re-evaluate the current status of a team and project are characteristic for agile methods (like Sprints in the Scrum process, see Section 2.4). Although meetings tend to be more complex to organize in distributed teams, they are not longer than in co-located team but are within the same duration ranges.

2. While for short, standardized communication situations remote communication is sufficient, face to face communication is very helpful when it comes to longer, informal meetings with multiple participants.

   Agile methods know a lot of informal communication and processes that require interaction. The author proposes that for some more straight forward activities like for example the daily standup meetings (see section 4.3.3) remote communication tools are in most cases enough and sufficient. Longer, informal communication situations with multiple participants like the review done in the Sprint Retrospective can on the other hand benefit a lot from co-located face to face communication.

3. Informal and frequent communication aspects of agile methods improve collaboration between sites and team members.

   Frequent and informal communication which is also strongly encouraged by the agile manifesto (see Chapter 2.2) which emphasizes interaction between individuals is a very important aspect in agile teams. Also Dorairaj, Noble, and Malik [14] recommend to strengthen informal communication in distributed agile teams to increase a teams success.

4. Usage of modern project management software and tools is a major factor for success of distributed teams.

   Chapter 3 introduced the challenges that distribution can bring and states that communication is a vital point. The different aspects of remote communication in section 3.5.5 indicate that choosing the right channel and tool for the right purpose is very important. This is also stated by the participants in the case study of Dorairaj, Noble, and Malik [14, p. 110] who stated that: „leveraging communication tools and techniques promoted effective communication in their distributed teams.“

5. Technical faults and limitations are posing a serious issue on distributed communication.

   Since distributed teams have to rely on remote communication techniques the infrastructure is an essential aspect. Infrastructure problems like a failing wifi or a slow internet connection can massively extend the general issues of remote communication described in section 3.5.5.

6. Beside extensive communicative skills there are no special requirements for team members in distributed teams compared to co-located teams.

   Agile methods generally favor individuals and interaction of individuals, which indicates that the communication in agile teams is very high. Since distance negatively impacts communication (as stated in section 3.4), team members in distributed teams need to be even more aware of the necessity of communication. Beside this requirement there are no further skills that are necessary for team members to be successful in a distributed environment compared to co-located situations.

These research propositions as well as the research questions combined with the literature research serve as starting point for the data collection and data analysis activity. They as well as the initial research questions are evaluated in Chapter 6.

5.3 Data Collection

The general method of data collection is decided in the design phase of the case study, but during the data collection phase itself there may occur unexpected situations and opportunities for gathering additional data [17, p. 32]. Lethbridge, Sim, and Singer [82, p. 313] distinct three degrees of

Figure 5.4: Procedure of data collection

data collection methods. The first degree are techniques that require direct involvement of participants: e.g. interviews, questionnaires, brainstorming, focus groups, et cetera. The second degree demands just indirect involvement by accessing the environment of individuals, it includes analysis of tool usage logs, databases or performed work. The final and third degree does not require contact to individuals but rather only access to work artifacts like documentation or source code.

This case study will employ first degree data collection, particularly interviews and group interviews. The interviews are conducted in the form of semi-structured interviews. Figure 5.4 shows the general flow of the data collection approach.

5.3.1 Interviews

Yin [16, p. 110] names interviews as „one of the most important sources of case study evidence.“ Furthermore, interviews are by far the most used methodology when it comes to empirical research in the social sciences, see [83, p. 434-435].

Also in the field of empirical research in software engineering interviews are one of the most used approaches and most of the existing case studies involve at least some kind of interviews either for validation other data as a primary data source. The reason for this is that a lot of knowledge important to specific research is only available in the minds of the individuals situated in the investigated cases. There are diverse ways of conducting interviews, varying in the length of the interview sessions, the kind of questions asked, and general structuring. [17, p. 50]

	Unstructured	Semistructured	Fully Structured
Typical foci	How individuals qualitatively experience the phenomenon	How individuals qualitatively and quantitatively experience the phenomenon	Researcher seeks to find relations between constructs
Interview questions	Interview guide with areas to focus on	Mix of open and closed questions	Closed questions
Objective	Exploratory	Descriptive and explanatory	Descriptive and explanatory

Figure 5.5: Interview types [17, p. 51]

Strengths and Weaknesses

The advantages of interviews are that they are interactive and the researcher is able to clarify unclear questions and react to unexpected responses. [82, p. 320] Furthermore they are highly targeted and directly focus on the topics of the case study and also provide personal views as well as explanations of interviewees. [16, p. 106]

Disadvantages of interviews as a method of data collection is that they are often time and cost inefficient: „Contact with the respondent needs to be scheduled and at least one person, usually the researcher, needs to travel to the meeting (unless it is conducted by phone but this lessens the rapport that can be achieved). If the data from interviews consists of audio or video tapes, this needs to be transcribed and/or coded“ [82, p. 320], although it is noted that note taking may be a suitable substitute if done carefully. Further weaknesses of interviews are the risk of bias because of poorly uttered questions, response bias or the risk of reflexivity - meaning that the interviewed person answers in a way he or she thinks the interviewer wants the answer to be. [16, p. 106]

Interview Types

There are multiple classifications of interview types, one is the distinction by the degree of structure, a coarse distinction is shown in Figure 5.5.

The range from fully structured (standardized) to unstructured (non standardized) interviews is not strictly differentiated but more a rather smooth transition. Fully structured interviews contain just closed questions with predefined answer possibilities and are asked in a defined order. Unstructured (or also called open-) interviews on the other hand do just have minimal specification, in extreme cases also just an interview topic. The more structured an interview is, the better are the claims on validity, objectivity and reliability. But such rigid structuring also comes at a price, with just closed questions, there is little to none access to information beyond the predefined answer possibilities. In most cases it is feasible to utilize a mixed approach, using both closed as well as open questions. Examples for lesser structured interview types would be the semi-structured interview, a focused interview or a narrative interview. [83, p. 437-438]

A narrative interview would be an unstructured form, which aims at gaining subjective statements from the interviewed person, the interviewer often just names the topic and encourages the interviewee to talk about it. The basic idea is to trigger a narrative dynamic which should lower the

inhibition threshold and leads to revelation of information that would not have been revealed in other forms of inquiry. [83, p. 540-542]

This case study will use a problem-centered, semi-structured interview, where questions are prepared ahead in an interview guideline. But in contrast to fully structured interviews, the order of the questions is not fixed it is rather the development of the individual conversation during an interview session that influences which question is asked by the interviewer. The prepared guideline is to make sure that all necessary questions are asked during the interview session, generally such an interview technique allows improvising and more exploring during a session. [17, p. 51]

In contrast to the narrative approach, the problem-centered interview technique is not about letting the interviewee talk and drift in any topic without control but rather guides the session by actively participating and using a predefined guideline. Furthermore it is recommended to begin such an interview with some quantitative questions that cover different classification information. This should be done separately to not disturb the main part of the interview with interposed questions about statistical data. [83, p. 542-543]

The Interview Session

Recording the interview sessions is a very convenient way of preserving the information as well as it allows a more detailed analysis later on, even though the process of transcribing takes a lot of energy and time. Taking notes during the interview session is often not that accurate and furthermore it could pose a distraction and disturb the conversation. But a recording device can just be used if the interviewed person agrees with it, so it is very important to open an interview session with first asking the interviewee for permission to create an audio record. Furthermore the interviewer must not make the mistake of thinking that the existence of a recording replaces attentive listening. [16, p. 110]

The duration of the interview session can range from prolonged interviews that are taking place for two or more hours in either a single session or split into multiple sessions to shorter interviews that are not taking longer than one hour. In such cases the interview session can still remain open ended, but it is recommended to follow the case study protocol more closely but allow a more open discussion to the end of the session. Interviews do not have to be a one on one session, if there are two or more persons interviewed at the same time it is called a group interview. Interviewing multiple people at the same time can result in a discussion about aspects of the case study and it is the interviewers job to moderate and try to bring out the views and opinions of each individual in the group. [16, p. 111-112]

After an interview there are multiple post interview activities that have to be carried out. After recording the material has to be transcribed into a text document before it can be analyzed. This transcribing is a time consuming assignment but often brings new insights that are discovered during this transcription process. [17, p. 53]

5.3.2 The Interview Guideline

The interview guideline was developed with the research questions and the propositions in mind and the topics and questions are trying to gather on the one hand specific experience and approaches that the interviewee or the team he or she speaks for encountered as well as general thoughts on the thesis topic. The used interview guideline can be found in the appendix A.1.

5.3.3 Other Data Sources

Beside interviews there are several other sources of evidence that could be utilized in a case study but apart from the interviews as a data source this case study does not use any other other data sources because it would exceed the resources of the thesis, but for the sake of completeness they are shortly introduced in the following enumeration:

Documentation
Documents may play an explicit role in data collection within case study research. Examples for useful documents would be for example letters, e-mails, agendas, administrative documents, formal studies or evaluations, news clippings or other media articles, etc. One very important usage of documents is to substantiate and support evidence gathered from other sources. [16, p. 107]

Archival records
This source, very similar to the previous source, can also be used in combination with other evidence sources. The utilization of archival data should be done carefully and it is suggested to „ascertain the conditions under which it was produced, as well as its accuracy. Sometimes, the archival records can be highly quantitative, but numbers alone should not automatically considered a sign of accuracy“ [16, p. 109]

Direct observations
The fact that a case study is meant to take place in a real world situation creates the opportunity to directly observe, such observations range from casual to formal activities of data collection. Less formal observations also can be made while gathering other forms of evidence, for example while conducting interviews. [16, p. 113]

Participant observation
This aspect is a special form of observation where the researcher is not just in the role of a passive observer, on the contrary, he or she may take on various roles within a situation in the field and actively participate in the actions that are being studied. [16, p. 115]

Physical artifacts
The last major source of evidence are cultural or physical artifacts, like technological devices, artworks, instruments or tools, or other physical evidence. Overall this kind of evidence has often less potential importance, but sometimes they can be of relevance in an overall case. [16, p. 117]

Figure 5.6 summarizes the strengths and weaknesses of those most used sources of evidence in case study research. For each source the procedures of collecting has to be developed independently and it has to be ensured that each is used properly.

5.3.4 Selection of Analysis Units

The search and selection for suited interview partners, representing good units of analysis was a very essential aspect of the case study. To reduce the risk of a biased selection, a rough profile of suitable characteristics was defined which have apply to people and teams to be suited as analysis units:

- Having at least one permanent team member who is not located with the rest of the team.

SOURCE OF EVIDENCE	Strengths	Weaknesses
Documentation	• Stable—can be reviewed repeatedly • Unobtrusive—not created as a result of the case study • Specific—can contain the exact names, references, and details of an event • Broad—can cover a long span of time, many events, and many settings	• Retrievability—can be difficult to find • Biased selectivity, if collection is incomplete • Reporting bias—reflects (unknown) bias of any given document's author • Access—may be deliberately withheld
Archival records	• *[Same as those for documentation]* • Precise and usually quantitative	• *[Same as those for documentation]* • Accessibility due to privacy reasons
Interviews	• Targeted—focuses directly on case study topics • Insightful—provides explanations as well as personal views (e.g., perceptions, attitudes, and meanings)	• Bias due to poorly articulated questions • Response bias • Inaccuracies due to poor recall • Reflexivity—interviewee gives what interviewer wants to hear
Direct observations	• Immediacy—covers actions in real time • Contextual—can cover the case's context	• Time-consuming • Selectivity—broad coverage difficult without a team of observers • Reflexivity—actions may proceed differently because they are being observed • Cost—hours needed by human observers
Participant-observation	• *[Same as above for direct observations]* • Insightful into interpersonal behavior and motives	• *[Same as above for direct observations]* • Bias due to participant-observer's manipulation of events
Physical artifacts	• Insightful into cultural features • Insightful into technical operations	• Selectivity • Availability

Figure 5.6: Six sources of evidence in case study research [16, p. 106]

- At least one office or site has to be located in Austria or Germany.
- Teams should define their process of working and development as being 'agile'.
- There is no strict criteria for the role of the interviewee, the only important thing is that he or she is or was directly involved in the teams he is talking about.

With this criteria in mind, there was an extensive search for suitable companies and individuals. There were multiple approaches to find suitable contacts like doing search engine inquiry for distributed teams or checking vacancy offer websites for companies that have job offers in multiple sites. Overall 60 candidates were contacted from which nine agreed to an interview.

5.3.5 Presentation of the Interviewed Cases

This chapter introduces the interviewed experts and the teams they speak for. Some interviews were performed as group interview with two people being interviewed at the same time. Furthermore in some situations it is necessary to distinguish between personal experience an interviewee might talking about that is not based on the current setting, such situations often refer to previous experiences and teams they were part of. The roles of the interviewed individuals differ, but the common ground is that most of them are either in leading positions or experts in the field of agile methodologies and therefore in consulting roles for other teams. All data is anonymized and the

different units of analysis are referred to as Alpha, Beta, Gamma,... cases, furthermore all the names that occur in the interview quotes are substituted by other names to further protect the privacy of the interviewed case units. Table 5.1 is giving an overview and following is an introduction of the interviewed experts and the teams they represent. This section, as well as the discussions in Chapter 6, includes various direct citations from the interviews. To improve readability those interview passages where translated into English by the author, the original quotes in German can be found in the appendix Section A.2.

Alpha
Alpha is a software engineering company with overall nine software engineers, some of them part-time employees. The company has two main fields of business, on the one hand it develops and sells in-house software products and on the other hand it provides software engineering services for customers. Team members are split over three different locations within Austria, where in each location are between two and four people. There are no fixed teams or team sizes, the teams are composed depending on the project requirements and are changing from project to project.

The interview was done as a group interview where both the CEO and CTO were taking part in the interview of the same time. Both are founding members of the company and are located in two different sites. Interviewee A is mainly responsible for project management, marketing and distribution. The main focus of the second person (B) is more on technical aspects and infrastructure, as well as expert for complex software engineering tasks.

The company was founded as distributed company with two different locations in the beginning, therefore the whole infrastructure and processes were set up with the distributed team situation in mind.

Beta
Beta is a software agency that does software projects for industry clients. The interview was held with the CEO who is the founder of the company and responsible for all projects. They have around 15 employees where some of them are permanent staff and some are hired temporary because of specific know-how or the need for more manpower.

> „We have started as a software agency and we are actually developing software for the industry sector where we also have the most projects going on. The development of web portals and the online marketing have been added because there is currently demand for it. What we do: We design, develop and market digital products - that is our unique selling point.“ [Beta #1 - CEO]

Gamma
The company of Gamma is a big software company with over 200 employees who's area of operations among other things include software development, consulting, coaching and project management. The interview partner of Gamma is the team leader of an eight software developer team that is distributed in two major cities in Germany. There is no fixed number of individuals per site, this is rather fluctuating depending on the need and availability of employees on the two locations.

Delta
Delta is a company that operates in multiple countries and has more than 70 employees where roughly the half is employed in the software engineering department. The software development is done in two locations in Austria as well as one site in Slovakia, where one of the Austrian sites

Unit	Interviewed Expert(s)	Team Size	Number of Sites	Countries	Agile Process Model
Alpha	CEO and CTO	2 to 5	3 Sites	Austria	Scrum
Beta	CEO	3 to 5	4 Sites	Austria	Scrum
Gamma	Team Leader	8 to 10	2 Sites	Germany	Scrum
Delta	Two Team Leaders	3 and 7	3 Sites	Austria and Slovakia	Scrum mixed with Kanban
Epsilon	Department Manager	Varying	4 Sites	Austria and Germany	Scrum and Kanban
Zeta	Department Manager	Varying	Multiple Sites	Austria, Slovakia, Poland and India	Scrum and Kanban
Eta	Scrum Master	4 to 7	4 Sites	Austria and Germany	Scrum
Theta	Agile Coach	5 to 10	3 Sites	Austria and Hungary	Scrum
Iota	Team Leader	5 to 8	Multiple Sites	Germany and multiple Neighbors	Kanban

Table 5.1: Overview of the interviewed experts

is the headquarter. The interview was held with two team leaders, one of them joined via a video call. Both are responsible for their own team, one consisting of three and one of seven people and both teams apply a similar process. Both teams have at least one member from the Slovakian site.

Epsilon
The company in Epsilon is a contractor in the field of software quality engineering. The Organization offers different types of services: First are trainings in the area of quality engineering for other companies. The second aspect are consulting services in the area of software quality. Thirdly is the evaluation of tools and advice for certain quality requirements as well as migration and training for certain tools and setups. The fourth and largest section is the operational services area which is a team of 20 individuals that is working in the field of operational testing. This division handles all stages within a test process: test management, analysis, design, execution, reporting and other tasks if required. The interview evolves around this team that is distributed over five different locations in Austria as well as Germany.

The expert interviewed is the division manager of this team, he is responsible for that group that within forms multiple teams with each team having a technical project leader and a variable group size depending on the requirements.

Zeta
The expert interviewed in Zeta is a division leader who has experience in multiple distributed teams. He has started as software engineer and had several positions from business analyst, technical analyst, team leader, project leader and is currently in charge of a software development department at a larger company. He has experience with far distributed teams that have members outsourced in India as well as teams with low spatial distance where the team is located in Austria and neighboring countries like Slovakia or Poland. In the interview session he is not referring to one specific team but to multiple teams he had worked in in the past.

Eta
Eta is an IT company that is developing in-house software products as well as developing software for customers. They have multiple teams which are all using Scrum as process model. Apart from the headquarter in Vienna the company has two other sites in Austria as well as one office location in Germany, some of the teams are co-located in Vienna but also multiple teams where the team members are composed from multiple locations.

The interviewee is involved in multiple bigger projects and also responsible for communication with external stakeholders.

> „For bigger projects I am a Scrum master - from the customers point of view maybe project manager - but in the agile world it is a Scrum master role, maybe in a slightly altered form. And I am a lot involved with customers, my job title there is that of a Customer Relationship Manager.“ [Eta #13- Scrum Master]

Theta
Overall there are around 30 people in the organization of Theta, divided into five software engineering teams sized between 5 to 10 people each. The interviewed expert is a certified Scrum master and describes himself as an agile coach at Theta, where he looks after five software development teams:

> „Since the beginning of my career in this company I had the role of a Scrum master, lately this role changed towards being an agile coach. Also before that I was operating as a Scrum master and nowadays I am working with nearly all teams that are involved in software development.“ [Theta #21 - Agile Coach]

Teams are located in two major cities in Austria as well as one office site in Hungary. Previously they also had team members in Russia which the interviewee also refers to in some cases. The team formation varied over the time, there are some teams that are completely co-located and other teams that are spread between two or - in more uncommon cases - also distributed over three locations.

Iota
The expert of the Iota started working with distributed teams in 2010 and has experience as a team leader and CTO of various companies and startups where he was involved with distributed teams in different situations. During the interview there is no particular team the interviewee refers to, but rather multiple teams and situations he has experienced as a team leader.

5.4 Data Analysis

Because case study research is a flexible method, it is common to use qualitative data analysis methods, with the basic objective to derive conclusions from the gathered data while keeping a traceable chain of evidence.

> „Analysis of qualitative research is characterized by having analysis carried out in parallel with the data collection. This is necessary since the analysis can reveal the need for collecting additional data. This is one reason why systematic analysis techniques are needed. Since it is constantly ongoing it is necessary to know exactly what was found out when in the analysis. Being systematic is one condition needed in order to present a chain of evidence.“ [17, p. 62]

Generally the steps from collecting data to analysis to reporting is done in iterations, it is a repeating circle of going from data collection to analysis and repeat back.

5.4.1 Qualitative Content Analysis

The initial point is a data-set in any form collected by the researcher, in this case the transcripts of the performed interviews. To analyze the material the inductive category formation method, developed by Mayring [80], was adapted to the requirements and situation of this thesis. This is because some steps that would be necessary in the original method as described by Mayring [80] have already been done in the definition of the case study and other other steps are not applicable in the scope of this thesis. The process model of the developed method is shown in Figure 5.7.

The first step is to develop *codes* which are used to arrange and classify information, this process means that „parts of the text is given a code representing a certain theme, area, construct, and so on. One code is usually assigned to many pieces of text, and one piece of text can be assigned more than one code. Codes can form a hierarchy of codes and sub-codes. The coded material can be combined with comments and reflections by the researcher (i.e., memos).“ [17, p. 62] There are multiple levels of formalism when it comes to analyze and structure reported data and Runeson

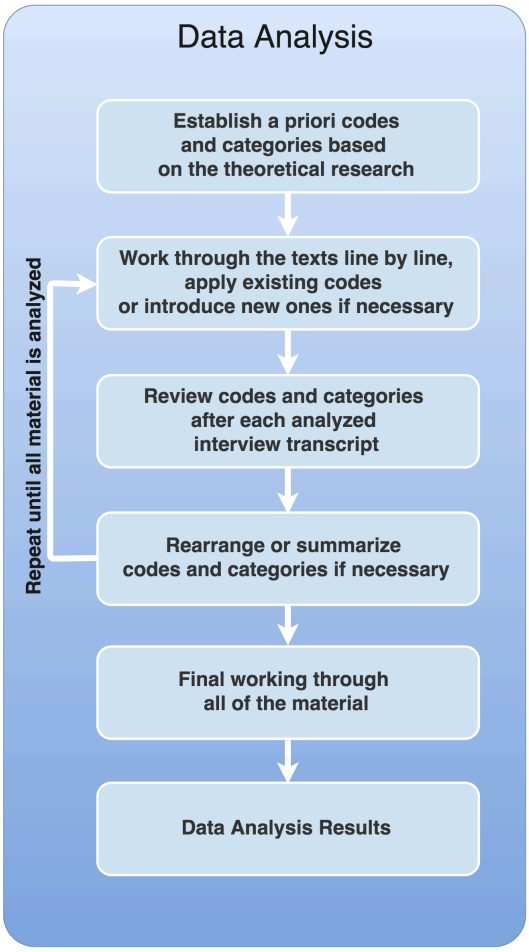

Figure 5.7: Content analysis: Steps of the inductive category development method

et al. [17, p. 64] suggest to use either *editing or template approaches*. Such approaches include *a priori* codings which are defined based on the previous conducted analysis or literature review.

Using this technique, some a priori codes based on the research questions and propositions were defined and used to analyze the first interview. During this process the codes were adapted the material was coded again. This is an iterative process, which is similar to the overall iterative procedure of collecting and analyzing data as described in the beginning of Section 5.4. Generally the codes and their grouping to main categories is repeated multiple times from step four back to step two, following the flow in Figure 5.7.

When the whole material is then worked through it results in multiple main categories that group the codes together, those final categories and their content are shown in Figure 5.8 as well as explained in detail in Section 5.4.2.

5.4.2 Categories and Codes

The coding of the transcribed data was done with a supportive data analysis tool called QCAmap[1]. Mayring [80], who is also the author of the base method that was adapted and used for the qualitative analysis of the interview data is also one of the main authors of this tool. Figure 5.8 shows the

[1] https://www.qcamap.org

codes that were used to structure and analyze the transcribed interviews, as well as the categories those codes then were assigned to group interview passages to different topics. Some categories also contain sub-categories, this is to improve the readability and emphasize some aspects within a broader topic.

The following list contains the codes that were used during the analysis phase:

- process-model
- control-aspect
- coordination-aspect
- process-model-history
- iterations
- agile-practices
- knowledge-transfer
- information-radiator
- documentation
- planned-processes
- not-practicable-processes
- communication
- communication-tools
- hardware
- meetings
- face-to-face
- language
- distribution
- reasons-for-distribution
- far-distributed
- distribution-advantages
- distribution-disadvantages
- cultural-aspects
- configurational-distance
- technical-problems
- teambuilding
- employee-requirements

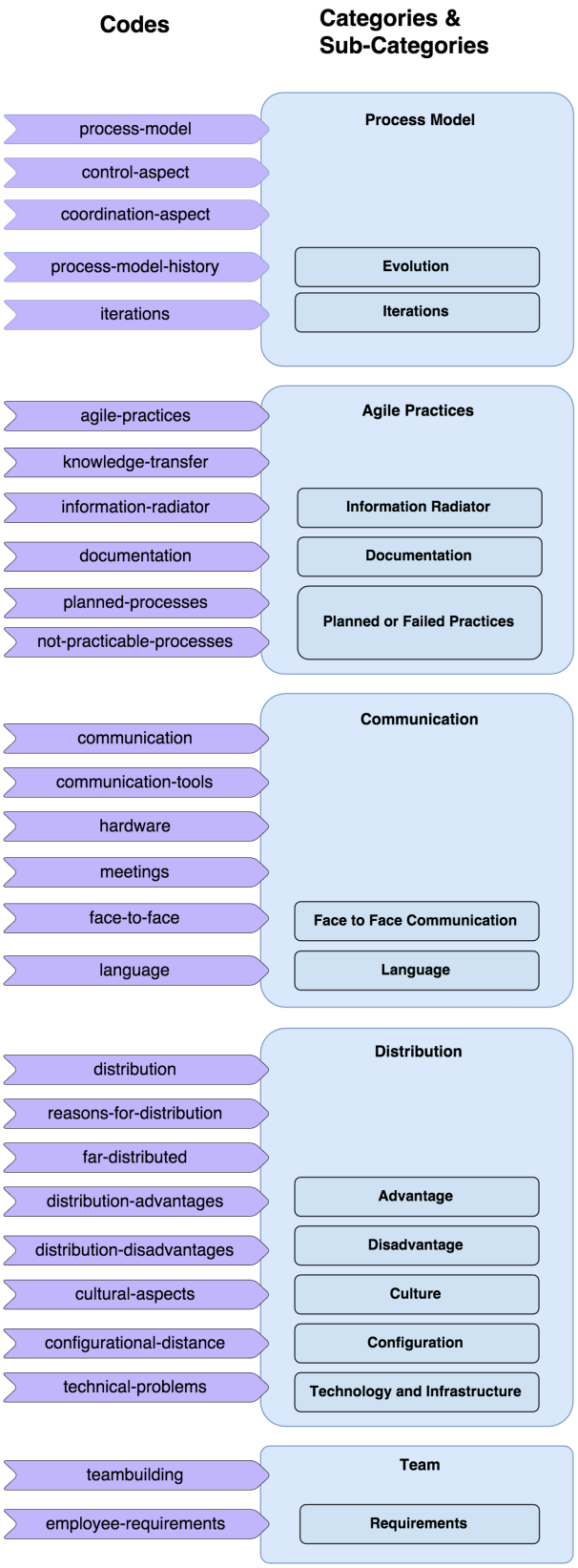

Figure 5.8: Codes used to categorize the interview data

The following categories are used to present the evaluation of the data. Some categories contain sub-categories which are more specific questions within its bigger category. This introduction of sub-categories intends to improve the readability to the reader since it introduces more structure to the multiple topics. Figure 5.8 shows the exact codes that were used as well as to which category or sub-category they belong. In the headlines of the categories the codes that were matched to that category are written in brackets.

Agile Development ***(process-model, control-aspect, coordination-aspect)***

This category refers to the general process model the interviewed person has experience with or is currently applying in a team. It is not limited to one single process model, some teams may apply different approaches for different situations, and some interview partners may be responsible (or have experience) for multiple teams applying different process models. This section analyzes how the main process model works, which important components there are and which tools and techniques the cases use to support the process. To better graduate the information, the following sub categories were introduced:

- **Evolution** ***(process-model-history)***
  What was the history that lead to the current situation? Was the team using any other approaches in the past and if so, why did it end up using the current solution?
- **Iterations** ***(iterations)***
  What is the duration of iteration cycles? Why were they chosen as they are?

Agile Practices ***(agile-practices, knowledge-transfer)***

What is the opinion on different agile practices, which were implemented and was there a need for alteration or adaption to be applicable? This section examines agile practices like pair programming, continuous integration or code reviews.

How is the transfer of knowledge done within the team and is there also knowledge transfer out of the team into the rest of the organization?

- **Documentation** ***(documentation)***
  What is the take on documentation? Is there a difference in documentation due to the distribution?
- **Information Radiator** ***(information-radiator)***
  Are there any experiences with the usage of information radiators?
- **Planned or Failed Practices** ***(planned-processes, not-practicable-processes)***
  Where there practices that have failed due to the distribution or are there any plans to introduce new practices in the future?

Communication ***(communication, communication-tools, hardware, meetings)***

This is one of the main topics during the interview and multiple questions are referring to this aspect. How is communication within the team and between multiple sites done, which meeting types are existing, and which communication channels are used by the team members? Is the communication structure changed depending on the occasion and which communication channels are used for different meeting types?

- **Face to Face Communication** ***(face-to-face)***
  How is the situation regarding face to face communication, how often does it happen in a distributed team and how valuable is it in the eyes of the interviewee?

- **Language** ***(language)***
  What role does the language of team members play in distributed teams? Are there difficulties if there are different native tongues within one team?

Distribution ***(distribution, reasons-for-distribution, far-distributed)***

This section explores general aspects of the distribution and also explores the initial cause of the team ending up in a distributed setting.

- **Advantages** ***(distribution-advantages)***
  Which advantages can be seen coming from this distribution?
- **Disadvantages** ***(distribution-disadvantages)***
  Which disadvantages can be seen coming from this distribution?
- **Culture** ***(cultural-aspects)***
  Is culture a factor in distributed teams limited to the constraint described in Section 3.2?
- **Configuration** ***(configurational-distance)***
  Relating to Section 3.3.4, this topic is dealing wit the importance of configuration within a team. Are there issues arising due to certain constellations?
- **Technology and Infrastructure** ***(technical-problems)***
  Is the available infrastructure or available tool-set posing problems?

Team ***(team building)***

This section is dealing with aspects within the team as well as team members. It is for looking closer at the team itself, requirements for team members and how team building is done.

Requirements ***(employee-requirements)***
Are there special requirements that team members have to have when working in a distributed setting?

6 Presentation of the Cases

This chapter presents the results from the data analysis of the different units of analysis.

6.1 Alpha

Agile Development
Alpha uses a Scrum like process model, which uses some basic concepts of Scrum as well as other agile techniques. Due to the company size of 10 people that form multiple, overlapping teams which are working on different projects they made some alterations and simplifications, but the reason for this is not the distribution.

> "We simply have few employees split to multiple Scrum projects and there you would need to have multiple roles for the different projects - that would be far too much overhead. In some situations there are projects where the team consists of two people, there it is just not possible. But that is more a problem due to the proportion of employees to the number of projects." [Alpha #1 - CEO]

They have fixed length sprints for each project and within a sprint they utilize daily standup meetings for each project and have a weekly "Jour Fixe" meeting where the whole team is participating. They have done Retrospective meetings in the past but stopped doing them, but account this that they just were not practical and the stop of holding them has nothing to do with the distribution. Furthermore it is not really enforced to close all issues within a sprint, there is not much overall planning between different projects therefore it may happen that work with lesser priority is left over at the end of an iteration.

> „We have discovered that for us - due to the multiple projects we have and their different priorities - the story points are just for a basic estimation on how hard an issue is, but it is not important for us how many issues we can do during a Sprint because that is very dependent on the number of projects that are done at that moment and there it comes to adjustments depending on the priorities.“ [Alpha #2 - CEO]

Regarding the tooling they use Jira for tracking Scrum boards, the Sprint Backlogs and the Product Backlog as well as error reports. Furthermore they use several other tools discussed in more details in the respective sections.

One important thing that was mentioned is the constant need to adapt and improve the process. They noticed that the geographical distribution needs to be met with more explicit communication to keep the work and knowledge status synchronized between different sites.

> „You have to adapt and improve the process constantly, that comes up very clear lately. We have made some changes during the last year where we have noticed that you have to do certain things because of the geographical distribution that you would not have to do otherwise. When it comes to synchronizing so everybody keeps in

> touch with what is happening in other locations, what do the people there do at the moment. Like this sprint planning - like in Scrum - so you better integrate and know what we do now and what we have to do next. So you do not work separated from each other.“ [Alpha #3 - CTO]

- **Evolution**
  They reported that they have started working in an somewhat Agile way right from the beginning. They did milestone planning and iterative development with the goal of getting early working prototypes to show for customers and getting frequent customer feedback. Over the years they kept adapting and improving their process by shortening the iteration length and experimenting with other activities.
- **Iterations**
  Their standard Sprint duration is two weeks and they report good results with this length. Initially they started with iterations lasting between one and two month but reported that they soon realized that such periods are too long and did not work for them. The short iterations combined with the daily meetings are the two aspects that brought the most improvement to their process.

Agile Practices
They use a modern setup with Continuous Integration servers for all projects as well as error reporting and a ticketing system. They also have a setup to quickly build test versions of software to be able to get early and contemporary feedback from customers as well as other test users.

One very important point that comes up during the interview is their way of doing code reviews: They do not do classic code review sessions but use so called Pull Requests - a functionality some code repositories provide - for this activity. This practice has beneath the improvement of code quality multiple other benefits like a communication as well as knowledge transferring effect.

> „What I find also very important are the code reviews in the form of pull requests. The way we have set it up is very easy to use and other team members learn what their team mates are doing. It is of course also very helpful in terms of code quality but it is just as useful to spread knowledge about what other people are currently working on.“ [Alpha #4 - CTO]

They are that happy with this practice because they feel that it in a certain way it substitutes the need for bigger code reviews and can detect and correct flaws earlier.

> „It is a better form of code reviews, it substitutes classical code reviews by about 90 percent because when we sit together and say: ’Today we are doing a code review’ then it is often already a very big piece where it is to late to do a review.“ [Alpha #5 - CEO]

- **Documentation**
  For documentation they use Evernote[1] as well as the issue tracking functionality of Jira. Previously they used Atlassians Confluence tool but moved on from it due to some issues they experienced. Although they document some important things they do not have dedicated

[1] A software solution for taking and sharing notes, see https://evernote.com

documentation for each project but rather see the issue tracking as a form of it. Smaller decisions, be it in design or functionality are discussed and therefore also documented within the ticketing system while bigger or more important principles are often written down as specific documents.

- **Information Radiator**
  They have nearly all of their information stored and distributed digitally. The Scrum board functionality offered by the Atlassian tools is used for visualizing which state a working package is. But also one interviewee expressed that he would also like the idea of a physical board with post-it notes stuck on it. But this is not practicable due to the distribution but also due to the fact that some team members occasionally work from home where they then would not have access to such information.

- **Planned or Failed Practices**
  They did not name any specific practice that failed because of the distribution but rather told that the process itself and single tasks are constantly changing and adapting and that they are continually experimenting with new practices to find out what is working best for them.

Communication
They use an online messaging application for the basic team communication called Slack[2]. This tool allows one on one text communication as well as groups which they use for different project teams. Furthermore it is integrated with their ticketing system and it also possible to integrate other tools like CI messages or error reports.

Beside the text communication that is the basis for every team member the team uses Skype for situations where two people need to talk and Goto Meeting[3], an audio and video meeting tool with screen sharing and remote desktop control transfer functionality for meetings where more team members attend.

The weekly held Jour Fixe meeting where the whole team is present is held as a video conference and the individuals in each site gather in a meeting location (if there is any) while remote people join from their current work station. In the main office there is also a big TV screen used to enhance team member visibility as well as dedicated microphones to enhance the audio quality.

The Daily Standup meetings within the projects are held via audio communication without video support. An interview partner also stated that he sees video as an obstacle in short common meetings like the Daily Standup since it is just supposed to quickly update every team member and be of short length.

- **Face to Face Communication**
  For daily communication as well as collaboration face to face communication is not that important. The highest importance of physical contact is more in team building aspects than in daily work.

  > „It is for sure necessary for the team constellation that you see each other and talk to each other, also about not work related topics. But it is not necessary all the time when it comes to implementing something.“ [Alpha #6 - CEO]

  Also for complex topics like architecture design or other strategy planning they prefer to gather the needed people in the same physical location to cope with a task.

[2] https://www.slack.com
[3] http://www.gotomeeting.com/

> „We want to collectively sit together and discuss a topic where we will talk for an hour or two and also do brainstorming. This is not working very well with GoTo meeting because it is not as interactive as being co-located. For such situations it is nice to sit together, but this are rather exception because normal project meetings, standup meetings or Sprint planning are working quite well with GoTo meeting.“ [Alpha #7 - CTO]

- **Language**
  Since all team members share the same native tongue, language is not a factor that came up during the conversation in any way.

Distribution
Alpha started out as a distributed team so the setup and everything they introduced was already with the distribution in mind. They have been encouraged by the enhancement in remote work infrastructure and technology and the fact that this way of working was adopted by other teams as well, so they decided to set up their team distributed over two locations.

> „We were distributed from the very beginning. Thereby that Andreas was in Linz and we were here, we have been from the beginning, from the very first minute on distributed and have therefore set up everything we did from this perspective. In parallel to this the field of remote work has advanced and because of this we have stayed with this way of working because we have seen that other people do the same thing.“ [Alpha #8 - CEO]

- **Advantages**
  Since Alpha started out as distributed team from the beginning they experienced various advantages of this way of working. The first argument is of personnel nature: Because of the fact that the main office is located in a smaller town it is not easy to find suitable employees and by having another location to offer as a workplace they increase their chances of finding developers.

  > „The topics in software development keep increasing and it gets more and more difficult in this broad spectrum of technologies to find a specialist for a certain subject. Somebody who is located at location X, wherever that may be. Therefore we said: Ok, we have gained experience in the distributed working since our beginning, then we stay on course and benefit from the advantages and so there is no regional or temporal condition for working at Alpha.“ [Alpha #9 - CEO]

  Another argument of the distributed work is the flexibility with current employees. When one team member changes his residence to another city that does not automatically mean an end to the employment. It is no problem to stay in the team also when moving to another city because the whole way of collaboration is constructed for distributed teams. A further benefit is the possibility of working from home if necessary, there is no need to be physically present in the office every day, this can be a great benefit for team members because combined with flexible working time it allows for better taking care of personal matters like taking care of family or children. This is also a factor that Alpha openly names as a benefit for themselves as well as employees.

  > „It happens every now and then that somebody does home office work. Because of the family - we all have children - this makes the situation much more easy

> when I can say: Ok, today I work from home because I have to handle some personal matter and then I can save the driveway to the office. [...] This is simply given because everything is prepared for remote collaboration, everything that I need is a stable internet connection. And this benefit is something that we can offer our employees as well.“ [Alpha #10 - CEO]

- **Disadvantages**
  As first general disadvantage they mention the lack of communication channels that are available when been physically present in the same room. Although they have a lot of communication channels set up by online collaboration tools the lack of face to face contact is still noticeable.

  > „But anyway, I would consider this as a disadvantage, you are a little bit detached. When I am in the main office you sometimes just shout something across the room, or you notice two other people talking about a problem where you maybe can contribute. This detachment is kind of a disadvantage for me.“ [Alpha #11 - CTO]

- **Culture**
  Similar to the language the topic of cultural differences is no matter in Alpha and did not come up in the interview.

- **Configuration**
  Initially when Alpha started the team consisted of three people distributed over two locations and while they never had a doubt about the general remote working process they still where aware of the problems that configuration of the team situation can have:

  > „The challenge really was: We never have asked ourselves if we can work as distributed team but we rather asked: Is it posing a danger that there may emerge two separated blocks or that Andreas feels excluded in Linz.“ [Alpha #12 - CEO]

  Furthermore one member of a site where there usually are two people located reported the problem of feeling separated, especially in situations where a high amount of team members gathered in one other location.

  > "Then it is not like four - two - four but it is four - two - eight; and there you notice that it is forming a whole new dynamic with eight people. There is much more talking among each other, there is much more progress in a project in contrast to the rest of the year and when you sit in Linz you are not really part of this." [Alpha #13 - CTO]

  Although this fact was declined immediately from the second interviewee who stated that this is a subjective feeling and it is in reality more the matter that he is also often not up to date what is going on in each team it is anyway noteworthy and a factor that has to be kept in mind permanently.

- **Technology and Infrastructure**
  Alpha reports that the current infrastructure is one of the most problematic points in the aspect of working as a distributed team. One issue is the internet connection that is sometimes not stable enough to maintain a continuous high quality audio and video transmission which can block a lot of work when preventing team members from communicating. Beside the lack in connection quality they also report issues with the remote communication tools, ranging from bad positioning of the camera at video conferences (so other attendees see just shadows of the remote participant) or bad microphone quality.

> „But there is still a lot to do. Partially there are technical things that are barriers, like how bad video via the internet is sometimes. Also with the tools we have not found the perfect solution yet when it comes to microphones and cameras. The technical interaction is not as smooth that it can be used without thinking. This is an aspect where the infrastructure and technology is behind the way of working.“ [Alpha #14 - CEO]

Team
Every few months the team comes together to do technology exploring workday where the team mixes up and works on freely chosen topics. This should on the one hand arouse interest in new technologies and offer room for trying new things but also improve the communication in the team.

Once or sometimes also twice a year they furthermore gather the whole team and do a two day activity where they reflect on the past work, present completed projects and talk about processes and improvement possibilities. This has some characteristics of Retrospective meetings but on an other level. The second day is normally for some team event where they do some leisure activities.

Requirements
The interviewed experts did not report any special requirements for their team members but indicated that a high level of self organization is necessary as well as being communicative in the team channels.

6.2 Beta

Agile Development
Beta reported they started with a Scrum process because they already where accustomed to it from previous work experience. They work with Sprint iterations where they conduct Daily Scrum meetings, Sprint planning meetings and Retrospective meetings. The CEO, who takes the role of the Product Owner in each team, travels between different sites to be physically present when needed. The team size ranges from at least three developers to more depending on the project size.

> „We have realized that within the team everybody is forced to improve himself and to concentrate on the essential things. It is important that every employee can work independently because of the distribution over multiple locations. But luckily this was working well right from the beginning.“ [Beta #2 - CEO]

They are using Jira for tracking work packages and errors and Grasshopper for digital Agile Boards.

- **Evolution**
  The team started with Scrum since they already knew the process from former employments. Following their description they stick to a standard Scrum process with the common adaptions necessary in every team. In the beginning they had to adapt some aspects to the project management tools they use which is currently Jira and Grasshopper.
- **Iterations**
  They are working in Sprints with a normal duration of three- or sometimes if the project is requiring shorter iterations, two weeks. Their Sprints are not based on single projects but rather on the time factor so they put working packages of multiple projects into one Sprint.

Agile Practices
Beta facilitates pair programming as an essential activity to improve communication between team members as well as an important activity for transferring knowledge. Pair Programming is done between co-located individuals as well as remote, they report that they have no issues practicing it using online screen sharing tools.

> „Pair Programming is mandatory for transferring knowledge. Because we have specialists and everybody is specialized in certain areas - one is a 3D developer, one is good in data processing, another one in machine learning or in fronted development. And we would like that people share their knowledge because if somebody leaves or becomes absent - for example because of going on holidays - another one should know what he was working on.“ [Beta #3 - CEO]

They furthermore do classic code reviews occasionally but not as a standardized process.

- **Documentation**
  Documentation is done in Confluence, a software for managing documents and an element of the Atlassian tool suit.
- **Information Radiator**
  They do not have any special information radiators but use the digital Scrum board functionality of Jira.
- **Planned or Failed Practices**
  In general they are satisfied with their current situation and process although they emphasize that their process is always evolving and the pursuit of reducing and optimizing organizational costs is a constant element.

  > „Currently we are very satisfied with our process and we want to stick with it. But of course we also would like to minimize the effort necessary for organizational tasks.“ [Beta #4 - CEO]

  One thing the CEO brings up is the idea of rotating the location of members. In his opinion this would be great to improve the homogeneity within the team.

Communication
The main communication channel for the team is Skype, each team member has a company account and they emphasize the usage of this company Skype account instead of private accounts - this should prevent team members from being distracted by private communication.

> „We want colleagues to concentrate on their work, (...) we have made the experience - in our previous jobs - that it is beneficial to not use private communication accounts for the daily work. People are just too distracted by this.“ [Beta #5 - CEO]

External stakeholders also get access to the relevant information and documents in Jira and for those who are overstrained by the tools complexity they use simpler shaped tools like Trello [4]. Beside the usage of Skype as a basic, daily communication tool they also utilize it in remote meetings like the Daily Standup or for Pair Programming sessions:

[4] https://trello.com/

> „We use Skype and share the screen with the agile board. The Scrum master then moves the tickets in the name of the present team members“ [Beta #6 - CEO]

The access to internal project management information combined with their participation in the Daily Scrum - which is a free choice for them to join - is also a way to hold steady communication with external stakeholders since they always have insights on the current status and the progress.

> „This way it is also more transparent for our customers because when they take part in the daily standup meeting they see what is going on and what they are spending their money for.“ [Beta #7 - CEO]

- **Face to Face Communication**
  Regarding the regular communication they do not have concerns for not having face to face communication, one aspect they argue why the remote communication is working well for them is because of fitting team members. Although face to face communication may provide benefits they also see aspects militating in favor to being distributed:

  > „I cannot tell if it is equally efficient, maybe it would be a bit more efficient if the team is co-located. But, and that is an advantage - because of the professionalism of the people - there are no email spam and team members first think about it before they contact another team member if they really want to disturb now. That is maybe an advantage.“ [Beta #8 - CEO]

  But while the remote communication is not an issue in the daily communication, every three Sprints one project team - or sometimes multiple teams - meet in one location to on the one hand discuss issues, plan upcoming tasks and impart knowledge as well as doing some leisure activities together.

- **Language**
  All team members share the same native tongue, therefore language is not a factor that came up during the conversation in any way.

Distribution
The reason for the distributed team is the need to be physically near to their customers locations because it is for some projects necessary to be physically present at the stakeholders site. Furthermore they have one office in Vienna especially for visibility reasons and for being present and thereby attract new customers.

- **Advantages**
  One mentioned advantage is the access to a larger market due to the location in or near different cities. But with mentioning this advantage the CEO simultaneously mentioned a small downside: The time on the road that is necessary to regularly visit to each location.

  > „What I wish for would be that I do not have to drive that often. But it is not important because I have chosen this myself and therefore we have a good area coverage and demand in different cities. It is often the case that we have customers from Vienna and because of our distributed team we have the chance to get such customers because they want to have on-site service.“ [Beta #9 - CEO]

- **Disadvantages**
  Besides the increased time for traveling a further disadvantage is the less close relationship between team members.

  > „The interpersonal relationships are not as good as in teams that are co-located and are having lunch together. People in the same place are of course closer to each other which is a big advantage. This is the disadvantage for us that we do not see each other in that frequency we would like to do.“ [Beta #10 - CEO]

- **Culture**
  Cultural aspects did not come up during the Beta interview.

- **Configuration**
  There where no real indications of specific configurational issues that where addressed in the interview.

- **Technology and Infrastructure**
  One of the main problems that occur with the distributed collaboration process again is the problem with unstable internet connection.

  > „To be honest, the choke point is the internet connection. When UPC does not work we have a problem. That is the reason why we have switched to connection provided by A1 which is a bit more stable. It is not the fastest but it works everywhere. The whole problem is that if we have no internet connection or when the connection is bad. But this is primarily a technical issue.“ [Beta #11 - CEO]

Team
While the interviewee states that the mere work activity is no problem in the distributed team they perform regular encounters of a team building nature where they meet on weekends and do some leisure activity together.

> „The team meets every two months and we do some activity together, for example on a weekend. The interpersonal connection to each other is there, also in private aspects. But not like in a company that meets every day after work.“ [Beta #12 - CEO]

Requirements
The team members in beta have a high qualification and are generally experienced developers which all have a background in the field of software engineering. The interview partner of Beta explained that it is their policy to only employ senior developers because their professional experience is adjuvant in distributed teams, although such employees also connote a higher financial aspect due to higher wages compared to junior level employees.

> „Being autonomous is very important and being responsible for one's own tasks is also important. It is not easy to find employees who can live up to that entirely.“ [Beta #13 - CEO]

Beta reported that they had to let team members go because they lacked those required self organizing and responsibility skills.

> „We already made the experience with developers who could not satisfy those requirements and we had to let them go because they did not fit our team very well.“ [Beta #14 - CEO]

6.3 Gamma

Agile Development
The team is following a classical Scrum approach with very short Sprint durations. But while the internal process is Scrum but they state that there are always adaptions to make, especially when it comes to delivering products and dealing with processes that are required by external stakeholders. They reported that it is often hard to get a highly available contact person from external customers which have the full domain knowledge that is necessary therefore they normally use the own project leader for that role:

> „That is why we have a proxy Product Owner, this role is carried out by the project leader who bundles certain know how elements and serves as a first contact for the team and then again checks with the real product owner. This is breaking up the whole process a bit because some team members have more visibility to an outside customer than others but in the end we would not be able to work without having such a proxy product owner. We would have incredibly long preliminary durations before a Sprint and very long Sprint durations where we would have to adapt the scope constantly because it would take to long waiting for answers to questions that team members have.“ [Gamma #1 - Team Leader]

The Sprint planning meetings are done via video conferencing and they bring the whole team together on a regular basis. Those gatherings are used to share knowledge and perform Retrospective meetings.

- **Evolution**
  They started with a Scrum Process that they had to adapt to their specific needs.

  > „We started from the very beginning with something that was similar to Scrum. We also had phases where we deviated because of lack in discipline. This is just the challenge with agile methods I think - that they are rather strict. They provide flexibility for product development but do not really tolerate deviation from the process model itself.“ [Gamma #2 - Team Leader]

- **Iterations**
  They are currently using one week Sprints which are working very good for them.

  > „We had three weeks at first because we had to try that and we had pending infrastructure tasks which always take some time to resolve. We came to the one week duration because of a simple reason: To quickly see results and to counter the trend of delaying and rescheduling things. One week requires a lot of discipline and to define the stories very precisely and appropriately so they can be done in the given time frame.“ [Gamma #3 - Team Leader]

  They reported that with longer iterations they tended to have user stories with to many acceptance criteria; and when only a single one is not met the story has to be reopened and was often not finished during a Sprint. This resulted in stories going back to the Product Backlog again and again and the overall state that can be seen by looking at the backlog status is not very precise. Interestingly they are very aware that one week sprints are not that common but maybe this awareness is exactly the reason it works so well for them:

> „All in all we are very happy with the one week duration and we would not want to switch back. Although people returned from external trainings where they where told that one week is impossible and can never work and a Sprint has to be at least two weeks long. This seems to be a bit of a religious question.“ [Gamma #4 - Team Leader]

Agile Practices
They have a modern setup with CI and continuous deployment, use practices like Pair Programming which is done with screen sharing tools and code reviews where a specific purpose tool is used. They report that peer reviews are highly used in the team as well as Pair Programming because it has review- as well as creation aspects. Those activities are done co-located as well as distributed, this does not make a big difference since it are always the same tool that are used for the process.

- **Documentation**
  They use a wiki where everything is documented. This is not somehow formalized but rather very offhandedly handled which keeps the communication simple and quick:

  > „We quickly transfer topics into tickets or in our wiki. If somebody creates something or needs a review we just take a picture of the sketches and add them to a ticket. We do not wait until it is formally created or drawn in a specific tool but we are very casual with such things. This way we reach short cycles: Taking a picture of the whiteboard, uploading it and then calling some remote team member: 'We are in a meeting and are thinking about an optimization, could you take a look I just have taken a picture.'“ [Gamma #5 - Team Leader]

- **Information Radiator**
  They actually have an accessible monitor in one of the locations but it turned out that it is not really used.

  > „We have a screen in one team office but I think it is rarely used. Also in other teams, when I pass by there are mostly playing funny Youtube clips or bad source code highlights. But real information is rarely radiated there. Also nobody looks at it, I would also not get up and go to a screen just for getting some information.“ [Gamma #6 - Team Leader]

- **Planned or Failed Practices**
  There where no processes that explicitly failed or are planned in the future. Those things that are planned are rather aspects that try to increase standards and improve processes:

  > „Not in the area of agile methods. It more refers to the fact that process automation is never at its end, infrastructure is never good enough, build times are never short enough and code quality is never good enough. So it means it is more referring to technical things and infrastructure.“ [Gamma #7 - Team Leader]

  This can be seen in the following example where the team leader talks about trying new communication tools:

  > „I think we will start experimenting with enterprise messenger software. We will have a look if people like it and if they do we use it, if we see that such things are not used and accepted then we end it. We also act agile in such aspects, we

> try things and make first steps and also improve them or discontinue them if they are not working. I think it is not reasonable to force specific tools on a team.“ [Gamma #8 - Team Leader]

Communication
They are overall very pleased with the communication within the team and see continuous communication as important factor for keeping knowledge synchronized.

> „The assumed communication overhead of most agile methods is not that high if you take a closer look. Fifteen minutes a day per team member, people are spending more time in coffee kitchen each day. A quarter of an hour of work time for communication each day is not at all painful.“ [Gamma #9 - Team Leader]

A lot of communication is done via audio communication. Even when use video conferencing they use the visual part not for sharing a view on individuals but rather use it to display information from planning tools or documentation.

> „Primarily via telephone or personally but there are also some colleagues that favor using instant messengers. Additionally the communication is supported by screen sharing tools like Teamviewer to be able to collectively have a look at a board or at some source code. But all in all surprisingly much telephone.“ [Gamma #10 - Team Leader]

Most of the meetings like the Daily Scrum Meeting as well as Sprint plannings are done either via telephone calls or sometimes using video conferencing tools. Apart from the daily work where the communication is done via audio and process management tools they use the regular gatherings where the whole team meets in one location to discuss more complex topics.

- **Face to Face Communication**
  Due to the lesser effective knowledge sharing and knowledge transfer as discussed in the disadvantages section, they regularly bring the whole team together to exchange knowledge and discuss bigger topics.

  > „And this leads to the fact that we bring together the whole team each few weeks. Normally we switch between locations to even that out and use that time to discuss new developments, plan upcoming topics, do pair programming and generally improve knowledge transfer.“ [Gamma #11 - Team Leader]

- **Language**
  They report that the language can be especially ambiguous when communication is not face to face.

  > „A situation that is troublesome is when colleagues do not speak German. Normally this handicap is not notable but it becomes an issue when using the telephone. This is an aspect I would give more attention to in the future because it is really not working very good. It is because you understand people bad via telephone and it becomes exhausting for everyone. And when communication becomes exhausting people tend to circumvent it.“ [Gamma #12 - Team Leader]

Distribution
Two reasons where named as cause for the distribution:

> „We are around 200 people and when you suddenly need a team of six to eight people - now it are already ten overall - then it is not that easy to find suitable team members. The second aspect is that the two locations have a historical background and evolved from different core focuses. One location had more experience with web portals and therefore we started with a small share of team members from one location but extended that share in the meantime because everything is working very well.“ [Gamma #13 - Team Leader]

They told that they where themselves surprised on how good the distributed team worked and that they by now it is of no importance from which location a new team member comes from.

- **Advantages**
  The first advantage of the distribution that is mentioned is the greater pool of human resources that can be used to build a team:

  > „We can access different experts from two locations because of the different core areas they have. An expert in for example back-end development may be located in one location and if we need help we can go there, while on the other hand a security expert may be located in the other location. Overall we have access to a larger network of colleagues that way.“ [Gamma #14 - Team Leader]

  Another reason that actually is seen as an advantage of the distribution is the focus on communication and the focus on carefully choosing and reviewing communication processes and other tools that are used in the team.

- **Disadvantages**
  The first big disadvantage of the distributed team that is mentioned are the high costs for travel that are needed and the increased complexity when it comes to doing something in person on a remote site. A second mentioned downside is the lesser efficiency of communication:

  > „One disadvantage is that know how transfer in distributed teams is in principle not that effective. People often use the telephone or sharing their screen but this is not as effective as just walking up to somebody sitting two meters away and asking for help. Tools, regardless how good they are, simply pose an obstacle.“ [Gamma #15 - Team Leader]

- **Culture**
  In their team they do not have any cultural aspects that came up, but the interviewee argues that culture may become a problem when working with locations where the cultural difference is very high.

  > „All projects I know have ended with flying in the remote team members to have them work in Germany. I am very skeptical when it comes to far distributed teams and I think it maybe can work in software maintenance where there are no high professional requirements. But since an agile team should include very skilled team members with different domain knowledge and not hand over responsibility to a project manager this is getting very difficult in my opinion.“ [Gamma #16 - Team Leader]

- **Configuration**
  An important factor is to agree and then stick to a process and toolset, otherwise this causes problems especially with distributed team members:

  > „When I have processes that are passing by a certain toolset the whole system looses its validity because I can not rely on it anymore. Every time I do not see something I have to ask if that has already happened or not.“ [Gamma #17 - Team Leader]

- **Technology and Infrastructure**
  Gamma reported that for simplicity as well as reliability they choose telephones as a base communication strategy. They also use internet based communication tools but always have the telephone infrastructure as a backup option. This extensive use of telephones may also be the reason they did not report any big issues with technology or network availability.

Team
They have a lot of team building activities that are done on each site as well as between sites. Since the team members from one site regularly visit the other location and this is done in turns there is a lot of physical contact also over the distant sites.

> „In our company face to face contact is important, this is maybe because of our company culture. Playing tabletop soccer together, doing coding activities with pizza and beer, or having innovation days where people can use work time for personal projects; this kind of collaboration and exchange has a high significance in our company.“ [Gamma #18 - Team Leader]

Requirements
From the interviewee's point of view team members in distributed agile teams have to have a high qualification. They have to be self organizing and communicative.

> „They have to be more qualified, this is notable. Agile methods in my opinion require better qualified and dedicated employees. Just so it does not get lost in the noise. Because processes are meant to be done quickly and continuous improvement plays a major role.“ [Gamma #19 - Team Leader]

6.4 Delta

Agile Development
Delta uses a process model that takes parts from Scrum as well as Kanban. Each team has a team leader who is the main contact person within a team and who also coordinates the communication outside the team. Furthermore - apart from dedicated developers - they also have other roles like quality managers, and tester.

> „In the Backlog we proceed like in Kanban but the planning we are doing like in Scrum. We have Retrospectives but we do not have them fixed in the process but rather do them when we feel like we need one.“ [Delta #1 - Team Leader A]

They have several kinds of meeting activities within their process, the most frequent being the daily standup meetings each day which last for 15 minutes. Furthermore they have regular grooming sessions together with agents from the product management department where they estimate upcoming tasks. For bigger upcoming topics they also have *planning meetings* where they gather the whole team in one location to discuss the requirements and do architectural planning.

- **Evolution**
  They started with Scrum about three years ago and also had one team in the company that used Kansan. They felt that Scrum was too restricted for them so they adapted it, weakened some of its rules and took in some elements from other process models. Having dedicated roles like quality managers and testers in the team indicates that the team comes from a more classical software engineering approach.

  > „We have started with Scrum some years ago, then later on one team used Kanban. Now we have a mixture of both, we have taken the best of both and adapted to our needs.“ [Delta #2 - Team Leader B]

- **Iterations**
  Initially they started with Sprints of two weeks length but changed that to one month lasting Sprints. The two week iterations where to short for some bigger work packages they had therefore they decided to extend the duration.

Agile Practices
They sometimes use Pair Programming when there are certain problems or in situations where they feel two people should have a look at some code. When utilized it is done either by co-located team members or via screen sharing tools that does not make a difference for them. But generally it is not a fixed part but just used occasionally. They have a CI system with Jenkins but do not use strict Test Driven Development (TDD) although they are paying heed to having new implemented functionality covered with appropriate tests.

They also do code reviews occasionally via Skype where they use the screen sharing functionality to go through the code.

They reported that they use GoTo-Meeting for meetings and sometimes record the meeting sessions to preserve it and make it available for other people who might not have participated in the meeting as well. This is used for informative meetings which a lot of people should witness:

> „There are always meetings where you do not get all employees to join, but some meetings - for example from the corporate management - are mandatory for everyone because those meetings discuss the next goals and steps“ [Delta #3 - Team Leader A]

Furthermore they are experimenting with doing more documentation in video form:

> „The next idea is to do more documentation using video recordings. In the past we just wrote topics like: 'How to use the password manager' in a Confluence document, but now we plan to have a video for that because it is faster. One can view that in five minutes and gets the information that he needs.“ [Delta #4 - Team Leader A]

- **Documentation**
  Generally they use the wiki functionality of Confluence from Atlassian as knowledge base.

This is the first place to go for a team member who is searching something, if it is not found there he or she may ask the team leader who then can forward to an appropriate contact person. But they also state that not everything is inside such wikis yet and there is still a problem with having not everything in it and the general usage of emails to inform team members of new entries.

- **Information Radiator**
  They reported that they used a non digital approach in the past but changed that to just the digital version:

  > „We have our planning board in Jira and everything is managed using Jira. There you can see the progress of each team and what still has to be done. We have had sticky notes in the past but there is the problem that the remote people do not see it. Using cameras was also not a solution and therefore we decided to stop that and just use the digital version.“ [Delta #5 - Team Leader B]

- **Planned or Failed Practices**
  Apart from the to short and fixed Sprint length discussed in the *Evolution* section above there where no specific processes that where reported as failed.

Communication
They use Skype as a general communication tool for the daily communication as well as for meetings where they also sometimes use GoTo-Meeting. Furthermore they made the experience that it is necessary to set a fixed meeting schedule because without dedicated motivation there is not enough communication between the sites.

> „Experience has shown that it is necessary to set fixed schedules for meetings to synchronize each other. Without fixed schedules you have not much contact to remote offices.“ [Delta #6 - Team Leader A]

The daily standup meeting is held in Skype where the the team leader enables screen sharing and displays the agile kanban board for all attendees.

- **Face to Face Communication**
  Although they bring together a team in certain situations they don't see an absolute requirement for a face to face meeting:

  > „Actually there was no situation in the last seven years where it would have been absolutely required to have everybody in the same place to be able to start a new project. But what we do regularly is when there are a lot of people from a certain location involved in a project then we invite them for a kick-off and combine that occasion with other procedures.“ [Delta #7 - Team Leader A]

- **Language**
  They state that not sharing the same native tongue can impose problems within a team. Having non German native speakers in the team results in doing the main communication in a team in English as a common chosen language:

  > „Sometimes - although this should normally be no problem - it is English. Some colleagues are having difficulties when they have to use English instead of their native tongue.“ [Delta #8 - Team Leader A]

Furthermore the remote communication imposes an additional difficulty in understand each other:

> „Face to face it is easier to understand each other, especially when you do not share the same native tongue. I am no native speaker myself and when I talk face to face it is much easier to understand my counterpart than when I use a telephone or Skype, there it comes to misunderstandings more often.“ [Delta #9 - Team Leader B]

Distribution
The main reason for the distribution is of workforce aspects. In the main location there where insufficient employees available therefore the company expanded to Vienna as well as Slovakia.

- **Advantages**
  The main advantage as argued by the team leaders is the flexibility in creating teams because the pool of potential team members for certain tasks is bigger and experts can be chosen from different sites depending on the current need. This increases the flexibility for creating and also scaling teams.

- **Disadvantages**
  They reported that generally remote communication feels more distant and is more complex than face to face communication. During the interview which was performed in one site with a second project leader of Delta joining the conversation with Skype there where multiple connection problems and slow downs during the interview session.

  > „What is for sure a disadvantage - we are currently experiencing it now - is communication with digital technologies. Because of Skype or GoTo meeting it is already a lot simpler than in the past but still not the same as being co-located in the same room. “ [Delta #10 - Team Leader B]

  Regarding the communication in the daily standup meeting the missing communication channels where mentioned explicitly. They reported that without seeing facial expressions and gestures from the other meeting participants it is much harder to understand their statements. They tried using cameras to display the image of other meeting attendees but found that this also was no solution for them because people then had to decide whether to concentrate on the people or on the shared screens, therefore complicating the situation even further. Another aspect that came up are general problems that arise when not being co-located:

  > „What we miss are some basic things like knowing if your team mate is still at work or already at home. If you are in the same room you can just take a look at his place or ask a coworker. But if he is from Linz you do not know it, sometimes his Skype is still active and you call but get no answer. This are some basic communication issues we are currently facing.“ [Delta #11 - Team Leader A]

  They furthermore told that they tried to fix this problem by each remote team member notifying the rest of the team via email that they get the information of his or her current status. But beside the problems of additional email traffic and the issue that the remote party is not aware of the status of the main site this could not really fix the issue to date.

  A good summarizing statement about the disadvantage in their distributed experience was:

> „Distributed teams cost a lot of time, this is the biggest disadvantage. Everything is going slower and this leads to decreasing acceptance where people are viewing meetings as useless time because they cannot really understand remote attendees anyway.“ [Delta #12 - Team Leader A]

- **Culture**
  Since the team consists of members from Austria as well as Slovakia the team has some limited experience with multiple cultures and they see having multiple cultures generally as a benefit.

  > „I think being multicultural is never bad, you learn other things and get to know other educations from universities. Also it is not bad if you bring together different nationalities and bring different know-how together.“ [Delta #13 - Team Leader A]

- **Configuration**
  One situation where the configurational distance was noticeable was the situation where most of the team is sited in the same open-plan office and just a few team members are sited in a neighboring country. They reported that it is much more easy to communicate with people on the same site due just being some meters away. This results in a lot of communication done just on this site which is completely invisible for team members on different sites.

- **Technology and Infrastructure**
  Delta report various obstacles introduced by the infrastructure and although they are not traveling but sited in fixed office locations the impediments caused by infrastructure are sometimes cumbersome.

  > „But we have this over and over again: Which microphone do we use, then the wifi disconnects, then a cable is defect; this are all things that cost a lot of time. I can not just simply go and have a meeting, I need to set up my notebook, open Skype and call somebody. This requires a few extra minutes every time and when you have four meetings a day you just waste a quarter of an hour setting up a connection. “ [Delta #14 - Team Leader A]

  They tried to improve the situation by standardizing certain aspects like which adapters are necessary to connect to communication infrastructure and providing each team member with the same necessary equipment.

  > „But you can see: It is always the same problems, and the main problem - in the year 2015 - is the infrastructure.“ [Delta #15 - Team Leader A]

  They furthermore reported situations where they just used the screen sharing functionality but muted the audio and instead used telephony for audio communication to improve the voice quality and avoid bad audio connection.

Team

Both teams that were topic in the interview were distributed over two locations and at least one team member in each team was from Slovakia. They did not mention explicit team building measures that the team performs on a regular basis, but a fixed cornerstone is the half yearly company gathering which lasts for one week and is considered helpful for team building.

> „In my opinion it is very important and we try to cover that point: We have a winter and summer week where the whole company gets together. There we meet and talk to each other which also improves interpersonal relationships. That just has had a positive effect so far.“ [Delta #16 - Team Leader B]

Requirements
There were no special requirements that came up for team members in the Delta teams.

6.5 Epsilon

Agile Development
The team of Epsilon is operating in the field of software quality, therefore they often are part of external projects where they may have been called in to support in certain processes like software testing, reviewing or consulting. In those cases where they are part of a project for some other company or team they have to adapt to that customers process. In those cases where they have a scope of own processes or are free to decide for them self they have two different processes which they apply: Scrum and Kanban. This depends if their customer is depending on some third party, in that case they can not use a time boxed iterative process since they are also depending on this third party. Therefore they apply Kanban in such situations because they are more flexible and do not have fixed time schedules but can adapt to the progress of the other parties.

If the choice would be totally free the interviewee would favor Scrum as a development process. For their agile boards and as a general project management software they use Jira.

- **Evolution**
  There is no information about the early evolution of the team and used process, but overall there are regular workshops where the team recaps the process and tries to improve.

- **Iterations**
  There is no fixed iteration length since it most of the time is depending on external factors, but generally they favor typical iteration lengths around two to three weeks.

Agile Practices
They have a modern setup with automated tests which are documented as well as code reviews, coding conventions and pair programming. They do Pair Programming remote and do not see any difference when doing it via screen sharing functionality. They regularly do meetings where they transfer knowledge this is done either in situations where the team gathers in one place but it is also commonly done in a distributed setting.

> „We have occasional know-how transfer meetings which I schedule which are done co-located or remote. They gather information from different projects where every time someone does a presentation about a newsworthy topic. This is working quite well with our normal tools, given the connection is stable“ [Epsilon #1 - Department Manager]

- **Documentation**
  They use tools and setups like Sharepoint for organizing and maintain documents to ensure and enhance knowledge transfer as well as documentation.

> „We also use Sharepoint and have a dedicated page there which has to be well structured. It is necessary to implement the knowledge transfer in a way that others can use it and are able to find the information they need.“ [Epsilon #2 - Department Manager]

- **Information Radiator**
  Epsilon argues that in his opinion there is no big difference between co-located and distributed teams when it comes to information radiators, even when using analog techniques:

  > „You can just set up a camera at a Scrum board and then use sticky notes, or you can do it online. I am more a fan of the sticky notes method but online is no problem, I do not see a deviation in this topic when a team is distributed.“ [Epsilon #7 - Department Manager]

- **Planned or Failed Practices**
  They reported that they had some problems with the self determination when doing Scrum while being dependent from external factors which led them to using Kansan for such situations.

  > „Once we had the situation where we started with Scrum but overlooked that we were very dependent from external factors so we could not regulate it ourself. A Scrum team should be self-contained, if a second stakeholder is onboard you cannot regulate well. That was the reason we switched to Kanban in that situation because it was much more flexible.“ [Epsilon #3 - Department Manager]

Communication
They use a variety of tools and communication channels, for the basic communication during the day they relay a lot on a text based chat communication.

> „This is a very interesting point: It has shown that often the most basic methods are the best. Very much communication is going on - you will not believe it - not by using a camera and sharing the image of one remote location to another location, no it is by using a chat software. Simple, old school chat software. Where you just join, others respond to you and you have a constant active communication channel during the whole day.“ [Epsilon #4 - Department Manager]

The reason for this is argued with the convenience and casual way of its utilization:

> „The reason for it is that it is easy applicable along the way. Using video and audio communication is always a direct synchronous interaction. This can be obstructive or you feel watched. A text communication channel has turned out to work very well for short and quick communication.“ [Epsilon #5 - Department Manager]

Furthermore they use Skype, GoTo-Meeting and Teamviewer for audio and video communication. Regarding this tools they have specific requirements that a tool has to meet to be usable:

> „What a tool needs: A stable connection, easy to configure, if it lags and the quality is bad it is ruled out. It is important to be able to share your screen, this is an essential feature. When we have two people from different locations working on the same task

> they share their screen to see what the other one is currently doing. If this is not possible it is a problem. But nowadays tools are good enough and support this, I do not see problems in that regards.“ [Epsilon #6 - Department Manager]

Also what they have is an etiquette and set of rules for remote communication, when a meeting is scheduled it has to be determined how attendees connect to it or who moderates it. Furthermore they have a rule that nobody turns of their microphone, is making faces or is rude in some other way. The daily standup meetings are done with just audio communication without video, this has no technical reasons but is just because they feel audio is enough for such short meetings.

- **Face to Face Communication**
  Face to face communication is important in terms of team building and to build and maintain a team spirit. Furthermore it is helpful in creative processes and group activities where multiple people are actively participating.

  > „Face to face is indispensable in some situations, for example when doing workshops or reviewing a product. Furthermore to actually build a team spirit it is necessary to sometimes meet in person. Getting to know each other personally requires a certain kind of communication.“ [Epsilon #8 - Department Manager]

  Another aspect for the relevance of being co-located is the certainty or uncertainty of a situation. The more undetermined a situation is, the more helpful is discussing it face to face. When nobody on the team knows anything about a new project remote communication is not very effective as a starting point and can lead to misunderstandings.

- **Language**
  The team itself does not have any difficulty with language but the interviewee recognizes that language can pose a significant problem especially when it comes to remote communication.

  > „The most important thing when communicating is language, and if this factor is not proper enough it leads to problems. If you are transnational you have a language barrier. This negatively impacts the efficiency of a team. If you have a team meeting and you talk to each other but you do not really understand the other attendees it comes to misunderstandings. It is very important that everyone has an equal level, whatever language is chosen, but if there are differences it gets difficult and wastes a lot of potential.“ [Epsilon #9 - Department Manager]

Distribution
The main reason for the distribution is the access to qualified employees. The area of software quality and testing is a very broad field that requires a wide range of skills and knowledge. Skills are not seen as individual competences but as competence of the team. The expert of Epsilon argues that it is one of the duties of the leader to bring the team together and create and foster a team spirit. Especially in distributed teams this means more traveling to visit remote sites and keep contact with the team members there.

> „It massively depends on the leadership, and this is an interpersonal aspect, some people are suited and others are not. The best tools can not help there. You can do everything like I said and it still may fail. Having this perception, visiting a team for a short period, reading and interpreting the atmosphere and passing it on at another

location. You sometimes have to be a multiplication and that is especially difficult when it comes to distributed teams.“ [Epsilon #10 - Department Manager]

- **Advantages**
  The argued biggest advantage is that it is possible to be on-site for customers and generally being present in a bigger area. Furthermore it is easier to recruit suitable employees and find needed skills.

- **Disadvantages**
  The interviewee argues that he sees it more like a challenge than a mere disadvantage that he has to travel a lot between different sites and generally has to invest more in terms of leadership.

  > „You have to invest more as a leader than in co-located teams. I am visiting all our remote offices at least every 14 days, I am always traveling.“ [Epsilon #11 - Department Manager]

  He sees it as his task to hold the team together and to ensure information exchange between sites. It is the responsibility of the leader to maintain a team spirit and make sure team members get to know each other in the first place as well as maintain communication. The biggest problem and disadvantage of the distribution is not due to processes or team members but because of technical limitations, discussed in the *Infrastructure* section.

- **Culture**
  Culture was not a topic that came up in the course of the interview session.

- **Configuration**
  Configurational aspects were not a specific topic during the interview, but the aspect came up in some situations when it came to increased problems due to single remote team members.

- **Technology and Infrastructure**
  Overall the team is very happy with their process, used hardware and tools, but one very big issue for the division leader is the bad internet connection that is available while traveling. Since he - and also some other team members - is traveling a lot between the team locations he is spending a lot of time in public transport like trains where he has little to no internet connectivity which is a huge hindrance for productivity.

  > „There is another big impediment that plays a huge role in our distributed team: The infrastructure within Europe regarding the mobile network services. This is for me, as a leader, as a team and as a service provider a very big obstacle. We are traveling a lot by train and would like to use this travel time to work but that is not possible. This is an absolute scandal.“ [Epsilon #12 - Department Manager]

  Apart from his personal strong affection of this problem it is a general issue in his opinion that is an obstacle for all individuals which are not having one fixed workplace.

Team

In distributed teams it is necessary to keep track of the team spirit and to invest more in terms of time and money to bring the team together on a regular basis. The team leader holds three appraisal interviews with every team member per year to exchange feedback. Additionally they have half-yearly gatherings where the whole company meets.

Requirements
In his opinion the communication skills of team members are very important and highly required in a distributed team.

> „The challenge is that you need very good communicative processes. That means also people that join our team have to have good communicative skills. The need to be able to early anticipate problems and build a team spirit and connection to the team without seeing the other team members in real.“ [Epsilon #13 - Department Manager]

6.6 Zeta

Agile Development
Development teams nearly exclusively used a Scrum process that was always adapted to match the specific needs of the company. On the other hand, when it comes to not develop new software but doing service and maintenance work in a software environment the Zeta expert used Kanban without time boxed iterations to be more flexible.

> „This is also my credo, especially bigger companies tend to lump together all teams even though it does not make sense. When I think about a team that does maintenance work: You have a stable product deployed at the customer where no new development happens but it is just maintained in the sense of fixing bugs and making small adaptions. In such a situation I for sure do not use Scrum, I rather use Kanban. Because the overhead of a product development process that I assume when I use Scrum is not there in this situation. On the contrary, I create much unnecessary overhead which I do not need.“ [Zeta #1 - Department Manager]

The team sizes were always at most eight or nine people and the interviewee specifies that his ideal team size is between seven and nine team members which relates to the recommendations of Scrum literature.

In the role as coordinator for multiple distributed teams the interviewee reported the advantage of iterative working in terms of control and fast feedback which is just as important for co-located teams as for distributed teams.

> „Especially agile methods make it easier because I am faster in control. Due to the daily standup and the fact that the upcoming work is broken down to small work item pieces I am able to see progress much faster. Also seen from a control aspect, I was project leader where I was in charge for seven to eight teams, the agile world is much better because I get much more feedback. When the user stories move through the different progress states.“ [Zeta #2 - Department Manager]

They also had clear rules regarding the artifacts in their Scrum processes as well as the role allocation.

> „One team one Backlog, that is a basic rule. One Scrum master for each team, but if a team was very stable and functioning well they do not need a full time Scrum master, they organize them self. In our situation, especially in the back-end area there were teams where one very experienced Scrum master had three teams, two of them

> also with Polish team members - which were very experienced and functional. In such situations one Product Owner for three teams is enough.“ [Zeta #3 - Department Manager]

While the *one single Backlog per team* rule was universal they had multiple models for the product owner role. Depending on the importance of the projects done a team had either one own dedicated product owner or one product owner proxy for multiple teams who then reported to some other entity.

- **Evolution**
  There was no real evolution to follow since the interview partner was not speaking for one specific team but rather reported experiences from multiple teams in different situations.

- **Iterations**
  The iteration length varied always between two to four weeks.

Agile Practices
The teams often were made up from experienced senior software engineers which the interviewee used as an argument to not do much Pair Programming because of the already vast knowledge of team members. On the other hand, TDD is very important and having code covered with tests is mandatory.

Another practices that was done regularly were grooming sessions where two or three team members used audio communication and screen sharing tools to inspect the backlog and update its contained work items.

- **Documentation**
  The interviewee reported different experiences with using a wiki for documentation, in those cases where the organization of documentation was following the idea of agile self organization and was the task of the whole team it seemed to be a positive and successful endeavor whereas situations where responsibility was taken from team members it was described as futile effort.

  Other documentation like specifications of functionality was documented in text documents which were then turned to use stories. One reported drawback in such cases was the change in media from office documents to specific project management tools, which resulted in sometimes unnecessary task overhead.

- **Information Radiator**
  While not being directly involved in teams that utilized physical information radiators the interviewee reported he has witnessed such systems from time to time and generally is very fond of such applications because they can increase team ethics and motivation.

  > „Especially when it comes to things like build status, or build is broken and then a light is flashing red. I see such things from time to time but would not reduce it to just distributed teams. This things are generally very helpful also in co-located situations because it is improving the team spirit and puts a kind of pressure on the team because they say: 'We cannot afford having this red light flashing.' I think this is a very applicable idea in distributed teams.“ [Zeta #4 - Department Manager]

- **Planned or Failed Practices**
  Depending on the independence of team members and how well teams were doing in the aspect of self organization, some situations required team leaders who set goals and were responsible for making decisions.

Communication
Teams used various communication channels which were chosen depending on the situation, but the biggest portion of communication was done with Skype or Lync.

> „We had multiple channels, Daily Standup meetings were done with Lync, the communicator software from Microsoft. For bigger coordination meetings like Sprint planning, Reviews or Retrospective we had dedicated video conferencing rooms and if some technical issues occurred there was always the telephone left. And for some special occasions there is still always the possibility of travel.“ [Zeta #6 - Department Manager]

Meetings like the Daily Scrum was done with video conferencing software where one team member in each location put his notebook on the desk and all other team members gathered in front of it. The interviewee generally favors video conferencing with the possibility to see the other team members:

> „Seeing is always better than hearing. Because if you hear something you don't get as much information as when also seeing it. If I see my counterpart, his posture, his mood, this are things nobody can hide. When I just communicate using audio then this is very different.“ [Zeta #7 - Department Manager]

He also argues this usage of visual information to be better able to get information about the status of remote teams because as a team leader he gets more feedback from other team members.

> „That is the reason why I insist on seeing my counterparts in the Daily Standup meetings and not just use audio. Because that is a channel where I can get further information from.“ [Zeta #8 - Department Manager]

- **Face to Face Communication**
  Face to face communication is especially important to acquaint team members with each other.

  > „I think it is very important that people get to know each other in person, that they really shake hands. This is something essential in my opinion especially if a collaboration should work smooth.“ [Zeta #9 - Department Manager]

  He reports that this has often been done, no matter where the team was distributed.

  > „When we start a collaboration I try to get the remote people to Vienna for some time. It does not matter if the remote team is from Poland or Slovakia or India. I always tried getting the core team to Vienna to train it here so they can afterwards pass on their gained knowledge when they are back. Those were big handover sessions we had.“ [Zeta #10 - Department Manager]

Furthermore bigger meetings that are scheduled at the start and end of development iterations were held face to face and those situations were furthermore used to build and maintain a team spirit.

> „Once a month we flew in the remote team from Poland. Then we did: Sprint Review, Sprint Retrospective, the next day then Sprint planning, the following day Sprint planning two. This way we have spread those meetings over two or three days and planned ahead for the next Sprints. Simultaneously we used the evenings for some social events like playing board games or eating pizza together. This has been working very well for us.“ [Zeta #11 - Department Manager]

- **Language**
  Though the expert had experience with multiple distributed teams the aspect of different native tongues did not come up in any situation during the interview.

Distribution
The interviewee names multiple aspects that can lead to distributed teams. The first one is cost pressure and the ambition to lower the costs of software development. This is true for offshoring projects that are done with India but also for adding team members from neighboring countries where the wage would be not as high as the original country.

- **Advantages**
  Advantages mentioned were the cheaper labor costs in some countries and generally being more flexible when it comes to put a team together.
- **Disadvantages**
  The interviewee argues that the number of locations is a very important aspect of distributed teams, and that more than two locations are increasingly problematic.

  > „When I have a part of the team in Vienna and the other part in Poland it is okay. But if I distribute a team over more than two locations then the communication overhead is immense, higher than the benefits of the distribution. This is something somebody would have to show me the calculations of the savings before I believe it.“ [Zeta #5 - Department Manager]

- **Culture**
  In the opinion of the interviewed expert of Zeta offshoring is not a good option because of the high cultural distance that is often present when working with teams distributed over multiple continents and which is causing serious obstacles. He argues that there is a very high human resource fluctuation and it is very hard to establish a stable team because team members tend to come and go very frequently. On the other hand he says that he has made some good experiences with teams distributed in neighboring countries if those are culturally similar.

  > „On the one hand there are models of collaboration in distributed teams which are working. Especially when I think about countries like Poland or Slovakia which are very similar to us in their cultural aspects. Those collaborations always have worked best for us because the culture and the main parts of the collaboration were very similar. Agile methods argue to be cross-functional but to have that aspect going well you always have to consider cultural aspects.“ [Zeta #12 - Department Manager]

The interviewee reported his experience with a lot of different situations of distribution, but in general reported that that the further countries were from each other in his opinion the more difficult the collaboration got.

- **Configuration**
  Although the interviewed expert reported multiple team situations the topic of configurational problems did not come during the interview session.

- **Technology and Infrastructure**
  Technology and infrastructure did not pose serious issues for the distributed teams.

> „The technical infrastructure is nowadays very mature and there are no technical obstacle for me. Video conferencing is cheap, everybody has Skype and especially within Europa, a flight to Poland or to Germany is very cheap, that is no problem.“ [Zeta #13 - Department Manager]

Team
One aspect brought up is the appreciation for team members in other locations and especially in other countries.

> „If I appreciate the remote team, maybe travel there and look at their working conditions and go drink a beer together then it is completely different than just sitting here and it also improves the work the remote team does.“ [Zeta #14 - Department Manager]

Requirements
There is no special requirement in terms of technical knowledge but being communicative and openly approaching other individuals is an important trait for members in a distributed team. Another mentioned requirement is the willingness to travel which is important when bringing a team together for some days in one location.

> „Communication is an essential part, introverts have it even harder in distributed teams. I think that it is necessary to have at least one or two communicative individuals on each site, if you just have introverts in one location it is very difficult. Technical knowledge is not important, a good developer is a good developer. What is important is the communication and I think it is important to look out for communicative individuals when building a distributed team. And also people that do not have a problem with traveling once in a while.“ [Zeta #15 - Department Manager]

6.7 Eta

Agile Development
Nearly all teams at Eta use Scrum, Kanban is sometimes used for situations where there is no new development but maintenance work to do. As a reason why they use Scrum they state that more important than the specific method is the general idea of Agile methodologies. Team size is between at least four people to rarely more than seven, if there are projects that require more resources they scale the number of teams working on that project.

Teams are following the proposed Scrum process quite closely, they have the team consisting of software developers, a Scrum master and a Product Owner who should be provided by the external customer if possible. The fact that the Product owner is also a member of the team but is mostly located in a separate location introduces another level of distribution within the team. This means it is rather common for a project team to be distributed over three different locations: The location of the external Product Owner and up to two locations of the team members of Eta. They use software and tools for which are dedicated for agile methods, namely they use Jira and Kunagi.

The interview partner has more than 20 years of experience in more traditional software engineering methodologies like the waterfall process. Compared to such practices he values the idea of frequent communication in agile methods and argues that frequent interactions and short cycles of development are enhancing the productivity and quality of software development in general just as well as in distributed teams.

> „Using agile methods in distributed situations is still a major advantage over classical approaches. I also have done a lot of successful waterfall projects, but issues emerge much faster when using agile methods, also if you are in a distributed team.“ [Eta #1- Scrum Master]

- **Evolution**
  Teams are using agile methods for more than six years now although their process model was not called Scrum but was rather an own developed and grown process.
- **Iterations**
  They have iterations between two to three weeks, but especially in distributed teams they tend to have the longer duration for iterations:

  > „We mainly apply three week Sprints, sometimes also two week Sprints. In my experience two weeks is sometimes a financial obstacle because team members have to travel a lot if you are distributed over three locations. And also if you apply two weeks iterations you need six hours for starting a new Sprint, this means one day is quickly over. Therefore we have made good experiences with a three weeks duration.“ [Eta #2- Scrum Master]

Agile Practices
Development at Eta is setting a high value on CI as well as getting prompt feedback in case anything is wrong with a compiled build. They also value testing but do not strictly follow test driven development paradigm.

Furthermore they emphasize Pair Programming as a welcome aspect for transferring knowledge as well as sharing responsibilities within the team.

> „On the one hand our goal is to have experts in their fields within our team but we do not want to have one single expert for one topic. Instead we would like the team to be able to do multiple things so we do not have bottlenecks. This is the reason why Pair Programming is a very important aspect, but it lies within the responsibility of the team to use it when they see fit.“ [Eta #3- Scrum Master]

In situations where familiar team members use Pair Programming it is no problem with remote communication, but in case of new employees that should be familiarized with the teams processes they prefer face to face contact. As with Pair Programming, code reviews are also a practice that lies in the responsibility of the team and is applied when individuals think it is beneficial.

- **Documentation**
  If documentation is necessary it has to be part of user stories. That always depends on the the project and the needs of the external customer. If documentation is a dedicated point of the contract or if just the regular amount is required.

  > „Our approach is simple: If documentation is important it has to be part of the User Story.“ [Eta #4- Scrum Master]

- **Information Radiator**
  They do not have explicit screens as information radiators rather they are using the functionality of their tools for such tasks. The prompt feedback that is provided by their CI systems in form of emails and other electronic notifications is their form of information radiator that is working well for their distributed teams.

- **Planned or Failed Practices**
  Eta did not really mention failed or special planned practices, they rather improve their processes constantly.

Communication
They use written communication in the form of team chats for the base communication within a team. Shorter meetings are done with just audio communication where team members stay at their working place while bigger meetings are done with dedicated communication hard- and software.

Apart from Skype as a basis communication tool they use different remote conferencing solutions like WebEx. They also have specific hardware in the form of video cameras to enhance the quality of meetings. Daily Standup meetings are the smallest and shortest meetings and are done using audio communication without video conferencing.

> „It has shown that for the 15 minutes Standup meeting everybody uses his screen to have a look at the agile board and we just do audio conferencing where everyone stays at his work place. There are a few teams that do video conferencing for it, but the majority just uses audio.“ [Eta #5- Scrum Master]

Furthermore they have regular bigger grooming meetings. The difference here is that using video conferencing equipment means that team members have to leave their current work place which is considered not necessary for the short daily standup meetings.

> „Groomings are those meetings where we re-evaluate and prepare the next Sprint. We are doing them with video conferencing if possible, we have a very good video conferencing setup on each of our remote locations where audio and video quality is excellent and it is possible to share your screen. In most cases also our customers have that at their sites so they can join as well. This means, even when distributed over three locations, video conferencing is working very well.“ [Eta #6- Scrum Master]

Transition from one Sprint to the next is preferably done face to face but also sometimes done with remote communication if it is not possible to bring the whole team together:

> „Retrospectives are an important part of our process and we take them seriously because they are used to improve our process. We sometimes do them remotely but especially at the Retrospectives it is beneficial to be co-located and communicate face to face.“ [Eta #7- Scrum Master]

- **Face to Face Communication**
  They value face to face communication very much and try to bring the team together on a regular basis.

  > „When it comes to transitioning to a new Sprint, this is normally taking us a day, we try to bring together the team in one single location. We always choose a different location for this, one time at the site of the customer, the next time in our office and so on. It is important to do this face to face if possible, but it is of course not always achievable.“ [Eta #8- Scrum Master]

  The argument is that in those situations where everything is going well within a project the need for a good team spirit is not that high, but when problems arise it is important to have a good team spirit and responsibility which is strongly improved by regular face to face communication. Also in more complex meetings like the already mentioned Retrospective they prefer doing it in a co-located situation.

- **Language**
  Language is not an issue in the current teams since they are distributed in Austria and Germany and all team members are speaking the language fluently. But the interviewee also reports his experience with teams that are distributed over multiple distant countries or even continents where language may become a quite relevant factor:

  > „This is an issue of project communication, how clear are things communicated or is the communication faulty because of insufficient language knowledge, this is a thing that is happening from time to time! When an extra language like English is used as communication language then it is working quite well as long as every team member has a good level but you never can assure this.“ [Eta #9- Scrum Master]

Distribution
The reason for the multiple office sites is because of the know-how distribution as well as the location of external customers.

- **Advantages**
  There are no definite advantages that would favor distributed teams over co-located teams, but they see the distributed work as the second best option given the situation with multiple office sites.

- **Disadvantages**
  The biggest disadvantage that came up is that it is more difficult to form a team and develop a team spirit when dealing with remote team members:

  > „Scrum is relying strongly on the team spirit and the team formation process. When you create new teams and they are distributed this takes much longer to form a functioning team. Another issue is the ability to solve problems which is harder to do in distributed teams.“ [Eta #10- Scrum Master]

- **Culture**
  Cultural aspects did not come up in the interview.

- **Configuration**
  Configurational aspects did not come up during the interview.

- **Technology and Infrastructure**
  The interview partner regards good and functioning infrastructure as absolute necessity and premise for distributed teams to work. Furthermore he underlines the importance of having multiple communication possibilities ready so a team can choose the right tools and channels which work best for them.

  > „If the Skype quality is bad in a daily Standup meeting and you are sitting in front of your scream and cannot understand two or three other team members you drift away and are not able to follow the topic. That is the reason why infrastructure is so important and is an aspect where you must not skimp. Also provide the team multiple communication channels they can choose which suits them best.“ [Eta #11- Scrum Master]

While the Eta expert did not report any serious problem with connectivity and infrastructure within the different office locations he pointed out that the infrastructure during traveling is very bad. He stated that when he travels he has given up trying to join remote meetings because the connection is not stable enough.

Team
The team spirit is a very essential aspects for teams in Eta, therefore they try to bring the team together on a regular basis.

Requirements
There are no special technical requirements for team members but the soft skills and personality of candidates is very important:

> „What is very important for us is to find colleagues which are willing to work in a team, to share knowledge and do for example Pair Programming. Also people which are willing to address issues in Retrospective and are also willing to endure Criticism and Feedback. But this is a general thing and nothing we say is especially important just because of the distribution.“ [Eta #12- Scrum Master]

6.8 Theta

Agile Development
The teams in Theta use Scrum as process models which they adapted to their needs. While using and implementing some strategies from Kanban, the interviewed expert generally favors Scrum over a pure Kanban as a process model:

> „With Kanban you need to synchronize more often, when you use Scrum you can say: ’Ok, we all meet face to face every two weeks and discuss those topics’. Otherwise you have the daily synchronizations but you have to schedule a dedicated session every time some issue comes up.“ [Theta #1 - Agile Coach]

Teams meet regularly in one physical location, this is done especially at the beginning of each Sprint, where the Sprint planning as well as other meetings that they prefer doing face to face are then done together.

> „We have our Sprint planning every second Monday where the remote team members travel to Vienna. This is the easiest way, in the past when the teams were more balanced we alternated the location but now it is always Vienna. How long they stay depends if there are other things apart from the Sprintplanning that have to be done like for example complex architecture discussions.“ [Theta #11 - Agile Coach]

- **Evolution**
  The expert of Theta started with with Scrum and reported a continuous improvement and adaptation instead of complete process changes.

  > „When I joined the company they already had iterative cycles, of course there were a lot of things that could be improved, but a lot was already predetermined. We do not have a Scrum out of the book, I think nobody has that, but it is also no requirement for me. We focus on being agile, we use practices from Scrum as well as Kanban and we also pursue automated roll-out processes and continuous delivery systems. Generally we are very happy with our Scrum process.“ [Theta #2 - Agile Coach]

- **Iterations**
  Their Sprints are lasting for two weeks.

Agile Practices

Doing code reviews is a major aspect at Theta and is most of the time done asynchronously. They report that co-located pair reviews where two people sit together on one work station and go through the code jointly has not been used by the teams, instead most of the code reviews is done via dedicated software. They use pull requests to review every new addition to a code base as well as more dedicated reviews were existing parts or components of a software are reviewed.

> „If somebody finishes a piece of source code he or she creates a code review with for example Fisheye, a tool from Atlassian. Or you make a pull request and mention other team members who are then notified and review your code. They can comment on that code or just simply call you via audio call.“ [Theta #3 - Agile Coach]

Pair programming is also done whenever needed but is not a regulated process. They have no problem doing that remote and are using screen sharing functionality of their regular communication tools for it. A lot of knowledge transfer, especially between different, sites is realized by the use of code reviews and including pull requests into their workflow.

> „After some time we intentionally established that code reviews are done from different locations. Especially when we had colleagues from Russia, which was a difficult challenge, this has worked very well.“ [Theta #4 - Agile Coach]

When reviewing bigger parts of source code they added a policy to share knowledge about that code with other teams as well:

> „We have established that every time some code has changed some team member from an other team has to join the code review process. So they at least passively get informed about changes.“ [Theta #5 - Agile Coach]

- **Documentation**
  There is no real difference in documentation between a co-located and a distributed team. Generally they do not create extensive documentation that can be read by somebody outside the project who has no context to what the team is doing, documentation is rather done one small summaries and notes that sum up made decisions and are often part of user stories. Creating notes for discussed topics and decisions is an aspect that sometimes has to be enforced in both, co-located as well as distributed teams.

- **Information Radiator**
  They do not have one clear form of information radiator and also no policy regarding this. Instead each team has its own adaption and while there are some that use classical whiteboards with sticky notes, most tend to use digital representations:

  > „For the dynamic things we simply use Jira. We pay attention to transfer all tasks to it because otherwise you would have extra costs of additional synchronization. And every team has a team chat, that is maybe also a form of information radiator.“ [Theta #6 - Agile Coach]

- **Planned or Failed Practices**
  They generally had no problems applying various agile principles but report that some alterations that are needed sometimes can feel awkward.

  > „We have mastered most of the best practices of Scrum in a way they are working for us, even when we for some situations say that has to be co-located. The only thing that is maybe a bit strange is that we do estimations using a planning poker, this is a bit strange with remote members who just type in their estimates. But all in all that is not big issue.“ [Theta #7 - Agile Coach]

Communication
As a general communication channel teams use standard tool which support text as well as video and audio communication. They use Google Hangout for the daily standup meetings and Skype for one on one communication between team members. Since the major communication is done in the different text communication channels of the tools they also have set some rules for each team member:

> „We have a gentleman's agreement that everybody checks Skype at least once every half an hour so it does not come to constant interruptions but also so that nothing gets lost.“ [Theta #8 - Agile Coach]

They report that their intensive usage of the basic text communication channels creates a virtual feeling of being near to each other:

> „People are sitting side by side, although just virtual. But if you are actively collaborating, if you for example would now work with a team mate from Budapest, there is always a chat channel open or an audio call. We really use this very extensively, the team members are chatting the whole time.“ [Theta #9 - Agile Coach]

They are doing daily standup meetings where they have dedicated meeting room with a big television screen and microphones where the team members from the same location gather. Those daily standup meetings are often rather short and sometimes over in two minutes because a lot of

synchronization and problem solving is done during the regular work time and the daily meetings are really just used to update other team members of the current status. While generally they are very happy with their meetings the interviewee also reports that the issue of side conversations is a factor that must not be underestimated, especially in meetings which are not straight forward and short like the daily standups:

> „It happens over and over again that people tend to start parallel discussions in a meeting. Normally this would not bother much but when you have remote people in the meeting you loose them very quickly. This is very interesting to find the right balance between not cutting out remote attendees and on the other hand deliberately cut of an often necessary discussion.“ [Theta #10 - Agile Coach]

He argues that this is on the one hand regulated by itself because the hierarchy within the team and also the whole company is very flat and team members can speak very openly about potential problems they have. But nevertheless it is in some situations necessary to intervene and keep an eye on the course of a conversation.

- **Face to Face Communication**
  The Theta expert feels that more complex topics and problems can be best solved in face to face situations:

  > „My preference, and this is something I often hear from other colleagues as well, is that I prefer discussing complex topics face to face. Maybe there are good tools for that nowadays but we have not found them yet, or better said we have not looked for them. I also often hear other colleagues say: 'Let us discuss this in two days when you are on-site'. Furthermore a lot of team members are generally one or two days a week on-site on a remote location.“ [Theta #12 - Agile Coach]

  The interviewed expert gives an explicit example of why he favors doing complex topics face to face:

  > „In the past I tried doing planning sessions remotely, but the costs were not justifiable. You just have so much overhead in communication, so much misunderstandings and callback inquiries that the efficiency cannot be compared to being co-located. Just such simple tasks like going through the backlog can take up three times as much time as when doing it face to face.“ [Theta #13 - Agile Coach]

  This is not only mentioned when it comes to the normal distribution of multiple offices but also for situations where single team members work from home from time to time. They mention that while it is perfectly fine to do home-office once in a while they often postpone more complex discussions to later when they see each other in person again.

- **Language**
  Language poses a very important factor in distributed teams because using remote communication and the lack of communication channels they bring with them intensify problems that having different languages can bring. Especially when there are different levels of language skills within a team:

  > „It is very important that people have a similar language skill level in English. If you have a colleague with whom you normally can communicate very well but

> switch then to English this can all change. If the other person can not express himself well enough this leads to an unbalanced situation. This is not as bad in a co-located situation because you see your counterpart and can wait until he formulated his sentence. But if you are just using audio you might not see his face and might not notice that he is still trying to formulate a sentence.“ [Theta #14 - Agile Coach]

Distribution
The first mentioned reason for the distribution is the necessity to find the right people having needed skills. A second argument was that in some situations there was no other choice for setting up the team with distributed members when knowledge or skills needed are not available in one location.

- **Advantages**
  Cultural aspects are the first thing that are mentioned when it comes to advantages and disadvantages in distributed teams. Theta reports that it is strongly depending on the culture how well a team is building a team spirit.

  Having multiple different cultures within a team can on the one hand be an advantage and bring new points of view and opinions to a project but also may require more discipline and steady communication.

- **Disadvantages**
  The biggest disadvantage of distributed teams is that building a team spirit becomes a lot more challenging. Without interpersonal relationships between team members there is less trust within a team which can lead to problems in certain situations.

  > „The general underlying problem is the communication during the daily work and the interpersonal relationships. I would not go as far and say it is a trust issue because if you work with a team for years you have built up some trust but it still is a bit different.“ [Theta #15 - Agile Coach]

- **Culture**
  The interviewee reports that in his experience culture is a very important factor for the success of distributed teams. He also explicitly points out that different cultures can have an impact when just being distributed over neighboring countries. This effect is not as strong if staying within certain boundaries like for example within Europe but still can have an impact.

- **Configuration**
  Depending on the configuration of the team the interviewee reported some problems they had in some meetings where a few individuals joined via remote video conferencing and the majority of the team was present in one physical location. This was specifically mentioned when doing remote Retrospective meetings where some remote attendees were rather reluctant while the co-located part of the team engaged in a lively discussion.

  > „If you have just one remote person and the rest of the team is co-located they tend to think that they are already completely gathered and when a discussion becomes intense the remote person may not even come to speak. If you have it equally divided where one half of the team sits in one location and the other in another then the normal situation is that one location says: ’Ok, this is our opinion, what do you guys think?’. This may be not that much of a fluid discussion but people still pay more attention.“ [Theta #16 - Agile Coach]

- **Technology and Infrastructure**
  They experienced some issues in their past but those seemed to fade over the years. Generally they report that unstable network connection of course can be a nuisance but all in all they did not report serious issues.

> „Nowadays this is not causing problems anymore. In the past it sometimes was unstable and shaky and establishing a meeting took sometimes between 10 and 15 minutes. This is the whole duration of the daily Standup meeting and therefore very irritating. But nowadays connection problems are exceptions.“ [Theta #17 - Agile Coach]

Team
The team at Theta meets on a regular basis around two times each month and apart from meetings and work related discussions this time is also used for team building. If remote team members are staying for two days they sometimes use the evening to have a drink together or organize smaller team events like table soccer contests or cooking something together.

When new teams form they also do dedicated workshops for team building and giving team members the opportunity to getting to know each other.

> „Nowadays we are also doing team building workshops where we gather the remote people in one single place and try to build up a team spirit and reach the communication goals I mentioned earlier.“ [Theta #18 - Agile Coach]

Requirements
They report no special requirements in terms of special knowledge, but again point out that being communicative and willing to openly communicate within a team is very important.

> „You have to be communicative or at least are willing to openly communicate. We already got feedback from our remote colleagues in Budapest where they said: 'Guys you have to work on your audio conferencing culture!'. But yeah, apart from being able to speak English and write software you just have to be a nice individual.“ [Theta #19 - Agile Coach]

6.9 Iota

Agile Development
The Iota expert argues that there is a general trend towards the usage of agile methods, independently from team distribution. In his opinion more classical methods are just as well suited for distributed teams as agile methods and that it comes down to the specific implementation and the team.

When being able to choose the process model in a distributed team the interviewee favors a Kanban like process which is enriched with other agile elements but not having strict iterations like the sprints in Scrum.

> „At my Kanban process there is a Kanban board and there is a simple prioritization process in parallel to the software implementation process. At the management I distinct between managing the software and managing the product, the latter is about

> informing all stakeholders about what is currently done, and how the project is evolving. Parallel to this there is as mentioned the implementation, and those things are just coupled in a sense of information exchange to always know what has to be done next. But we can release every day, we can release bug fixes every day and it is generally a more loose coupling and not bound to harder checks and reviews in the middle of development.“ [Iota #2 - Team Leader]

- **Evolution**
  The Iota expert talks about different experiences and therefore there is no clear evolution in the process models to see.
- **Iterations**
  There are no fixed time boxed iterations in the sense of Scrum Sprints. But there are several regular meetings which assess the current progress, work that has to be done as well as updating several stakeholders. Those iterations are held weekly with a smaller scope as well as every two month where the process itself is reviewed.

Agile Practices
Iota tries emphasizing and using a lot of common agile practices in distributed teams. One very important topic is to set up a good continuous integration setup which automatically performs checks and tests as well as automated build systems that simply allow creating new builds and deployments.

The Iota expert also uses two kind of retrospectives in his development process, one to assess a certain product as well as one which aims at reviewing and improving the process of working itself. The first is done on a weekly basis while the second one that is dealing with the process and is not bound to a specific project is done in longer cycles.

> „On the one hand there is a weekly product retrospective meeting and on the other hand I do bigger retrospectives for the IT about every two months in specific workshops where all developers take part. There we discuss the process as a whole, what is going well and where we feel could be problems or also things like where would people feel a refactoring would be necessary. Because if you just focus on your features then you may notice some flaws but it is very individually handled. If you reserve time for it and talk about it in a dedicated setting then you get new insights about which things are really bothersome and should be improved.“ [Iota #3 - Team Leader]

Iota is using code reviews extensively in the form of pull requests, where each new code that is added to an existing code base is reviewed by another team member. This improves the code quality and also helps with transferring knowledge between team members.

> „We always do code reviews to ensure a certain quality as well as encourage a knowledge transfer between team members. Every time somebody implements a new feature he opens a pull request in the source code repository and some other team member then has a look at it, reviews the code and maybe the author and reviewer then call each other and discuss the code. After the feedback they fix potential issues and then they merge the new code into the code base.“ [Iota #4 - Team Leader]

Another agile aspect with the use of pull requests is that code is a collectively owned thing in a project and a team member should not identify with own written code in terms of being its owner.

This also means that everyone in the team - also if it is not his or her field of expertise - is allowed to make suggestions and improvements to existing source code.

> „Also it divides the responsibility for some pieces of source code because that person that is doing the review then also merges it into the code base and thereby there is no single individual who owns the code. For met it is important that we have Code Stewardship, that means that the whole team is responsible for the whole code base.“ [Iota #5 - Team Leader]

Knowledge transfer is done a lot by the usage of pull requests where other team members get to know techniques and code of their peers. Another fixed part of the process is a defined communication channel for exchanging knowledge and presenting new ideas:

> „We have a specific tech channel in our chat system for knowledge transfer purposes. Every time someone finds something new or tries something he can share the new knowledge there with the rest of the team. Another form would be writing blog articles, doing presentations or simply report what one has learned at a conference. This are things I always allocate some time for that such knowledge transfer can happen.“ [Iota #6 - Team Leader]

- **Documentation**
  The interviewee argues that bad documentation or the lack of documentation in general is discovered more quickly and is having a more severe impact in distributed teams than in co-located situations.

  > „When you have a team co-located in the same room then there is a kind of information bubble hanging over them that is just there. If you have a distributed team you do not have something like that, this means you have to actively share and spread information. You have to explicitly watch out that everyone gets new information.“ [Iota #7 - Team Leader]

- **Information Radiator**
  The interviewee states that he has experienced both, analog information radiators like big whiteboards full of sticky notes as well as digital variants which he prefers.

  > „For me digital tools are the natural way of working because the advantages are just overwhelming. You can access them everywhere, also when you are on the way, I can search it, I can mass-edit it and I can easily regroup it. This possibilities to manipulate and re-arrange data is much more complex and versatile than physical boards. But I also like the virtual Kanban boards which look like the real ones because there is more information in the graphical representation of the single cards and their position. I can remember that better than if I just have a list of issues like in classic bug tracking software.“ [Iota #8 - Team Leader]

  He argues that that apart from the mentioned advantages in the quote they are also generally a lot more versatile: It is possible to notify involved people via email depending on their interest, send reports and also connect other tools and data from and to it, like linking to source. Furthermore there are no limitations in fitting text in one card and it generally is also better preserved.

- **Planned or Failed Practices**
  The interviewee did not mention any planned practices that he has not already tried and also did not name specific practices that failed. Instead it can rather be said he generally has made better experiences using Kanban than Scrum.

Communication
As a general communication channel the Iota expert favors text based chat communication, and reports very good results with using Slack[5], a text messaging system for teams that includes features like group communication, one on one communication, file sharing, audio calls and also a lot integration possibilities.

> „Text chats are a very non-bonding communication in terms of when is a message received and when is it read. That means you always need a more committing tool where you can set tasks. This is always some kind of task tracking software where you can create tasks, assign them to somebody, see who is responsible for it and what the progress is.“ [Iota #10 - Team Leader]

Apart from those text communication channels the Iota expert argues that audio is always necessary and also video conferencing has a lot of benefits:

> „With text communication there is a lot of additional information missing, the bandwidth is low that means you need some kind of audio communication, like Google Hangouts or something like that. Furthermore a camera, this is not stringently required but very helpful in discussions for example to see if somebody would like to speak. You can not notice that if you do not see the other participants. This is something I have seen in every company so far: A text communication system, a Task-tracker and something for audio communication. In the past this was a lot of phone calls nowadays it shifted to video conferences. Skype is often the norm.“ [Iota #11 - Team Leader]

- **Face to Face Communication**
  The Iota expert argues that face to face communication and getting to know other team members in real life is not optional. Especially because of the remote situation the probability of misunderstandings and general communication problems is higher than in co-located teams.

  > „You have to get to know your opposite, you have to understand his or her motivations and fears to be able to comprehend its actions. And this understanding is something you just can get when you regularly meet in person, not only in a professional context also outside the normal work area. Go get lunch together, have an argument or solve some problems together. This things work best if you are face to face.“ [Iota #12 - Team Leader]

- **Language**
  The Iota expert pointed out that while having multiple languages is fine as long as there is one of them that is commonly agreed for communication it can get troublesome if information has to be translated twice to be passed between two individuals:

[5] https://slack.com/

> „In my experience it is very troublesome if you have two languages you have to translate to. For example if you have to translate from German to English and then again English to Ukrainian and then those steps backwards it gets complex. Also because words may have a different meaning in every language. If I am in such a situation again i would care for that more and try to avoid translating things twice.“ [Iota #17 - Team Leader]

Distribution
The Iota expert mentions two main reasons for the formation of a distributed team that he experienced. The first reason is that a co-located team member starts working remote because of for example personal reasons like family or relocation. The second reason is the need for know-how and expertise that is not available at a certain location.

- **Advantages**
  The biggest advantage as argued by Iota is the flexibility that comes with distributed teams. Working from remote enables much more freedom to the developers which in turn then also may produce better results.

  > „I think that working remotely has more advantages than disadvantages. But it is also an aspect that introduces a whole new set of problems, it is not just that an employee is sitting in another office and everything is the same. But it is exactly this that there are happening new unexpected things which you are not really aware of in the beginning.“ [Iota #13 - Team Leader]

- **Disadvantages**
  Building a team is more difficult and communication has to be enforced more specifically. A team has to be aware and to care about remote team members, which may sometimes be accompanied by a higher effort.

- **Culture**
  Including different cultures is an aspect that can bring new ideas and views into a team:

  > „I like having team members from different cultures, from different political or economic system because they bring in new opinions. How they have grown up, what is important for them, how do they feel about things, how do they recognize text or user interfaces. Such topics require a high amount of sensibility, for example we Germans are very straight forward among ourselves, something that would totally not work in other cultures. But bringing in cultural topics can provide a new view and may be quite valuable.“ [Iota #14 - Team Leader]

  But this cultural differences are not very high or nearly not existing when a team is just distributed over just neighboring countries and is having a much higher impact with further remote locations.

- **Configuration**
  Beside the typical problems that may appear in remote meeting sessions, configuration was not a topic that came up during the interview.

- **Technology and Infrastructure**
  In terms of technology and available tools the expert is happy and thinks that the currently available tools are good enough for supporting distributed teams. He argues that the selection of tools is huge and they are constantly evolving.

Infrastructure on the other hand is a topic that again is mentioned as problematic, the major issue that comes up is again a stable and fast enough internet connection which is often simply not available.

Team

It is very important to regularly meet face to face with other team members and if somebody joins the team it is best to give that new member some time on-site with other colleagues.

> „This is for two or three weeks, depending on the complexity of the task, where a new team member is present at the location of his colleagues. This creates a certain proximity and improves knowledge about other team mates. This is also a foundation for later on to better know why he or she reacted in a certain way or why they did something a certain way.“ [Iota #15 - Team Leader]

Requirements

There are no special technical requirements for team members, but the interviewee argues that while not being of technical nature, there are certain characteristics members of a distributed team should have:

> „Not every employee is suited for it. It is important to bring a high level of self organization, that you actively look for work and are able to work without requirement constant help from others. And you need employees who do not loose sight of the long term goal, if you have to stand by all the time and look out that they do not loose their way it is not working. This are two characteristics not everybody possess.“ [Iota #16 - Team Leader]

7 Discussion

This chapter presents the results of this thesis. It begins with summarizing the data gathered and presented in Chapter 6, and then attempts to discuss the research propositions as well as answering the research questions. Furthermore it states the limitations of the research performed in this thesis.

7.1 Cross-Case Analysis

The cross-case analysis summarizes the different units of analysis presented in Chapter 6. It investigates the single cases and points out similarities and differences within the aspects that where collected through the interviews and presented in the sub chapters of Section 6.

7.1.1 Agile Development

The following quote from Iota gives a good summary of the situation how most of the interviewed experts felt about the discussed topic.

> „There is a clear trend towards agile methods and there is a trend towards more flexible work conditions and working remote. Those trends have happened over the last ten years simultaneously. In my opinion there is no reason why you should not be able to combine the one thing with the other.“ [Iota #1 - Team Leader]

Generally all teams reported that they were very satisfied with the application of agile methodologies in their distributed settings and nobody reported that he would prefer some other process model.

Nearly all investigated teams and experts report that they are using an adapted Scrum process model. Two teams (Epsilon and Zeta) reported the additional usage of Kanban elements as well as sometimes using Kanban if they felt it was better suited. Just one interviewee, the expert of Iota, reported that he generally favors Kanban over Scrum, but the approach he described still is a very iterative process that includes regular Retrospective meetings and other typical Scrum features.

Every interviewee reported the application of short daily standup meetings which all followed the standardized Scrum like suggestions of the three questions and being of short duration, as presented in Section 2.4.4.

All teams that used Scrum also execute Sprint planning meetings where they define the scopes of the upcoming Sprint and most teams were also doing Retrospective meetings where they reflected about the past iteration.

- **Evolution**
  Six out of the nine units of analysis explicitly reported that they started with some kind of agile process model, while from the other three (Epsilon, Zeta and Iota), due to the fact that those experts where talking about multiple different teams and experiences, it was not

possible to identify a specific kind of evolution. Overall there was no straight forward evolution, in all cases it was reported that the adaption was an incremental process that progressively introduces new things as well as alters and changes deployed practices.

- **Iterations**
  The typical duration of iterations or Sprints is very similar to that of co-located teams and ranges from one week (deployed by Gamma) to a maximum of four weeks (applied by Delta), where all other teams have sprints with the typical duration of two or three weeks. This means that in terms of iterations there is no notable difference between distributed and co-located teams.

7.1.2 Agile Practices

Code reviews were a topic that came up in every interview but the implementations of that practice differed in some ways. Some teams reported that they use pull requests as code review process (namely Alpha, Theta and Iota), which means that whenever new source code is added to a repository or also if existing source code is edited or refactored, that changes have to be reviewed by some other team member before they are merged into the code base. Those teams that used that review variation reported very good results with it and praised that method as being highly effective. The reason why this practice becomes such a good assessment is because it serves multiple purposes: not only does it improve code quality but it also has a knowledge sharing aspect because multiple team members review code and edit and improve it collaboratively. Furthermore it shifts away from a single code owner since everybody who approved a pull request also takes up some responsibility for that source code.

Pair programming was another agile practice that most of the teams (Beta, Gamma, Delta, Epsilon, Eta, Theta) reported but no team had a defined policy for it. The overall approach was that team members use it if they feel it could help but it is nowhere mandatory. No team of those applying pair programming reported any problems with performing that activity remotely, on the contrary the Beta expert also argued that pair programming is helpful because team members have a dedicated session to interact, share knowledge and communicate updates:

> „Pair Programming is mandatory for transferring knowledge. Because we have specialists and everybody is specialized in certain areas - one is a 3D developer, one is good in data processing, another one in machine learning or in fronted development. And we would like that people share their knowledge because if somebody leaves or becomes absent - for example because of going on holidays - another one should know what he was working on.“ [Beta #3 - CEO]

All teams reported that they intensively use a CI system that performs different tests and also have set up a continuous delivery system that allows to create new builds of a software easy and frequently.

- **Documentation**
  For documentation there was no major mutuality, some teams used a wiki system or note taking and organizing tools like Evernote (mentioned by Alpha). One common practice most teams applied was the usage of established process management tools like Jira or Confluence from Atlassian (those two tools where explicitly mentioned by Alpha, Beta, Delta, Epsilon, Eta and Theta) where they include a lot of documentation into created work items.

On the other hand some experts reported that in distributed teams documentation is a more important aspect than in a co-located situation since there is sometimes fewer communication going on. This aspect is hinted by Epsilon and Iota who argued that having a good documentation solution is necessary to enable knowledge transfer between locations.

- **Information Radiator**
  All teams used digital boards from their project management tools for radiating information, there was little application of dedicated analog systems like a board with sticky notes on it. Two experts (Epsilon and Iota) mentioned that they had experience with analog boards, and two (one expert of Alpha and Epsilon) also proposed they would like the idea of having a real board but overall there seemed to be no real demand by the teams. The applied tools and communication channels were sufficient enough in every team to keep the team members informed and updated on current events.

  Such information radiators are not only valuable inside a team, but also for external Stakeholder. This comes as far as that the Beta expert reported they give their customers access to the Jira boards they use. In those cases where such tools are too complex for their external stakeholders they also create a second stripped down board where customers can view the teams progress.

  > „External stakeholder get access to our Jira board. Those who are not familiar with it can also have a second board, for example via Trello which is more lightweight and which can be used by external employees and also by our Customers.“ [Beta #15 - CEO]

- **Planned or Failed Practices**
  Some teams reported that they adapted longer meetings like Sprint Planning and Retrospectives from a complete remote approach to gathering the team at least regularly in one location. Two experts, Eta and Theta, reported they co-locate the team at the beginning of every Sprint.

  > „In the past I tried doing planning sessions remotely, but the costs were not justifiable. You just have so much overhead in communication, so much misunderstandings and callback inquiries that the efficiency cannot be compared to being co-located. Just such simple tasks like going through the backlog can take up three times as much time as when doing it face to face.“ [Theta #13 - Agile Coach]

  When it comes to shorter, formal practices like the Daily Scrum meeting or Code Reviews, it appeared as if the interviewed experts teams did not really attach great importance whether a Standup meeting was done in Skype or co-located, all reported that the digital version was working just as well as the traditional, co-located approach for them.

  There were no practices mentioned that were not yet applied but planned for the future. Teams rather reported that when they came up with new ideas and suggestions they discussed and tried that in the following iterations rather than planning long ahead.

7.1.3 Communication

Every team has some tool or setup they use as a basic communication channel. Most teams use messenger tools, especially Skype (applied by Beta, Delta, Epsilon, Zeta, Eta and Theta), while

two teams used to use Skype in the past, but shifted to a more dedicated text communication tool named Slack (which is used by Alpha and Iota).

The reported mutual benefit of those tools is that they provide a constant communication channel that every team member can use at any time but simultaneously is without much obligation. This creates a constant connection between team members regardless of their physical location.

Meetings are done with common audio and video communication tools. Some teams (Zeta, Theta reported that they have special meeting rooms for team meetings with dedicated hardware like microphones, big screens or even video projectors.

There was no consent about the usage of video communication, some teams reported that they use audio as well as video channels while other teams just talk and use their computer screens to look at agile boards or other metrics from their used project management tools.

- **Face to Face Communication**
  All teams point out that face to face communication is very important, especially for two reasons.

  Building trust and interpersonal relationships between team members is one of the main goals that most of the teams try to achieve when meeting in one physical location, because during the distributed collaboration the communication mainly focuses on professional aspects and not personal matters. In co-located situations there are a lot of situations like lunch or coffee-breaks which are used to also engage in personal chats which form relationships between colleagues. This is not happening in distributed situations, therefore the situations where team meets in person are also used to catch up on those aspects.

  A second aspect is that when it comes to complex topics that have to be discussed or meetings with a lot of attendees are preferably done when all team members are in one location. The expert from Iota argues in a similar way as the Theta expert in the previous "Failed Practices" section, he stated that for the Retrospective meetings it is necessary for the team members to be together in a mutual location:

  > „Especially there it is important to have all people in one location [at the Retrospective meetings]. The discussions are taking longer, you need a whiteboard from time to time to sketch something and explain something to others. This is very difficult when doing it over remote, it does not work.“ [Iota #19 - Team Leader]

  The exact realization differs from team to team. Some teams, like Eta and Theta bring the team together every new Sprint iteration to do the Sprint planning meeting co-located. Other teams do not co-locate the team that often but also on a regular basis, for example Gamma - who apply short weekly Sprints - gather the team each few weeks to do the Sprint planning for the next Sprint as well as do some up front planning for the next upcoming iterations.

- **Language**
  Having team members that do not share the same native tongue was an aspect some teams reported to raise complexity especially when communicating remotely. This is an aspect that also can occur in teams that are just distributed over neighboring countries or also within the same country. The experts of Gamma, Delta, Theta and Iota explicitly mentioned that having different native tongues within a distributed team can cause problems and is factor that must not be underestimated. In such a case it has to be taken seriously and it is important to ensure a good team collaboration and avoid penalizing individual team members.

7.1.4 Distribution

The main reason that a team ended up as a distributed team in the first place was because of human resource aspects. It always was the case that needed know-how or resources were not available in one location and therefore the only way to solve that problem was to take in remote team members.

- **Advantages**
  One of the main advantages is also the reason for the distribution, namely having access to more employee options and being able to choose from a wider range of skilled individuals. The importance of this advantage can be seen by the consistency of the argument since it was brought up by every interviewed expert.

  Another advantage is that distributed teams also are often more flexible because having the processes and tools set up for remote work also means that team members can work without a problem from other places than the regular office. Such liberty is often well appreciated by team members, as argued explicitly by Alpha and Iota.

  A third aspect (mentioned by Zeta and Eta) was the ability to be near to customers and generally being present in multiple locations as a company and therefore increasing visibility.

- **Disadvantages**
  Due to the nature of distribution communication in general is more difficult and has to be more cared for. Team members are more detached from each other and there is a certain distance between locations. Furthermore communication is the biggest aspect that is impaired by the distance between team members and is something that has to be taken into account explicitly. Finding the right communication tools that work best for the team can be time consuming and remote communication setups like conferencing rooms and hardware (like TVs or LCD Projectors, external microphones and speakers) have to be serviced and maintained. Generally this additional effort adds complexity which can again may become error sources in the future.

  Another issue some interviewees reported are the increased travel times and costs traveling entails. Beta, Gamma, Epsilon and Zeta where directly addressing this aspect but also argued that the increased travel effort is necessary and it is not wise to cut the increased costs in time and money those travel times bring.

- **Culture**
  Due to the spatial limitation of the research subjects, there where some teams that were rather homogeneous when it comes to cultural aspects of team members.

  Nevertheless there were teams which had team members from multiple nationalities and all of them reported that having a multi cultural team can be very enriching because it can bring diversity and new insights and opinions inside a team. But it was also stated that this effect was not very high due to the cultures being very similar. Some experts also reported experiences with team members from more distant locations where they on the one hand argued that the diversity can bring valuable new insights but can also be very troublesome due to different work styles and priorities.

- **Configuration**
  Configurational distance was mentioned explicitly by Alpha and Theta but was also hinted in some other interviews. Theta stated that this imbalance can be especially problematic in meeting sessions where the majority of the team is co-located in one place and just a few meeting individuals are attending with remote tools. Unbalanced situations like this

reinforce the typical problems of remote communication, introduced in Section 3.5.5, like side conversations or feeling left out.

- **Technology and Infrastructure**
  More than half of the interviewees reported that bad infrastructure and internet availability is one of the biggest problems when working in remote teams since it has a direct negative impact on communication tools. This aspect was directly addressed by Alpha, Beta and Delta and mentioned in the context of bad connectivity while traveling between sites by Epsilon and Eta. In summary a good share of communication issues lead back to poor network connectivity which in turn then cause problems with remote communication.

7.1.5 Team

Having a communicative and trusting team is something that all interviewees reported as a basic condition for successful remote collaboration. All teams have some sort of team building strategies like bringing the whole team together on a regular basis for some sort of team events. Those teams that regularly meet in one place, for example when doing Sprint planning meetings, often use those situations to do some activities together after work. This is regarded to compensate the lack of informal communication which is often coming short in distributed communication situations.

- **Requirements**
  No team named special technical requirements for remote team members. But most of them agreed that especially in distributed agile teams it is necessary for team members to be communicative and self organizing. Initiating communication is not as easy as in co-located teams and therefore it is necessary that every team member is aware of this fact and still holds up communication.

7.2 Examination of the Research Propositions

1. *The duration of iterations in distributed agile teams is similar to the duration of iterations in co-located teams.*

   Interviewees agreed that the periodic activities like Sprint planning or Retrospective meetings are very valuable, especially in distributed teams. Such activities update every team member with the latest news and status of a project and what is going on in remote locations. Furthermore it increases the trust in remote team members when everybody sees that - although not physically present - those members are still participating and delivering valuable input to a project.

   Those short iteration cycles are also helping with the problems of distance, discussed in Section 3.4. This can be seen by the statements of several interviewed experts like Eta who argued that „issues emerge much faster when using agile methods“ [Eta #1- Scrum Master]. As discussed in the previous Section 7.1.1, all teams stayed within the classic maximal Sprint duration of 30 days (which was applied just by the Delta team). Apart from the Gamma team (which applied one week Sprints) seven out of the nine interviewed teams had Sprint durations of two or three weeks.

   Therefore the empirical data supports this proposition and its validity in a distributed setting, and it can also be noted that a lot of the investigated teams followed the suggestion to an

ideal Sprint duration of two weeks for distributed Scrum teams, as mentioned in Section 4.3.2.

2. *While for short, standardized communication situations remote communication is sufficient, face to face communication is very helpful when it comes to longer, informal meetings with multiple participants.*

   This trend can be seen in most of the interviews. The daily meetings that are very predetermined and of a short duration are done without problem using video conferencing. Some teams even reported that they just use audio communication and use their screens to look at their agile boards or other screens of their used project management software.

   But there is also a consent when it comes to informal, complex communication situations, every expert reported that they regularly bring their teams together in one physical location. This mutual time in the same location is used for more extensive discussions like architecture or design meetings and also for things like the Retrospectives or planning meetings.

3. *Informal and frequent communication aspects of agile methods improve collaboration between sites and team members.*

   All interviewed teams had some sort of constant basic remote communication channel that was used by the team as default way of getting in touch with each other. The majority (Beta, Delta, Epsilon, Zeta, Eta, Theta) named Skype as a tool they used for basic text communication, while Alpha and Iota reported they use Slack for that purpose. This text based communication is characterized as very informal and therefore convenient to use which simultaneously reduces the feeling of distance between team members.

   This is also stated by several of the interviewed experts, the Epsilon and Theta experts both formulated this aspect with nearly identical words stating that this usage of a channel creates a proximity and a feeling of sitting together, while the expert of Iota began the topic of communication with the following statement:

   > „Communication is the essential thing, software engineering is basically nothing else than we talk about something and then put that into words, and I mean software is nothing else than words and is also basically written down communication. That is a thing you just have to deal with.“ [Iota #9 - Team Leader]

   This communication channels between team locations are an essential factor to enable the self-organization (a cornerstone of agile teams, described in Section 2.3.1) because they allow team members frequent and easy interaction.

4. *Usage of modern project management software and tools is a major factor for success of distributed teams.*

   All teams used software tools designed for agile project management, the most used one being Jira from Atlassian. Those project management tools provide virtual boards and visual representations which are used by most teams to spread and share information between multiple locations. The majority of the interviewed experts reported that they used those tools to coordinate their work progress or that they do not see a disadvantage of digital tools over analog information radiators like boards with sticky notes.

   > „Regarding the tools we have available to support our mode of operation I do not see a problem. The available selection you can pick from is huge and they are still constantly enhancing. There for sure is room to still upgrade and improve but for my daily routine there are very few challenges that have not already been solved.“ [Iota #18 - Team Leader]

The factor why the tools are so important is because they pose an essential link between different locations. They function as information radiators to keep distant team members updated and are not bound to a specific location and therefore can be accessed from anywhere. This information radiator functionality is not only valuable within a team but also for external stakeholders who can get information about the project status. (This was directly mentioned by the Beta expert, as discussed in Section 7.1.2.)

The empirical data clearly shows the importance of collaboration tools. All teams invested a significant amount of time in finding the right tools for them and in optimizing how they use them.

5. *Technical faults and limitations are posing a serious issue on distributed communication.*

   Infrastructure was an issue that was pointed out in multiple situations during the interview. Alpha, Beta, Delta, Epsilon and Iota explicitly mentioned bad internet connection and infrastructure as a major downside of the distributed working due to the negative impact on communication. This resembles closely to the challenges of video conferencing mentioned in Section 3.5.5.

   Eta and Epsilon furthermore pointed out the dire situation during travel. While the clear statement form the Epsilon expert is already mentioned in the *Technology and Infrastructure* paragraph in Section 6.5, also the Eta expert told a similar point of view:

   > „In transit [between the two offices] I do not join any meeting. Maybe briefly if somebody calls me - but generally, you know that on this route the network connection fails two or three times, you can not really participate in a meeting.“ [Eta #14- Scrum Master]

6. *Beside extensive communicative skills there are no special requirements for team members in distributed teams compared to co-located teams.*

   This is an aspect that came out clearly during the interviews. There was no team that named any technical requirements that are needed to be able to be successful in a distributed team. But all stated that the ability to communicate is a vital skill every team member has to posses. This relates to the increase difficulty of communicating with team members over a distance, if an individual additionally is a very introvert character or simply someone who does not like to communicate this problem tremendously intensifies. Furthermore being self-reliant is a characteristic which importance was highlighted multiple times. Those clear statements are indicating empirical evidence for this proposition.

7.3 Answering the Research Questions

After the previous section discussed the research propositions that were defined in the course of the case study, this section again uses and summarizes those insights to finally answer the research questions of this thesis.

1. *How can agile methods be used in distributed teams (limited to a low spatial and time dispersion)?*

   When applying agile methods like Scrum or Kanban in a distributed team the overall process and the applied practices stay basically the same. The big difference - the distance between team members which prevents face to face communication and direct interaction is met by replacing this direct communication and interaction channel by different, digital channels

like text chats, audio or video calls and conferences. Interacting with remote team mates is done with the excessive use of collaboration and management tools which are also a form of information radiator to keep distant team members informed about what is going on at different locations.

When using agile methods distributed teams started just the same as co-located teams, they set up the same processes and practices. In such situations the teams also used the suggestions and recommendations from classic literature. This can be for example seen in the Sprint duration or team size, discussed in more detail in the second research question.

The result of the cross case analysis shows that all investigated teams successfully applied agile methods in their distributed teams by using that strategy. The comment from the Epsilon expert, who told the following opinion during discussing the agile manifest, summarizes the overall point of view of all experts in the case study very well:

> „In my opinion this [the constraint of co-location] has changed by now, people are noticing and hearing things because they are constantly chatting with each other, that is similar to talking. Maybe it is even better because it does not disturb you, you can inquire when you want and are not forced to listen at a certain moment. That is definitely an advantage.“ [Epsilon #14 - Department Manager]

This quote as well as the overall attitude of all the interview partners suggests that the application of agile methodology in distributed teams does not pose problems any more, on the contrary all experts expressed their opinion that agile ways of collaboration are beneficial in distributed teams because they bring the team closer together and allow quicker reactions when problems arise.

Similar to the theory part about the right organization of teams in Section 4.3.1, the investigated teams all tried to embrace the distance as a part of their team and accepting it. The approach of the interviewed experts is therefore very similar to van Solingens advice to „bring the distance within the team“. [43, min. 28]

To not just report the status of the investigated teams, the author proposes 11 recommendations for distributed teams which are derived from the empirical insights done in this thesis in the following Section 7.4.

2. *To what extent have the principles of agile methods be adopted to be applicable in such a distributed setting?*

   The first thing that can be seen from the case study units is that there is no difference in the Sprint duration length as well as in the team size. Both metrics are similar to what is proposed in the basic literature. The typical iteration length of Sprints was chosen to be two or three weeks (apart Gamma with a one week and Delta with a four week iteration length). This means that the data from the case units does not differ compared to the classical suggestion of between one to four weeks. No team had reported durations longer than four weeks, which correlates strongly with the suggested Sprint lengths discussed in Section 2.4.4.

   Apart from the Sprint duration length it furthermore can be seen from the case study units that there is no difference in the team size. The largest mentioned team size was not more than ten people while the reported average ranked at six team members. Again this metric is very similar to the originally proposed size of around seven people as mentioned in Section 2.4.6.

   Communication, self-organization and feedback are still the cornerstones in distributed agile teams and instead of cutting down on communication and interaction between remote

locations teams try to strengthen those remote bonds and overcome the distance by encouraging collaboration (for example through remote pair programming) and also find alternative communication channels (like code reviews in the form of pull-requests, open text and audio channels, virtual task boards and feedback from CI systems) to improve cohesion of the team. Also for the communication with external stakeholders teams use their project management tools and virtual boards to communicate progress to business partners.

The principles of agile development do not really have to be altered, teams rather reported that instead there are other things that have to be adapted like scheduling of meetings, extra effort to improve communication or co-locating the team every few weeks.

> „What is important if I do not have the optimum like being co-located in one big room is to try and solve the problems another way. We for example say: If we are distributed we do at least the Sprint transitions in the same location.“ [Eta #15- Scrum Master]

This regular co-location was very important to most of the teams, and therefore is a significant aspect of the low spatial distance, since the remote locations where still reachable within acceptable costs.

3. *Which challenges have to be faced utilizing agile methods and how can those issues be handled?*

   This section investigates the impact and significance of the three challenges of distribution, as discussed in Section 3.4, as well as other impediments that were identified by analyzing the empirical data.

   Coordination
   Coordination was regarded no problem when using modern project management tools. Information can be accessed independently from being present in a certain location, every team member can check the status of a project, the remaining work that is to do, planned scopes and all other things that might be of interest.

   Control
   Generally the problems of control (introduced in Section 3.4.2) are one of the main mentioned issues in distributed development, but seem to be mitigated by agile methods, an aspect that is still valid when it comes to distributed teams. The benefits of agile methods on this challenge are discussed in the following last research question.

   Communication
   Out of the three classic challenges, *communication* is argued to be the biggest hindrance in remote agile teams. Agile methods strongly rely on frequent communication and a certain amount of trust within a team. Although it is not regarded as an insuperable barrier it still requires attention and strategies to improve the situation. Poor communication negatively impacts the two previously listed aspects and is in turn threatened by the distance and lack of face to face contact. Therefore most teams tended on bringing the whole team together in one physical location on a regular basis.

   Language
   Team members who do not share the same native tongue were quite common despite the focus on low spatial distance and such a case was directly mentioned in four out of the nine interview sessions.

   Interviewees in such situations reported that language can be a critical aspect and should be considered. This is especially important in distributed teams because of

the increased usage of remote communication and the fact that face to face communication is just happening on some occasions. Remote communication lacks various channels which makes it especially hard for non native speakers to follow conversations and also to argue with colleagues. Different language levels therefore are having a bigger impact than in face to face situations. It can be seen that having multiple languages has to be accounted for, otherwise it might evolve into a communication obstacle impeding the whole team collaboration.

Awareness of remote team members

Some teams reported that creating and maintaining awareness about the status of remote team members can be challenging. Such a situation was explicitly mentioned by one of the Delta experts, who stated: „What we miss are some basic things like knowing if your team mate is still at work or already at home. If you are in the same room you can just take a look at his place or ask a coworker.“ [Delta #11 - Team Leader A]

Overcoming the distance between geographical locations is a major tasks for totally integrated agile teams and therefore also needs specific attention and arrangements on how to tackle such basic issues.

Technology and connectivity

Communication and collaboration in distributed teams directly relies on technology and internet connectivity. This is a very practical issue that a lot of teams mentioned they have faced in the past or are still having troubles with. A bad connection massively impedes direct and frequent communication which in turn has a negative impact on collaboration, coordination and control.

4. *Which benefits result from pursuing agile methods in such a distributed setting?*

The empirical data suggests that agile methods in distributed teams have a positive impact on especially the control and coordination challenges common in distributed teams. Modern communication technology as well as the possibility to meet face to face in one physical location if it should be necessary are two important cornerstones of well functioning distributed agile teams. Therefore the following benefits were identified which suggest that agile methods have a positive impact on geographically dispersed teams:

Compensate the control challenge due to short iterations and fast feedback

As already discussed in the previous research question, problems with control are a typical issue in distributed teams, but agile methodologies reduce the issue and therefore are having a positive impact in this regard. Setting up clear schedules and practices are some forms of formal control, and practices like a daily Standup meeting can bring forth a form of informal self-control when each team member daily communicates his or her progress and commits one self to certain tasks by telling plans for the near future. This means, that control aspects are very well handled within an agile team even if the team members are distributed.

The control aspect is also directly mentioned by the Zeta expert, who argued that: „agile methods make it easier because I am faster in control. Due to the daily standup and the fact that upcoming work is broken down to small work item pieces I am able to see progress much faster.“ [Zeta #2 - Department Manager]

Increase team spirit and collaboration between remote sites

Agile methods strongly rely on frequent communication which is just as necessary in distributed teams as it is in co-located situations. This leads to the fact that teams accept the distance and the high amount of communication within the team positively impacts collaboration between geographical locations.

Reveals communication and collaboration issues faster

Instead of specifying and planning work tasks for a longer period, agile methods rely on short iterations and continuous communication and feedback. Communication problems like misapprehensions can lead to much severe outcomes in distributed teams but the frequent communication and manifold practices that further encourage collaboration reveal obstacles very swiftly. This is also summed up by the Theta expert who issued the following statement:

> „The reason why I think agile methods are very helpful in distributed teams is that they generally focus on an open and short-term communication, communication in short iterations. What I have seen in distributed teams without short communication iterations is that they drift apart very easily.“ [Theta #20 - Agile Coach]

Agile practices spread information between sites

Automated tests and CI that give quick feedback about a build status are an excellent way of communicating information to multiple locations. Remote team members which where not actively participating in implementing a certain feature still may get notified automatically from the build system and are thereby constantly informed about a project's progress.

Low spatial distance allows to co-locate teams for longer, informal meetings

In contrast to global distributed teams it is comparatively easy to co-locate team if necessary. Teams regularly made use of this possibility to gather in one place for various activities. As discussed in the second research proposition in Section 7.2, teams tend to co-locate for planning or Retrospective meetings. This is an exclusive benefit of low spatial distribution, because the costs in time and money for the temporary relocation of team members is still affordable and accepted facing the benefits those regular face-to-face sessions bring.

7.4 Recommendations for Distributed Teams

The discussion of the research questions in the previous section summarized six challenges and five benefits of agile methods in distributed teams. Based on those findings the author makes the following ten proposals derived from the examined teams to further improve the effectiveness of distributed teams and get the most out of agile methods in such settings:

Provide channels and encourage communication

Teams need to have the possibility to communicate whenever they want, getting in touch with a remote colleague has to be as easy as possible. To reach such situations there is on the one hand the need for a good communication infrastructure like microphones, speakers, TV-screens or headphones as well as software tools. Furthermore it is necessary to let the team decide which communication channels they want to use. This is well summed up in a statement from the Eta expert: „That [bad quality of remote communication tools] is the reason why infrastructure is so important and is an aspect where you must not skimp. Also provide the team multiple communication channels they can choose which suits them best.“ [Eta #11- Scrum Master]

Have a plan B

When distributed teams communicate via audio or video channels there are multiple things

that can go wrong, like a failing internet connection or hardware defects. Therefore it is advisable to have a backup plan for communication channels. This could be using telephones or written communication, an alternative internet access or standby hardware.

Plan for communication

Spreading information in distributed teams does not happen as naturally as in co-located situations. There is no space where individuals meet for informal and personal communication. Also getting information through listening to other conversations in the same room (the principle of osmotic communication as mentioned in Section 2.3.2) is impeded by the distribution. Therefore it is advisable to schedule dedicated communication opportunities and meetings. This can also be done with more informal topics, Theta for example reported they once clinked glasses virtually in a video conference on some team members birthday or Alpha who planned introducing a virtual coffee break where everyone who wants can join a video conference and drink coffee together.

Use management tools to radiate information

Project management tools like issue tracker or digital Kanban boards are vital components of a distributed agile team. Those tools are functioning as an information radiator and are essential to make information accessible for remote team members. The investigated teams often used them as a central information source where team members can find information about things like requirements, documentation or planned schedules.

Use software development practices to radiate information

Various software development practices are well suited not only to produce source code but can also be used to foster communication and spread information. Pair programming for example can be used to share knowledge between different locations when remote team members work together. Also text based code reviews or the usage of pull-requests as a form of code review is a practice that as a side effect communicates information and can inform remote team members about progress.

Do Retrospectives

The physical distance between team members can often cause disconnectedness between team members and increase misunderstandings. Therefore it is especially important to do regular Retrospective meetings to evaluate the used methodology and established processes and identify eventual problems within a team as early as possible.

Adapt the process model

Scrum and other agile process models were originally built for co-located teams and build a lot on direct face-to-face communication. When using an agile process model in a distributed teams it is advisable to adapt it to the actual situation. One example would be doing practices that where originally done in a synchronous communication situation asynchronously. This could be Standup meetings that are held in a text chat where members report their status not all at the exact same time but in the course of a halfway or code reviews that are discussed via text comments.

Stick to the defined processes and practices

Especially in distributed teams it is important to stick to defined processes and practices. This does not mean that such procedures should never be changed, on the contrary, it is important to constantly adapt end evolve (as argued in the previous bullet point). It means that team members should be able to rely on established processes. When a reoccurring meeting is scheduled at a certain date, this date should be as steady as possible. Re-scheduling on short notice may be more problematic especially with remote team members since all of

them have to be noticed and it furthermore disturbs established habits and can bring uncertainty.

Find the right team members
When it comes to build a team that collaborates over distributed locations it is vital to employ the right team members. The interviews showed that there where little to none technical requirements exclusively necessary for team members, but communication skills on the other hand where mentioned quite often. For introvert characters it may be more difficult in a distributed team because communication has to be pursued more actively.

Co-locate the team together regularly
Because of the fewer times team members see each other in person, building and maintaining a team spirit is more difficult. Due to the situation that the distance between teams is not that huge it can be overcome without too high costs which leads to the situation that nearly all teams regularly gather in one place. This costs in time and money for regularly co-locating a team may still be relevant, but are just a fact that has to be accepted if a team is distributed.

Build a team spirit
Having a good team spirit is important in every software development team, but especially critical when team members do not have that much time together. Collective activities, either when being co-located or also using virtual communication channels are important to build and maintain a good team. Enabling new team members to become acquainted with normally remote colleagues as well as improving trust and build more personal connections between remote co-workers.

7.5 Comparison to Related Work

Section 1.2 already introduced related work at the beginning of this thesis, this closing section now brings up the conclusions of some related scientific work to compare the outcome of this research and it can be seen that the results of the case study are resembling closely to the findings of other researchers.

Paasivaara, Durasiewicz, and Lassenius [66] did a multiple case study on three globally distributed software development projects using Scrum. They found quite similar numbers when it comes to the iteration length and also reported that the daily Scrum meeting „was clearly the most important Scrum practice used by all case projects.“ [66, p. 197] Their results regarding challenges and benefits report a quite similar outcome as this case study did, but also contains a lot of issues that are typical for a global distribution. Aspects like different time zones and cultures where mentioned as impediments, which had no real significance in this thesis. The problems with different native tongues on the other hand is an aspect that also came up in this research.

Korkala and Abrahamsson [84] did two separate semi-industrial case studies, where the first one was distributed in the same city and the other one between Oulu and Helsinki in Finland. They focused on the communication aspect of the distributed team and discussed their findings against recommendations on distributed software development team from [85]. One very interesting finding, that also matches with the results from this case study, was the recommendation that: „teams should be able to communicate directly in order to achieve successful results. The lack of direct peer-to-peer communication can result in significant problems.“ [84]

Dorairaj, Noble, and Malik [14] also focused on communication in globally distributed software development teams where they interviewed 18 agile practitioners and concluded that „distributed

Agile teams face communication challenges caused by the time zone, lack of communication tools, language barriers, and lack of teamwork. The participants adopted several practical strategies to overcome communication challenges in distributed Agile software development by reducing time zone, leveraging communication tools and techniques, addressing language barriers, developing trusted relationships, increasing formal communication, and increasing informal communication.“ [14, p. 114] They furthermore presented several aspects that cause communication issues as well as strategies to overcome those obstacles. Apart from typical global issues like time zone differences they mention some aspects that were also concluded in this research like the importance of the right communication tools and techniques or the necessity to still have a certain amount of formal communication through several scheduled meetings.

Nuevo, Piattini, and Pino [40] proposed an agile methodology for distributed software development which is based on combining Scrum with the RUP. They concluded that Scrum is beneficial to distributed development because „these methods emphasize the most critical challenges of distributed development, such as communication and coordination. The application of RUP to distributed software development strengthens it by means of an established development process, and by ensuring that there is the documentation needed to enable project monitoring for distributed teams.“ [40, p. 73] Although the clear differences of that approach due to the addition of elements from the RUP, there are still some similarities to this thesis: Likewise to the outcome of this research they for example propose the importance of CI the daily Scrum meeting and the Retrospective meeting.

Karsten and Cannizzo [86] describe the story of a distributed team adapting agile methodologies, namely a combination of Scrum and XP. In their description they state some findings that resemble the outcome of this study, for example the usage of several practices to not only improve the software development process but also improve the communication between remote sites. Also they pointed out the importance of regularly co-locating remote team members and the necessity of continuous process improvements. Overall they concluded that „high bandwidth communication is one of the most critical aspects of software development, especially, as in this case, when the group is distributed geographically. But through commitment in adopting agile methodologies and judicious use of travel, co-location, and new technologies it is possible to create an environment where teams can survive, grow and thrive all, delivering quality software quickly.“ [86, p. 239]

7.6 Limitations of this Thesis

Internal validity is especially important for explanatory and causal studies and not so much for exploratory or descriptive studies. To ensure internal validity, Yin suggests various analytic tactics that should be done in the analytic phase of the case study. One general strategy is to rely on theoretical propositions which was done in this research. [16, p. 139] Also the analysis method of doing a cross-case synthesis where data is gathered from multiple sources, grouped and compared to each other was a technique that was used in this research. [16, p. 164]

External validity of a case study refers to the generalizability of the outcome. Since the focus on this case study units was set to a very narrow area with defined requirements (as described in Section 5.3.4) the findings of this study cannot be generalized for all software development teams. Especially insights regarding the infrastructure situation are very specific for certain areas and may vary depending on the location. It is also not reasonable to generalize cultural aspects since this is also an area that is very diverse.

Another main limitation of this thesis is that it just uses interviews as a data source. As stated in Section 5.3.3 there are various other sources of data which could be used in a case study to

increase the accuracy of the research. This limitation was forced due to the limited resources of the author and the scope of the thesis.

When it comes to the qualitative content analysis of the gathered interview material described in Section 5.4, Mayring [80] suggests to give the gathered material to another researcher who also should do the coding of the material independently. Afterwards the two researcher should compare their results and use those gained insights to improve their original coding. Again this was not really doable in the scope of this thesis since the author was conducting the research by himself without a second researcher participating.

The proposed measurements that are listed at the last part of the research question Section 7.3 are derived from the empirical data and are not checked for their validity. Those proposed aspects could be the subject of a followup study that evaluates them explicitly to check their effectiveness.

8 Conclusion

Using agile methods in distributed teams did not seem to be the way to strive for when taking a look at the original literature about Scrum or XP. But the progress in technology and the improvements in collaboration tools increased the number of software engineers working in remote teams year by year. In 2015 a survey among software developers reported that around 30 percent have experience in working part-time or full-time remote. Simultaneously more and more teams are adopting agile methods like Scrum as their way of work and collaboration.

Agile methodologies value interaction and communication over processes and tools, they propose frequent informal communication between team members to share knowledge and coordinate working. Agile process models like Scrum or XP strongly recommend to co-locate the software development team in one big office, and claim that co-location is a vital skill for agile teams. When it comes to distributed teams on the other hand there are the three main aspects that are often referred to as the challenges of distance, namely *coordination*, *control* and *communication*.

The goal of this thesis was to investigate how these two aspects can be combined. This was done by starting with an extensive literature research to build a basis for the empirical research. On top of this theoretical foundation a case study was designed and performed, where the author interviewed nine experts that represented one or multiple agile teams, which build the source of evidence for the study.

There are already several case studies investigating distributed agile teams, but most of them are looking at the topic from a global perspective. Contrary to this worldwide view which discusses teams that are spread over different time zones and continents this thesis brought focus on teams that are distributed within a low spatial distance. To accurately confine the boundaries of the empirical study, a list of criteria was defined that teams and experts had to meet in order to be subject to the case study:

- Having at least one permanent team member who is not located with the rest of the team.
- At least one office or site had to be located in Austria or Germany.
- Teams should define their process of working and development as being 'agile'.
- There was no strict criteria for the role of the interviewee, the only important thing was that he or she is or was directly involved with the teams.

Analyzing the empirical data showed that distributed teams started just the same as co-located teams when using agile methodologies, they set up the same processes and practices. Teams generally followed the suggestions and recommendations from classic literature. There were no deviations to most of the basic agile principles like team size and iteration length proposed in original literature for Scrum or XP. Furthermore there were no agile practices that were doomed to fail due to the distributed setting of a team, instead every team managed to apply all the techniques and processes they felt worth using.

As a result this thesis identified six challenges as well as five benefits of agile methods in distributed teams. Conclusively eleven recommendations derived from the analyzed data were presented which aim at improving the application of agile methods in such environments.

The data indicates that the short iterations and the increased and encouraged communication of agile methods are very effective in solving problems in the areas of coordination and control. Iteratively and constantly reviewing, reevaluating and re-prioritizing the workflow and work that has to be done increases social cohesion in distributed teams. No team reported any serious obstacles in terms of control and coordination and most of them named procedures like short Sprint cycles and frequent short meetings like daily standups, plannings and Retrospectives as reasons for this.

The importance of communication was significant, and the topic came up in various situations and forms during the research. This is not surprising since agility and its requirement for a self-organizing team builds a lot on communication and interaction within a team. Modern technology in the form of a wide variety of text, audio or video communication tools, are nowadays providing enough flexibility and convenience to create a constant stream of communication between remote team members.

An important distinction between far and near distributed teams is the fact that the teams investigated in this research all had the characteristic of being able to gather all team members in one place within reasonable costs. This is maybe one of the biggest differences in the actively applied processes: that all teams frequently bring the team together. Such situations are used for taking on more complex topics like architecture planning and also for agile practices like retrospective meetings. Furthermore the regular face to face contact is used to build and foster a team spirit and interpersonal relationships between team members. All experts reported that in their situations they see face to face contact as a very important aspect in their teams.

Cultural distance was not having a huge impact due to the limitation of near distributed teams but was still relevant in some situations. Overall having team members from multiple cultures was regarded as very valuable because of new insights and other perspectives on certain topics. But it is also argued that a too big cultural distance can lead to big differences if not handled appropriately. A similar issue are posing different native tongues within a team. While such differences in language are mostly no problem in co-located teams due to the high amount of simultaneously present communication channels, differences in language abilities are notable in remote communication. A strong impediment was named with the available infrastructure especially regarding internet connectivity. Bad network connectivity directly impacts the quality of remote communication when it comes to audio and video conferencing which in turn spoils communication.

Regarding the skills of team members there were no notable differences in technical know how that are required to work in a distributed team. Instead there was a common emphasis on communication skills and also self organization. Due to the obstacles that remote communication introduces it is necessary to have communicative team members who compensate those barriers by actively seeking communication.

In summary the data of this thesis indicates that applying agile methods in distributed teams with low spatial distance poses no problem but instead brings forth several benefits. Common agile practices like pair programming, Continuous Integration, code reviews, daily standup meetings or Retrospective meetings were all mastered successfully by the distributed teams by replacing face to face communication with a variety of digital communication channels. Those practices not only improve the software engineering process but furthermore pose additional communication channels which further strengthen the team.

Regarding the outlook in this area it seems that there is much to come. Around 30 percent of software engineers are already working at least part time remote and there is no indication that this number will decrease anywhere soon. Due to the nature of empirical qualitative research the resulting arguments and findings of the case study within the defined bounds could be evaluated in a quantitative study to gather further insights and also strengthen the findings or suggest alterations.

Bibliography

References

[1] Mike Cohn. *Succeeding with Agile: Software Development Using Scrum*. 1st. Addison-Wesley Professional, 2009. ISBN: 0-321-57936-4, 978-0-321-57936-2.

[2] Mira Kajko-Mattsson, Gayane Azizyan, and Miganoush K. Magarian. „Classes of Distributed Agile Development Problems“. In: *Agile Conference (AGILE), 2010*. 2010, pp. 51–58. DOI: 10.1109/AGILE.2010.14.

[5] Lene Pries-Heje and Jan Pries-Heje. „Why Scrum Works: A Case Study from an Agile Distributed Project in Denmark and India“. In: *Agile Conference (AGILE), 2011*. 2011, pp. 20–28. DOI: 10.1109/AGILE.2011.34.

[6] Hans-Christian Estler et al. „Agile vs. Structured Distributed Software Development: A Case Study“. In: *Global Software Engineering (ICGSE), 2012 IEEE Seventh International Conference on*. 2012, pp. 11–20. DOI: 10.1109/ICGSE.2012.22.

[7] Nick V. Flor. „Globally Distributed Software Development and Pair Programming“. In: *Commun. ACM* 49.10 (Oct. 2006), pp. 57–58. ISSN: 0001-0782. DOI: 10.1145/1164394.1164421. URL: http://doi.acm.org/10.1145/1164394.1164421.

[8] Darja Smite, Nils Brede Moe, and Pr J. Gerfalk. *Agility Across Time and Space: Implementing Agile Methods in Global Software Projects*. 1st. Springer Publishing Company, Incorporated, 2010. ISBN: 3642124410, 9783642124419.

[9] Erran Carmel and Ritu Agarwal. „Tactical Approaches for Alleviating Distance in Global Software Development“. In: *IEEE Softw.* 18.2 (Mar. 2001), pp. 22–29. ISSN: 0740-7459. DOI: 10.1109/52.914734. URL: http://dx.doi.org/10.1109/52.914734.

[10] Pär J Ågerfalk et al. „A framework for considering opportunities and threats in distributed software development“. In: *In Proceedings of the International Workshop on Distributed Software Development (Paris, Aug. 29, 2005). Austrian Computer Society*, pp. 47–61.

[11] Liz Lee-Kelley and Tim Sankey. „Global virtual teams for value creation and project success: A case study“. In: *International Journal of Project Management* 26.1 (2008). European Academy of Management (EURAM 2007) Conference, pp. 51 –62. ISSN: 0263-7863. DOI: http://dx.doi.org/10.1016/j.ijproman.2007.08.010. URL: http://www.sciencedirect.com/science/article/pii/S0263786307001305.

[12] M. Rita Thissen et al. „Communication Tools for Distributed Software Development Teams“. In: *Proceedings of the 2007 ACM SIGMIS CPR Conference on Computer Personnel Research: The Global Information Technology Workforce*. SIGMIS CPR '07. St. Louis, Missouri, USA: ACM, 2007, pp. 28–35. ISBN: 978-1-59593-641-7. DOI: 10.1145/1235000.1235007. URL: http://doi.acm.org/10.1145/1235000.1235007.

[13] Elizabeth Woodward, Steffan Surdek, and Matthew Ganis. *A Practical Guide to Distributed Scrum*. 1st. IBM Press, 2010. ISBN: 0137041136, 9780137041138.

[14] Siva Dorairaj, James Noble, and Petra Malik. „Agile Processes in Software Engineering and Extreme Programming: 12th International Conference, XP 2011, Madrid, Spain, May 10-13, 2011. Proceedings“. In: ed. by Alberto Sillitti et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. Chap. Effective Communication in Distributed Agile Software Development Teams, pp. 102–116. ISBN: 978-3-642-20677-1. DOI: 10.1007/978-3-642-20677-1_8. URL: http://dx.doi.org/10.1007/978-3-642-20677-1_8.

[15] Darja Šmite et al. „An empirically based terminology and taxonomy for global software engineering“. English. In: *Empirical Software Engineering* 19.1 (2014), pp. 105–153. ISSN: 1382-3256. DOI: 10.1007/s10664-012-9217-9. URL: http://dx.doi.org/10.1007/s10664-012-9217-9.

[16] Robert K. Yin. *Case Study Research: Design and Methods: Design and Methods*. Fifth. SAGE Publications, 2014. ISBN: 9781452242569. URL: https://books.google.at/books?id=AjV1AwAAQBAJ.

[17] Per Runeson et al. *Case Study Research in Software Engineering: Guidelines and Examples*. Wiley, 2012. ISBN: 9781118104354. URL: https://books.google.at/books?id=BU2YZwEACAAJ.

[18] James Shore and Shane Warden. *The Art of Agile Development*. First. O'Reilly, 2007. ISBN: 9780596527679.

[19] Ken Schwaber and Mike Beedle. *Agile Software Development with Scrum*. 1st. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001. ISBN: 0130676349.

[20] Laurie Williams and Alistair Cockburn. „Agile Software Development: It's about feedback and change“. In: *IEEE Computer* 36.6 (2003), pp. 39–43.

[21] Torgeir Dingsøyr, Tore Dybå, and Nils Brede Moe. *Agile Software Development: Current Research and Future Directions*. 1st. Springer Publishing Company, Incorporated, 2010. ISBN: 3642125743, 9783642125744.

[22] Sanjiv Augustine. *Managing Agile Projects*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2005. ISBN: 0131240714.

[24] Jim Highsmith. *Agile Project Management: Creating Innovative Products*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 2004. ISBN: 0321219775.

[25] Andrew Stellman and Jennifer Greene. *Learning Agile: Understanding Scrum, XP, Lean, and Kanban*. Oreilly & Associates Incorporated, 2014. ISBN: 9781449331924.

[26] Alan Moran. *Managing Agile: Strategy, Implementation, Organisation and People*. Springer International Publishing, 2015. ISBN: 9783319162614.

[27] Kenneth S. Rubin. *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. 1st. Addison-Wesley Professional, 2012. ISBN: 0137043295, 9780137043293.

[28] Tsun Chow and Dac-Buu Cao. „A Survey Study of Critical Success Factors in Agile Software Projects“. In: *J. Syst. Softw.* 81.6 (June 2008), pp. 961–971. ISSN: 0164-1212. DOI: 10.1016/j.jss.2007.08.020. URL: http://dx.doi.org/10.1016/j.jss.2007.08.020.

[29] James O. Coplien and Neil B. Harrison. *Organizational Patterns of Agile Software Development*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2004. ISBN: 0131467409.

[30] Kent Beck and Cynthia Andres. *Extreme Programming Explained: Embrace Change (2Nd Edition)*. Addison-Wesley Professional, 2004. ISBN: 0321278658.

[31] Alistair Cockburn. *Agile Software Development: The Cooperative Game (2Nd Edition) (Agile Software Development Series)*. Addison-Wesley Professional, 2006. ISBN: 0321482751.

[32] Alistair Cockburn. *Crystal Clear a Human-powered Methodology for Small Teams*. First. Addison-Wesley Professional, 2004. ISBN: 0201699478.

[33] Stephanie Teasley et al. „Rapid software development through team collocation“. In: *Software Engineering, IEEE Transactions on* 28.7 (2002), pp. 671–683. ISSN: 0098-5589. DOI: 10.1109/TSE.2002.1019481.

[35] Muhammad O. Ahmad, Jouni Markkula, and Markku Oivo. „Kanban in software development: A systematic literature review“. In: *Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on*. 2013, pp. 9–16. DOI: 10.1109/SEAA.2013.28.

[36] Marcus Hammarberg and Joakim Sunden. *Kanban in Action*. 1st. Greenwich, CT, USA: Manning Publications Co., 2014. ISBN: 1617291056, 9781617291050.

[37] Henning Wolf, Stefan Roock, and Martin Lippert. „eXtreme Programming - eine Einführung mit Empfehlungen und Erfahrungen aus der Praxis (2. Aufl.)“ In: dpunkt.verlag, 2005. ISBN: 3-89864-339-5.

[38] George Ellis. „Chapter 8 - Agile Project Management: Scrum, eXtreme Programming, and Scrumban“. In: *Project Management in Product Development*. Ed. by George Ellis. Boston: Butterworth-Heinemann, 2016, pp. 223 –260. ISBN: 978-0-12-802322-8. DOI: http://dx.doi.org/10.1016/B978-0-12-802322-8.00008-5. URL: http://www.sciencedirect.com/science/article/pii/B9780128023228000085.

[39] Ajay Reddy. *The Scrumban [R]Evolution: Getting the Most Out of Agile, Scrum, and Lean Kanban*. 1st. Addison-Wesley Professional, 2015. ISBN: 013408621X, 9780134086217.

[40] Eva del Nuevo, Mario Piattini, and Francisco J. Pino. „Scrum-based Methodology for Distributed Software Development“. In: *2011 IEEE Sixth International Conference on Global Software Engineering*. 2011, pp. 66–74. DOI: 10.1109/ICGSE.2011.23.

[41] Michael B. O'leary and Jonathon N. Cummings. *The Spatial, Temporal, and Configurational Characteristics of Geographic Dispersion in Teams*. 2007. URL: http://ssrn.com/abstract=1739905.

[42] Christof Ebert. *Global Software and IT: A Guide to Distributed Development, Projects, and Outsourcing*. 1st. Wiley-IEEE Computer Society Pr, 2011. ISBN: 047063619X, 9780470636190.

[43] Rini van Solingen and Volker Mosthaf. *Episode 181: Distributed Scrum with Rini van Solingen*. Dec. 2011. URL: http://www.se-radio.net/2011/12/episode-181-distributed-scrum-with-rini-van-solingen/.

[44] Rosalie J. Ocker et al. „Leadership Dynamics in Partially Distributed Teams: an Exploratory Study of the Effects of Configuration and Distance“. English. In: *Group Decision and Negotiation* 20.3 (2011), pp. 273–292. ISSN: 0926-2644. DOI: 10.1007/s10726-009-9180-z. URL: http://dx.doi.org/10.1007/s10726-009-9180-z.

[45] Thomas J. Allen and Gunter Henn. *The Organization and Architecture of Innovation*. Taylor & Francis, 2007. ISBN: 0-7506-8236-1, 978-0-7506-8236-1.

[46] Valentine Casey. „Imparting the Importance of Culture to Global Software Development“. In: *ACM Inroads* 1.3 (Sept. 2011), pp. 51–57. ISSN: 2153-2184. DOI: 10.1145/1835428.1835443. URL: http://doi.acm.org/10.1145/1835428.1835443.

[47] Line Dubé and Guy Paré. „The Multi-faceted Nature of Virtual Teams“. In: *In D.J. Pauleen (Ed.), Virtual teams: Projects, protocols, and practices*. Idea Group Publishing, 2002, pp. 1–39.

[48] J.Roberto Evaristo et al. „A dimensional analysis of geographically distributed project teams: a case study“. In: *Journal of Engineering and Technology Management* 21.3 (2004), pp. 175 –189. ISSN: 0923-4748. DOI: http://dx.doi.org/10.1016/j.jengtecman.2003.05.001. URL: http://www.sciencedirect.com/science/article/pii/S0923474804000293.

[49] Liz Lee-Kelley. „Locus of control and attitudes to working in virtual teams“. In: *International Journal of Project Management* 24.3 (2006), pp. 234 –243. ISSN: 0263-7863. DOI: http://dx.doi.org/10.1016/j.ijproman.2006.01.003. URL: http://www.sciencedirect.com/science/article/pii/S0263786306000135.

[50] Thomas W. Malone and Kevin Crowston. „The Interdisciplinary Study of Coordination“. In: *ACM Comput. Surv.* 26.1 (Mar. 1994), pp. 87–119. ISSN: 0360-0300. DOI: 10.1145/174666.174668. URL: http://doi.acm.org/10.1145/174666.174668.

[51] Gamel O. Wiredu. „A Framework for the Analysis of Coordination in Global Software Development“. In: *Proceedings of the 2006 International Workshop on Global Software Development for the Practitioner*. GSD '06. Shanghai, China: ACM, 2006, pp. 38–44. ISBN: 1-59593-404-9. DOI: 10.1145/1138506.1138516. URL: http://doi.acm.org/10.1145/1138506.1138516.

[52] James D. Herbsleb. „Global Software Engineering: The Future of Socio-technical Coordination“. In: *2007 Future of Software Engineering*. FOSE '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 188–198. ISBN: 0-7695-2829-5. DOI: 10.1109/FOSE.2007.11. URL: http://dx.doi.org/10.1109/FOSE.2007.11.

[53] Päivi Ovaska, Matti Rossi, and Pentti Marttiin. „Architecture as a coordination tool in multi-site software development“. In: *Software Process: Improvement and Practice*. 2003, pp. 233–247.

[54] Jo Ellen Moore, Clay K. Williams, and Mary Sumner. „The Role of Informal Control in PMO Lite Environments“. In: *Proceedings of the 50th Annual Conference on Computers and People Research*. SIGMIS-CPR '12. Milwaukee, Wisconsin, USA: ACM, 2012, pp. 27–30. ISBN: 978-1-4503-1110-6. DOI: 10.1145/2214091.2214101. URL: http://doi.acm.org/10.1145/2214091.2214101.

[55] Ravi Narayanaswamy and Raymond M. Henry. „Effects of Culture on Control Mechanisms in Offshore Outsourced IT Projects“. In: *Proceedings of the 2005 ACM SIGMIS CPR Conference on Computer Personnel Research*. SIGMIS CPR '05. Atlanta, Georgia, USA: ACM, 2005, pp. 139–145. ISBN: 1-59593-011-6. DOI: 10.1145/1055973.1056004. URL: http://doi.acm.org/10.1145/1055973.1056004.

[56] Melvin E. Conway. „How do committees invent“. In: *Datamation* 14.4 (1968), pp. 28–31. URL: http://www.melconway.com/Home/pdf/committees.pdf.

[57] Alan MacCormack, Carliss Baldwin, and John Rusnak. „Exploring the duality between product and organizational architectures: A test of the “mirroring” hypothesis“. In: *Research Policy* 41.8 (2012), pp. 1309–1324. URL: http://ideas.repec.org/a/eee/respol/v41y2012i8p1309-1324.html.

[58] James D. Herbsleb and Rebecca E. Grinter. „Splitting the Organization and Integrating the Code: Conway's Law Revisited“. In: *Proceedings of the 21st International Conference on Software Engineering*. ICSE '99. Los Angeles, California, USA: ACM, 1999, pp. 85–95. ISBN: 1-58113-074-0. DOI: 10.1145/302405.302455. URL: http://doi.acm.org/10.1145/302405.302455.

[59] Paul Watzlawick, Janet Helmick-Beavin, and Don D. Jackson. *Pragmatics of Human Communication - a study of interactional patterns, pathologies, and paradoxes*. New York: Norton, 1967. ISBN: 0-393-01009-0.

[60] Friedemann Schulz von Thun. *Miteinander reden 1 – Störungen und Klärungen. Allgemeine Psychologie der Kommunikation.* 51st ed. Rowholt Taschenbuch Verlag, 2014. ISBN: 978-3-499-17489-6.

[61] Richard L. Daft and Robert H. Lengel. „Information Richness: A New Approach to Managerial Behaviour and Organizational Design“. In: *Research in Organizational Behaviour* 6 (1984). Ed. by Bm Staw, pp. 191–233.

[62] Alan R. Dennis and Joseph S. Valacich. „Rethinking Media Richness: Towards a Theory of Media Synchronicity“. In: *Proceedings of the Thirty-Second Annual Hawaii International Conference on System Sciences-Volume 1 - Volume 1*. HICSS '99. Washington, DC, USA: IEEE Computer Society, 1999, pp. 1017–. ISBN: 0-7695-0001-3. URL: http://dl.acm.org/citation.cfm?id=874068.875924.

[63] Kristi Lewis Tyran, Craig K. Tyran, and Morgan Shepherd. „Exploring Emerging Leadership in Virtual Teams“. In: *Virtual Teams That Work: Creating Conditions for Effective Virtual Teams*. Jossey-Bass Inc., Publishers, 2003, pp. 183–195. ISBN: 0787961620.

[64] Erin Bradner, Gloria Mark, and Tammie D. Hertel. „Effects of team size on participation, awareness, and technology choice in geographically distributed teams“. In: *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*. 2003, 10 pp.–. DOI: 10.1109/HICSS.2003.1174795.

[65] Jeff Sutherland et al. „Distributed Scrum: Agile Project Management with Outsourced Development Teams“. In: *Proceedings of the 40th Annual Hawaii International Conference on System Sciences*. HICSS '07. Washington, DC, USA: IEEE Computer Society, 2007, 274a–. ISBN: 0-7695-2755-8. DOI: 10.1109/HICSS.2007.180. URL: http://dx.doi.org/10.1109/HICSS.2007.180.

[66] Maria Paasivaara, Sandra Durasiewicz, and Casper Lassenius. „Using Scrum in Distributed Agile Development: A Multiple Case Study.“ In: *ICGSE*. IEEE, 2009, pp. 195–204. ISBN: 978-0-7695-3710-8. URL: http://dblp.uni-trier.de/db/conf/icgse/icgse2009.html#PaasivaaraDL09.

[67] Bernardo José da Silva Estácio and Rafael Prikladnicki. „Distributed Pair Programming: A Systematic Literature Review“. In: *Inf. Softw. Technol.* 63.C (July 2015), pp. 1–10. ISSN: 0950-5849. DOI: 10.1016/j.infsof.2015.02.011. URL: http://dx.doi.org/10.1016/j.infsof.2015.02.011.

[68] Jo E. Hannay et al. „The Effectiveness of Pair Programming: A Meta-analysis“. In: *Inf. Softw. Technol.* 51.7 (July 2009), pp. 1110–1122. ISSN: 0950-5849. DOI: 10.1016/j.infsof.2009.02.001. URL: http://dx.doi.org/10.1016/j.infsof.2009.02.001.

[69] David Stotts et al. „Virtual Teaming: Experiments and Experiences with Distributed Pair Programming“. English. In: *Extreme Programming and Agile Methods - XP/Agile Universe 2003*. Ed. by Frank Maurer and Don Wells. Vol. 2753. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2003, pp. 129–141. ISBN: 978-3-540-40662-4. DOI: 10.1007/978-3-540-45122-8_15. URL: http://dx.doi.org/10.1007/978-3-540-45122-8_15.

[70] Brian Hanks. „Empirical Evaluation of Distributed Pair Programming“. In: *Int. J. Hum.-Comput. Stud.* 66.7 (July 2008), pp. 530–544. ISSN: 1071-5819. DOI: 10.1016/j.ijhcs.2007.10.003. URL: http://dx.doi.org/10.1016/j.ijhcs.2007.10.003.

[71] Michael Reeves and Jihan Zhu. „Moomba – A Collaborative Environment for Supporting Distributed Extreme Programming in Global Software Development“. English. In: *Extreme Programming and Agile Processes in Software Engineering*. Ed. by Jutta Eckstein and Hubert Baumeister. Vol. 3092. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, pp. 38–50. ISBN: 978-3-540-22137-1. DOI: 10.1007/978-3-540-24853-8_5. URL: http://dx.doi.org/10.1007/978-3-540-24853-8_5.

[72] Jesus Favela et al. „Empirical Evaluation of Collaborative Support for Distributed Pair Programming“. English. In: *Groupware: Design, Implementation, and Use*. Ed. by Gert-Jan de Vreede, LuisA. Guerrero, and Gabriela Marín Raventós. Vol. 3198. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, pp. 215–222. ISBN: 978-3-540-23016-8. DOI: 10.1007/978-3-540-30112-7_18. URL: http://dx.doi.org/10.1007/978-3-540-30112-7_18.

[73] Despina Tsompanoudi, Maya Satratzemi, and Stelios Xinogalos. „Exploring the Effects of Collaboration Scripts Embedded in a Distributed Pair Programming System“. In: *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education*. ITiCSE '13. Canterbury, England, UK: ACM, 2013, pp. 225–230. ISBN: 978-1-4503-2078-8. DOI: 10.1145/2462476.2462500. URL: http://doi.acm.org/10.1145/2462476.2462500.

[74] Gerardo Canfora et al. „How distribution affects the success of pair programming“. In: *International Journal of Software Engineering and Knowledge Engineering* 16.02 (2006), pp. 293–313.

[75] Prashant Baheti, Edward F. Gehringer, and P. David Stotts. „Exploring the Efficacy of Distributed Pair Programming“. In: *Proceedings of the Second XP Universe and First Agile Universe Conference on Extreme Programming and Agile Methods - XP/Agile Universe 2002*. London, UK, UK: Springer-Verlag, 2002, pp. 208–220. ISBN: 3-540-44024-0. URL: http://dl.acm.org/citation.cfm?id=647276.722333.

[76] Gerardo Canfora, Aaniello Cimitile, and Corrado A. Visaggio. „Lessons learned about distributed pair programming: what are the knowledge needs to address?“ In: *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on*. 2003, pp. 314–319. DOI: 10.1109/ENABL.2003.1231429.

[77] Paul Duvall, Stephen M. Matyas, and Andrew Glover. *Continuous Integration: Improving Software Quality and Reducing Risk (The Addison-Wesley Signature Series)*. Addison-Wesley Professional, 2007. ISBN: 0321336380.

[79] Ade Miller. „A Hundred Days of Continuous Integration“. In: *Agile, 2008. AGILE '08. Conference*. 2008, pp. 289–293. DOI: 10.1109/Agile.2008.8.

[81] J.M. Verner et al. „Guidelines for industrially-based multiple case studies in software engineering“. In: *Research Challenges in Information Science, 2009. RCIS 2009. Third International Conference on*. 2009, pp. 313–324. DOI: 10.1109/RCIS.2009.5089295.

[82] Timothy C. Lethbridge, Susan Elliott Sim, and Janice Singer. „Studying Software Engineers: Data Collection Techniques for Software Field Studies“. In: *Empirical Softw. Engg.* 10.3 (July 2005), pp. 311–341. ISSN: 1382-3256. DOI: 10.1007/s10664-005-1290-x. URL: http://dx.doi.org/10.1007/s10664-005-1290-x.

[83] Andreas Diekmann. *Empirische Sozialforschung: Grundlagen, Methoden, Anwendungen. Reinbek bei Hamburg*. 2010.

[84] Mikko Korkala and Pekka Abrahamsson. „Communication in Distributed Agile Development: A Case Study“. In: *33rd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO 2007)*. 2007, pp. 203–210. DOI: 10.1109/EUROMICRO.2007.23.

[85] Lucas Layman et al. „Essential communication practices for Extreme Programming in a global software development team“. In: *Information and Software Technology* 48.9 (2006), pp. 781–794. URL: http://www.sciencedirect.com/science/article/B6V0B-4JF8H93-2/2/625aa9c8d29f7c60b2a5a38425f9a41f.

[86] Paul Karsten and Fabrizio Cannizzo. „The Creation of a Distributed Agile Team“. In: *Proceedings of the 8th International Conference on Agile Processes in Software Engineering and Extreme Programming*. XP'07. Como, Italy: Springer-Verlag, 2007, pp. 235–239. ISBN: 978-3-540-73100-9. URL: http://dl.acm.org/citation.cfm?id=1768961.1769014.

Online References

[3] Stack Overflow. *Stack Overflow Developer Survey 2016*. 2016. URL: http://stackoverflow.com/research/developer-survey-2016#work-remote (visited on 06/19/2016).

[4] Stack Overflow. *Stack Overflow Developer Survey 2015*. 2015. URL: http://stackoverflow.com/research/developer-survey-2015#work-remote (visited on 06/19/2016).

[23] Kent Beck et al. *Manifesto for Agile Software Development*. 2001. URL: http://www.agilemanifesto.org/ (visited on 06/19/2016).

[34] Scott W. Ambler. *Agile Adoption Rate Survey Results: February 2008*. June 2001. URL: http://www.drdobbs.com/open-source/the-agile-manifesto/184414755 (visited on 06/19/2016).

[78] Martin Fowler. *Continuous Integration*. May 2006. URL: http://www.martinfowler.com/articles/continuousIntegration.html (visited on 06/19/2016).

[80] Philipp Mayring. *Qualitative content analysis: theoretical foundation, basic procedures and software solution*. 2014. URL: http://nbn-resolving.de/urn:nbn:de:0168-ssoar-395173 (visited on 06/19/2016).

A Appendix

A.1 Interview Guideline

Interviewleitfaden - Agile Development in Distributed Teams

1. Herzlichen Dank, dass Sie sich Zeit genommen haben für dieses Interview. Würden Sie sich bitte kurz vorstellen?
2. Könnten Sie bitte kurz ein paar Eckdaten zu Ihrer Firma/Team liefern? Welche Art von Projekten führen Sie durch und wieviele Personen sind beschäftigt?
3. Je nach Tätigkeit der interviewten Person:
    a) Wieviele Personen arbeiten in Ihrem Team?
    b) Was ist ihre konkrete Aufgabe im Team?
4. Wie sieht die geographische Verteilung in Ihrem Team aus? (Auf wie viele Orte ist das Team verteilt und wie viele Personen befinden sich an dem jeweiligen Ort? Wie weit befinden sich die Orte auseinander?)
5. Aus welchen Gründen ist es zu dieser Verteilung gekommen?
6. Welche Vor und Nachteile ergeben sich Ihrer Meinung nach aus der Verteilung?
7. Welches (Agile) Prozessmodell setzen Sie in Ihrem Team ein?
8. Aus welchem Grund haben Sie sich entschieden, Agile Prozesse in ihrem Team einzusetzen?
    a) Seit wann setzen sie auf Agile?
    b) Mit welchem Prozessmodell haben Sie vorher gearbeitet?
9. Können Sie kurz skizzieren, wie ein Projekt abläuft, welche Phasen es gibt?
10. Wie läuft eine Iteration (Sprint) ab und welche Phasen es gibt?
11. Wie wird in Ihrem Team kommuniziert? (Welche technischen Lösungen setzen Sie ein um die Distanz zu überbrücken?)
    a) Wie funktioniert die Kommunikation mit externen Stakeholdern?
12. Wie wichtig schätzen Sie direkten face-to-face Kontakt zwischen Teammitgliedern ein?
13. Wie funktioniert der Wissensaustausch in ihrem Team?
14. Gibt es Situationen wo sich das gesamte Team am selben Ort trifft? Wenn ja, wie oft und zu welchen Anlässen kommt es dazu?
    a) Wenn nein: Gibt es Situation wo sich zumindest einzelne Teammitglieder persönlich treffen?

15. Welche Agilen Prozesse werden sonst noch in Ihrem Team eingesetzt? (Iterationen, Pair Programming, Daily Scrum, Sprintplanung, TDD, Retrospektiven, Code Reviews . . .)
    a) Wie genau sieht der Einsatz von Prozess X in dem verteilten Umfeld aus?
    b) Ergeben sich durch die verteilte Situation Probleme beim Einsatz von Prozess X?
    c) Mussten Sie Prozess X anpassen, um ihn in ihrem Team anwenden zu können?
    d) Setzen Sie spezielle (Software)Lösungen als Hilfsmittel für Prozess X ein?
16. Setzen Sie irgendwelche generellen Tools oder auch Hardware ein um diese Prozesse zu unterstützen? (zb Jira, Trello, Redmine, . . .)
17. Setzen Sie Techniken ein um gewisse Informationen innerhalb des Teams (Buildstatus, Projektstatus, . . .) dauerhaft einsehbar zu kommunizieren? (Information Radiators)
18. Was und wie wird bei Ihnen dokumentiert? Setzen sie aufgrund der Verteilung auf mehr Dokumentation?
19. Wie sieht es in Ihrem Team bei den Aspekten Koordination und Kontrolle aus?
20. Wie zufrieden sind Sie mit der Kommunikation im Team? Wo liegen Stärken und Schwächen?
21. Gibt es Prozesse die Sie eingesetzt haben, die aber aufgrund der verteilten Situation gescheitert sind bzw so nicht praktikabel waren?
22. Gibt es Prozesse die Sie gerne einsetzen würden, aber bis jetzt (aufgrund der Verteilung) nicht tun?
    a) Aus welchen Gründen?
    b) Planen Sie diesen Prozess noch einzuführen?
23. Denken Sie, dass Agile Prozesse in verteilten Teams hilfreich sind? Wo liegen Ihrer Meinung nach die Vor- und Nachteile?
    a) Was sind die wichtigsten Aspekte bei Agiler Softwareentwicklung in verteilten Teams?
24. Macht es einen Unterschied ob Teams im selben Land oder über mehrere Kontinente verteilt sind? Wo liegen die Unterschiede?

A.2 Quotes

This section lists the original quotes from the interviews, grouped by the unit of analysis and also pointing out the role of the person who said it.

A.2.1 Alpha

Alpha #1 - CEO „Wir haben halt einfach sehr wenige Mitarbeiter auf sehr viele Scrum Projekte aufgeteilt und du müsstest dann mehrere Rollen in verschiedenen Projekten haben, das würde viel zu viel Overhead sein. Teilweise sind es auch Projekte wo das Team aus 2 Leuten besteht, da geht das einfach nicht. Das ist aber eher ein Problem vom Verhältnis von Mitarbeitern und verschiedenen Projekten.“

Alpha #2 - CEO „Wir sind draufgekommen das es für uns, eben wegen der viele Projekte und unterschiedliche Prioritäten bei diesen, diese Story Points eher für eine grundsätzliche Einschätzung wie schwer ein Issue ist geeignet sind, uns ist es aber dann egal ist wieviele Issues wir durchbringen weil das sehr davon abhängt wieviele Projekte gerade durchgeführt werden und je nach Priorität sich da Verschiebungen ergeben."

Alpha #3 - CTO „Man muss halt natürlich auch - und das zeigt sich schon sehr deutlich in letzter Zeit - den Prozess immer wieder anpassen. Wir haben ja einige Veränderungen in den letzten Jahren gemacht wo wir einfach gemerkt haben, durch die örtliche Trennung muss man gewisse Dinge machen, die man sonst nicht machen muss. Von der Abstimmung her, das man besser mitbekommt was passiert denn am anderen Ort, was tun denn die gerade. Wie eben, diese Sprintplanung - Scrum mäßig - das man das eben etwas besser verflechtet und besser weiß was machen wir denn jetzt, was ist zu tun. Damit man nicht so nebeneinander hinarbeitet."

Alpha #4 - CTO „Was ich noch einwerfen würde was irrsinnig wichtig ist, ist das Code Review in Form von Pull Requests. Das - so wie wir es aufgesetzt haben - auch einfach funktioniert und das wir dadurch auch mitbekommen was tun denn die anderen. Es ist natürlich auch nützlich für die Code Qualität aber man bekommt einfach auch mit wer an was arbeitet."

Alpha #5 - CEO „Es ist eine bessere Form von Code Reviews, es ersetzt für uns zu 90 Prozent die Code Reviews, weil wenn wir uns hinsetzten und sagen: Heute machen wir ein Code Review, dann stehst du vor so einem großen Ding, wo es schon viel zu spät ist ein Code Review zu machen."

Alpha #6 - CEO „Es ist sicherlich nötig für dieses Teamgefüge, dass man sich da wiedermal sieht, dass man sich gemeinsam austauscht. Auch zu nicht Job Themen vielleicht. Es ist aber nicht permanent notwendig wenn es darum geht etwas umzusetzen."

Alpha #7 - CTO „Wir wollen uns auch noch zusammensetzen wo wir dann über ein Thema diskutieren wo wir vielleicht eine Stunde oder zwei diskutieren müssen, ein bisschen Brainstormen, das geht halt im GoTo Meeting nicht so gut, einfach weil man nicht so interaktiv und so gut beieinander ist. Da ist es einmal nett sich so zusammenzusetzen. Aber das sind eher die Ausnahmen, weil die meisten Projektbesprechungen, Standups, Sprintplanungen und so weiter gehen gut über GoTo Meeting."

Alpha #8 - CEO „Wir waren von Beginn an verteilt. Dadurch, dass Andreas in Linz und wir hier waren, waren wir von Beginn an, von der ersten Minute an verteilt und haben deshalb auch von Beginn an in diese Richtung das aufgesetzt. Parallel dazu hat sich auch, sag ich mal, in der Entwicklung in Richtung von Remote Work was getan und somit sind wir mehr oder weniger auf dem Zug drangeblieben weil man gesehen hat, das machen andere auch so."

Alpha #9 - CEO „Das Themenfeld in der Entwicklung wird immer breiter, es wird immer schwieriger das man zu einem gewissen Thema in diesem breiten Spektrum an Technologien Spezialisten findet, die dann genau am Punkt x - oder wo auch immer - sitzen. Somit haben wir gesagt: Naja wenn wir sowieso von Beginn an so Erfahrungen damit gesammelt haben, dann bleiben wir da drauf und nutzen die Vorteile daraus und es gibt somit eigentlich keine örtliche

oder zeitliche Bedingung an die Arbeit bei Alpha. [Hier wurde der Name des Unternehmens durch 'Alpha' ersetzt]“

Alpha #10 - CEO „Es kommt immer wieder vor das einer Homeoffice macht. Einfach wegen der Familie, wir haben alle Kinder, und das erleichtert das ganze extrem wenn ich sagen kann: Ok ich arbeite heute den Tag zuhause weil man am frühen Morgen noch etwas erledigen muss und sich den Weg ins Büro einfach erspart. [...] Das ist einfach automatisch gegeben weil alles vorbereitet ist für remote, da ist Home-Office automatisch dabei, alles was ich brauche ist eine Internetverbindung - und das kann man dann natürlich auch Mitarbeitern bieten.“

Alpha #11 - CTO „Aber trotzdem, das würde ich jetzt als kleinen- oder halt als Nachteil bezeichnen, man ist halt schon ein bisschen entkoppelt. Wenn ich wieder einen Tag da bin so wie heute, dann ruft man sich eben mal schnell quer durch den Raum irgendetwas zu, oder bekommt mit das die einen Zwei da an dem Problem gerade arbeiten und du könntest da jetzt auch einen Beitrag leisten oder auch halt nicht. Diese Entkopplung ist ein bisschen ein Nachteil, ist schon ein bisschen negativ für mich finde ich.“

Alpha #12 - CEO „Die Herausforderung war eigentlich: Wir haben uns nie die Frage gestellt ob wir so verteilt arbeiten können, sondern wir haben uns eher die Frage gestellt: Ist es eine Gefahr, dass sich dann zwei Blöcke bilden, oder das Andreas in Linz ausgeschlossener wäre.“

Alpha #13 - CTO „Dann ist es nicht so das es vier - zwei - vier ist, sondern dann ist es vier - zwei -acht und da merkt man schon, da entsteht eine ganz andere Dynamik bei den acht Leuten. Da wird viel mehr untereinander geredet, da geht in dem Projekt viel mehr weiter als sonst unterm Jahr, und man ist aber in Linz nicht mitbeteiligt an dem so richtig.“

Alpha #14 - CEO „Aber es gibt noch jede Menge zu tun, und es ist teilweise schon auch so das technische Sachen Hemmnisse sind. Wie schlecht teilweise echt eine Video über Internet Konferenz funktioniert. Bei den Tools dafür haben wir auch noch nicht das perfekte: Mikrofone, Kameras… dieses technische Zusammenspiel ist leider ein Punkt wo es noch nicht so smooth geht das man es einfach so nutzt. Das ist einfach etwas wo die Technik dem wie die Arbeit sich entwickelt hat nachhängt.“

A.2.2 Beta

Beta #1 - CEO „Wir haben eigentlich begonnen als Software Agentur, wir bedienen eigentlich die Industrie und machen damit auch die meisten Projekte. Die Entwicklung von Web-Portalen und das Online Marketing ist hinzugekommen weil zurzeit Bedarf da ist. Was wir tun ist: wir konzipieren, entwickeln und vermarkten digitale Produkte - das ist eigentlich genau unser Alleinstellungsmerkmal.“

Beta #2 - CEO „Wir haben gemerkt innerhalb des Teams ist jeder in Zugzwang sich zu verbessern und sich auf das Wesentliche zu konzentrieren. Es ist wichtig, dass jeder Mitarbeiter selbstständig arbeiten kann eben weil wir an verschiedenen Standorten sind, also das hat sowieso zum Glück von Anfang an funktioniert.“ [Beta #2 - CEO]

Beta #3 - CEO „Pair Programming ist sowieso obligatorisch im Sinne der Wissensübertragung, wir haben eben Spezialisten und jeder für sich ist spezialisiert in einem gewissen Bereich - der eine betreibt 3D-Entwicklung, der andere Datenverarbeitung der nächste für Machine Learning, ein dritter eben für Frontend Entwicklung. Und da möchten wir schon, dass Leute ihr Wissen hin und wieder übergeben, einfach nur wenn jemand ausfällt - zum Beispiel in Urlaub geht - das der andere weiß was er gemacht hat.“ [Beta #3 - CEO]

Beta #4 - CEO „Gegenwärtig sind wir sehr zufrieden mit dem Prozess, da möchten wir dabei bleiben. Natürlich möchten wir den Organisationsaufwand so weit als möglich gering halten.“ [Beta #4 - CEO]

Beta #5 - CEO „Wir möchten dass sich die Mitarbeiter in der Arbeit auf die Arbeit konzentrieren können, (...) wir haben die Erfahrung gemacht - also von unseren vorigen Jobs - das es vorteilhaft ist wenn man das private Communicator Konto eben nicht verwendet für die alltägliche Arbeit. Man ist einfach zu abgelenkt.“ [Beta #5 - CEO]

Beta #6 - CEO „Wir benutzen Skype und da wird eben der Screen geshared mit dem Board, und eben der Scrum Master kümmert sich darum das er die Tickets verschiebt im Namen der Kollegen die anwesend sind.“ [Beta #6 - CEO]

Beta #7 - CEO „So ist es auch für den Kunden transparenter wenn er an den Daily Standup Meetings teilnimmt: eben das da wirklich auch was passiert und so weiß er wofür er sein Geld ausgibt.“ [Beta #7 - CEO]

Beta #8 - CEO „Ob es gleich effizient ist kann ich jetzt nicht sagen, es wäre sicher ein bisschen effizienter wenn man vor Ort ist. Auf der anderen Seite ist sicher der Vorteil - oder aufgrund der Professionalität der Leute - gibt es bei uns keine email Lawinen oder man überlegt sich in dem Fall wenn man nicht direkt vor Ort ist: Möchte ich diese Person jetzt stören oder nicht die an meinem Projekt arbeitet? Das ist vielleicht ein Vorteil.“ [Beta #8 - CEO]

Beta #9 - CEO „Was ich mir natürlich wünschen würde wäre wenn ich nicht so oft fahren müsste. Aber das ist für mich kein Thema, das haben wir uns so ausgesucht deswegen haben wir gute Bereichsabdeckung und Nachfragen von verschiedenen Ballungszentrum. Also oft ist es so, dass wir Wiener Unternehmen als Kunden haben, wir haben durch die Verteilung eben die Chance bei Kunden reinzukommen die eigentlich einen Vor-Ort Service haben wollen.“ [Beta #9 - CEO]

Beta #10 - CEO „Die persönlichen Beziehungen zueinander sind sicher nicht so eng wie in Teams die jetzt alle in einem Raum sitzen und zusammen Mittagessen gehen. Die Personen vor Ort verstehen sich natürlich sehr freundschaftlich das ist schon ein riesengroßer Vorteil. Der Nachteil ist eben das wir uns nicht so oft sehen wie wir wollen, sagen wir mal so.“ [Beta #10 - CEO]

Beta #11 - CEO „Also ganz ehrlich, das Nadelöhr am Ganzen ist die Internetverbindung. Also wenn UPC einmal nicht geht dann haben wir natürlich ein Problem. Deswegen sind wir umgestiegen auf A1 Internet, das ist ein wenig stabiler. Es ist zwar nicht das schnellste aber funktioniert überall. Das Nadelöhr am Ganzen, der Nachteil ist eben was wenn kein Internet da ist, oder wenn die Verbindung schlecht ist. Das ist vor allem ein technisches Problem.“ [Beta #11 - CEO]

Beta #12 - CEO „Wir treffen uns alle zwei Monate irgendwo und unternehmen irgendwas, also zum Beispiel am Wochenende. Also das Verhältnis zueinander ist schon auch zueinander im privaten Leben vorhanden. Aber nicht so wie in einer Softwarefirma wo man sich am Abend zusammensetzt.“ [Beta #12 - CEO]

Beta #13 - CEO „Selbstständigkeit ist bei uns sehr sehr wichtig, und die Eigenverantwortlichkeit für die eigenen Aufgaben ist sehr sehr wichtig. Und das ist nicht so einfach Personal zu finden oder Entwickler zu finden die das auch hundertprozentig leben können.“ [Beta #13 - CEO]

Beta #14 - CEO „Wir haben schon auch die Erfahrung gemacht das wir Entwickler gehabt haben die genau diesen Anspruch eben nicht erfüllt haben von denen wir uns dann natürlich getrennt haben weil es nicht in unser Gruppenmodell reinpasst.“ [Beta #14 - CEO]

Beta #15 - CEO „Die Externen bekommen einen Zugang zu dem Jira Board. Diejenigen die sich damit nicht auskennen haben ein zweites Board, eben über Trello zum Beispiel, welches viel einfacher gestaltet ist und das auch externe Mitarbeiter beziehungsweise Teammitglieder nutzen können und auch Kunden nutzen können. “ [Beta #15 - CEO]

A.2.3 Gamma

Gamma #1 - Team Leader „Deshalb haben wir letztendlich eine Art Proxy Product Owner, das macht der Projektleiter der eben letztendlich gewisse KnowHow Konzepte bündelt und als erster Ansprechpartner für das Team als Product Owner dient und dann punktuell Rückfragen stellt. Es löst sich dadurch letztendlich ein wenig auf weil einzelne Leute im Team mehr Kundensichtbarkeit erlangen weil andere, aber letztendlich würden wir ohne einen Proxy Product Owner nicht arbeiten können. Wir hätten dann unglaublich lange Sprintvorlaufzeiten und unglaublich lange Sprintdauern in denen wir dauernd wieder den Scope ändern müssten weil Antworten auf Fragen zu lange dauern.“ [Gamma #1 - Team Leader]

Gamma #2 - Team Leader „Wir sind sofort mit etwas an Scrum orientiertem gestartet. Und wir hatten auch Phasen in denen wir abgewichen sind, einfach aus mangelnder Disziplin. Das ist einfach eine Herausforderung bei agilen Methoden finde ich, dass sie sehr strikt sind und zwar für die Produktentwicklung Flexibilität bieten aber Abweichung vom Prozessmodell wenig tolerieren.“ [Gamma #2 - Team Leader]

Gamma #3 - Team Leader „Wir hatten am Anfang drei Wochen weil wir das erst austesten mussten und viele Infrastruktur Themen da waren die immer lange dauern. Auf die eine Woche sind wir aus einem ganz einfachen Grund gekommen: Das man wirklich schneller Ergebnisse sieht und das man die Tendenz etwas eindämmt die Sachen zu schieben oder fertig zu lassen. Eine Woche erfordert eben sehr viel Disziplin und die Stories entsprechend klein und präzise zu schneiden. “ [Gamma #3 - Team Leader]

Gamma #4 - Team Leader „ Also insgesamt sind wir mit der Woche sehr zufrieden, wir würden nicht wieder zurück wechseln wollen, obwohl auch Leute aus Trainings zurückkamen von uns die meinten: Der Trainer meinte eine Woche ginge gar nicht, könne gar nicht funktionieren. Scrum also mindestens zwei und eher drei Wochen. Das scheint eine religiöse Frage zu sein.“ [Gamma #4 - Team Leader]

Gamma #5 - Team Leader „Wir übernehmen Themen sehr schnell in Tickets oder ins Wiki, das bedeutet wenn jemand etwas macht oder auch ein Review braucht dann kommen halt die Fotos vom Whiteboard an ein Ticket dran oder auch ins Wiki. Da warten wir dann nicht bis man das formal irgendwo reingezeichnet hat oder in speziellen Tools visualisiert, sondern das ist sehr locker. Auf die Art schafft man auch ja relativ kurze Zyklen, also Foto vom Whiteboard machen und einstellen, oder eine Telefonspinne geschnappt, anrufen und sagen: "Wir sitzen hier gerade zusammen und überlegen uns eine Optimierung, kannst du bitte mal draufkucken, ich hab das mal abfotografiert.“ [Gamma #5 - Team Leader]

Gamma #6 - Team Leader „In einem Teambüro haben wir einen Monitor, aber der wird glaube ich kaum verwendet. Auch in anderen Teams laufen auf den Monitoren wenn ich dort vorbeilaufe eher witzige Youtube Videos oder sonst irgendwas, oder Code Highlights - in der Regel schlechter Code - aber tatsächlich Information wird darüber kaum verteilt. Da kuckt niemand, ich würde auch nicht zu einem Monitor hinlaufen und das tut auch sonst keiner.“ [Gamma #6 - Team Leader]

Gamma #7 - Team Leader „Nicht im Bereich Agiler Methoden oder Vorgehensweisen. Das bezieht sich eher darauf das natürlich Prozessautomatisierung nie am Ende ist; und Infrastruktur nie gut genug; und Buildzeiten nie kurz genug; und Codequalität nie gut genug. Also das heißt das bezieht sich eher auf technische Dinge und Infrastruktur.“ [Gamma #7 - Team Leader]

Gamma #8 - Team Leader „Wir werden denke ich mal mit einem Enterprise Messenger Experimente starten. Mal sehen, wenn die Leute sagen das ist cool und hilft uns, wir werden besser dadurch, dann machen wir es. Wenn wir feststellen, dass die Sachen nicht genutzt werden stellen wir sie halt wieder ein. Also auch da arbeiten wir eher Agil, das wir Sachen ausprobieren, erste Schritte machen, sie auch verbessern aber auch wieder verwerfen. Weil es wenig bringt jetzt glaube ich einem Team die Arbeitsmittel aufzudrücken. “ [Gamma #8 - Team Leader]

Gamma #9 - Team Leader „Der gefühlte Kommunikationsoverhead bei den meisten Agilen Methoden ist wenn man das mal genauer betrachtet glaube ich garnicht so hoch. Täglich eine Viertelstunde pro Teammitglied - da verbringen die Leute mehr Zeit in der Kaffeeküche über den Tag. Also eine Viertelstunde Arbeitszeit für Kommunikation am Tag offiziell sind überhaupt kein Schmerz.“ [Gamma #9 - Team Leader]

Gamma #10 - Team Leader „In erster Linie per Telefon. per Telefon oder persönlich. Es gibt aber auch Leute die setzen ganz gerne Instant Messenger ein. In der Regel wird die Kommunikation flankiert von Desktop Sharing Tools, Teamviewer um wirklich gemeinsam auf das Board zu sehen oder auf den Code zu sehen oder was auch immer. Aber sonst wirklich überraschend viel Telefon.“ [Gamma #10 - Team Leader]

Gamma #11 - Team Leader „So und das führt eben dazu, das wir alle paar Wochen die Teams zusammenziehen, meist wechselnd mal in LOCATION mal in LOCATION um uns da wieder abzugleichen, neue Entwicklung oder neue größere Themen gemeinsam anzugehen, zu planen, Pair Programmings zu machen um halt genau den Know How Austausch zu fördern.“ [Gamma #11 - Team Leader]

Gamma #12 - Team Leader „Was ein bisschen schwierig ist was wir feststellen, wenn man Mitarbeiter hat die nicht Deutsch sprechen, und bei denen man das Handycap im Normalfall kaum

feststellt - am Telefon aber schon - das ist ein immer währendes Thema. Darauf würde ich in Zukunft auch stärker achten. Das funktioniert nämlich wirklich nicht so gut. Also wenn man Leute halt einfach über das Telefon schlechter versteht, und das dann für Kollegen anstrengend wird. Und wenn Kommunikation anstrengend wird, dann umgeht man sie natürlich.“ [Gamma #12 - Team Leader]

Gamma #13 - Team Leader „Wir sind etwa 200 Leute und wenn man plötzlich ein Team von sechs bis acht Leuten braucht, inzwischen sind es zehn insgesamt, dann ist es bei 200 Leuten garnicht so einfach das passend hinzubekommen, das ist der eine Punkt. Der zweite Punkt ist, dass die Standorte aus historischen Gründen etwas unterschiedliche Schwerpunkte von der Technologie hatten, und Portalgeschäft hatten einige aus LOCATION mehr Erfahrung, deshalb haben wir mit einem kleinen Teamanteil begonnen und den Teil dann ausgebaut weil es so gut klappt. “ [Gamma #13 - Team Leader]

Gamma #14 - Team Leader „Wir können sowohl in LOCATION als auch in LOCATION auf unterschiedliche Kompetenzen zugreifen, weil die Standorte unterschiedliche Schwerpunkte haben. Bei dem einen ist halt irgendwie das Java-Backend Know How ausgeprägter und wenn man dort Hilfe braucht um irgendwelche seltsamen Bugs zu jagen findet man die halt an dem einem Standort vielleicht besser, während man Security Experten im Web Bereich am anderen Standort besser findet. So haben wir in Summe zugriff auf ein deutlich größeres Mitarbeiternetzwerk.“ [Gamma #14 - Team Leader]

Gamma #15 - Team Leader „Ein kleinerer Nachteil ist das sich KnowHow in Verteilung - und zwar ich glaube prinzipiell - nicht so gut harmonisieren lässt. Also die Leute greifen häufig zum Telefon und machen häufig einen Teamviewer auf, aber es ist das nicht das Gleiche wie wenn ich zwei Meter weiter laufe und einen Kollegen frage ob er mit mir schnell mal auf den Bildschirm sieht. Da sind Tools, so gut sie sein mögen, halt doch nochmal eine Hürde.“ [Gamma #15 - Team Leader]

Gamma #16 - Team Leader „Also alle Projekte die ich kenne enden immer damit, das man die Offshore partner nach Deutschland fliegt und sie dann hier arbeiten. Also da bin ich extrem skeptisch und das muss schon, ja, das kann glaub ich gehen bei Software Wartung, also wenn man rein technische Aspekte hat, aber sobald es fachlich wird wird es halt schwierig. Und weil man die Fachlichkeit in einem agilen Team hat, und nicht an einen Anforderungsmanager abdrückt sondern der Entwickler das sozusagen mitmacht oder gefordert ist, vermute ich das das sehr sehr schwierig wird. “ [Gamma #16 - Team Leader]

Gamma #17 - Team Leader „Wenn ich da unterschiedliche Prozesse hab die teilweise neben dem Toolset laufen, verliert das ganze System an Aussagekraft weil ich mich plötzlich auf nichts mehr verlassen kann. Weil jedesmal wenn ich etwas nicht sehe da nachhaken müsste ob jetzt was passiert ist oder nicht.“ [Gamma #17 - Team Leader]

Gamma #18 - Team Leader „ Für unsere Firma würde ich sagen ist direkter Kontakt wichtig, das liegt aber auch daran, dass wir so von der Firmenkultur aufgestellt sind. Also nachmittagliches Kickern, abends Coding Dojos, mal Pizza und Bier, Freitags Innovationsthemen wo sich Leute im Prinzip von der Firma zur Verfügung gestellter freier Zeit mit eigenen Themen befassen und die mit anderen Kollegen vorantreiben. Also diese Zusammenarbeit und der Austausch hat halt einfach einen hohen Stellenwert. “ [Gamma #18 - Team Leader]

Gamma #19 - Team Leader „Die müssen höher qualifiziert sein, das macht sich bemerkbar. Agile Vorgehensweisen erfordern in meinen Augen besser qualifizierte und engagiertere Mitarbeiter. Einfach damit es nicht untergeht im Rauschen. Weil man auch sichtbar wird und weil die Prozesse darauf ausgelegt sind, zügig durchgeführt zu werden und die kontinuierliche, ständige Verbesserung im Fokus steht.“ [Gamma #19 - Team Leader]

A.2.4 Delta

Delta #1 - Team Leader A „Zum Beispiel im Backlog gehen wir nach Kanban vor im Team, Planung ist aber eher so auf Scrum. Natürlich haben wir wie man es in Scrum kennt so Retros, die machen wir aber nicht fix, sondern wir machen das je nach Bedarf.“ [Delta #1 - Team Leader A]

Delta #2 - Team Leader B „Wir haben am Anfang nur Scrum verwendet für einige Jahre, dann hatten wir ein Team mit Kanban. Jetzt haben wir eine Mischform, wir haben das Beste von jeder Technik genommen und an uns angepasst.“ [Delta #2 - Team Leader B]

Delta #3 - Team Leader A „Es gibt immer Meetings wo man es nie schaffen wird das alle Mitarbeiter verfügbar sind. Aber es gibt auch Meetings - wie zum Beispiel von der Unternehmensführung - die sollte jeder mitbekommen. Da gehts darum was sind die nächsten Schritte, Ziele, und so weiter.“ [Delta #3 - Team Leader A]

Delta #4 - Team Leader A „ Und es ist jetzt auch die Idee immer mehr Dokumentation über Video zu machen - auch intern - weil früher hat man das ins Confluence reingeschrieben: Wie benutze ich den Passwort manager? Und dazu soll es aber in Zukunft ein Video geben, weil es einfach schneller geht. Jemand schaut sich das fünf Minuten an und versteht es einfacher was gemeint ist.“ [Delta #4 - Team Leader A]

Delta #5 - Team Leader B „Wir haben unser Planning Board in Jira, es wird alles über Jira gemanaged. Im Jira kann man von jedem Team den Fortschritt ansehen, was noch offen ist. Wir haben früher Post-Its gehabt, und da ist natürlich das Problem das sehen die Leute remote nicht - mit Kamera sieht mans auch nicht so gut - und deswegen wurde entschieden - doppelt wollten wir es auch nicht machen mit post-its - deswegen verwenden wir nur Jira.“ [Delta #5 - Team Leader B]

Delta #6 - Team Leader A „Die Erfahrung hat gezeigt: von selbst aus machen es wenig Leute, man muss fast immer fixe Meetings machen wo man sich abstimmt. Von selbst aus bekommt man wenig mit als Aussenstelle wie es jetzt jetzt zum Beispiel in Wien ist wenn man keine fixen Termine hat. “ [Delta #6 - Team Leader A]

Delta #7 - Team Leader A „Eigentlich muss man sagen hätte ich noch nie in den letzten sieben Jahren eine Situation bei uns ergeben wo man sagt die müssen jetzt alle kommen damit man das Projekt starten kann. Was man natürlich schon oft macht ist, wenn zum Beispiel bei einem Thema viele aus Kosice beteiligt sind und es ist eine Art kickoff dann laden wir die schon ein, dann sagen wir wir haben was anderes auch noch zu erledigen oder wir sind abgesprochen wenn die Designer auch in Wien sind, die kommen dann schon oft direkt her.“ [Delta #7 - Team Leader A]

Delta #8 - Team Leader A „Manchmal auch noch - das sollte zwar eigentlich kein Problem sein - aber Englisch. Manche tun sich einfach schwerer wenn sie in Englisch etwas reden müssen als wenn sie in ihrer Muttersprache, also Deutsch sprechen könnten.“ [Delta #8 - Team Leader A]

Delta #9 - Team Leader B „Face to Face ist es leichter einander zu verstehen, vor allem wenn man nicht die gleiche Sprache spricht. Ich bin auch kein Native Speaker in Deutsch und wenn man Face to Face spricht ist es immer viel leichter sich zu verstehen, über Telefon oder Skype kommt es leichter zu Missverständnissen.“ [Delta #9 - Team Leader B]

Delta #10 - Team Leader B „Zu den Nachteilen gehört sicher - wie wir selber gerade merken - kommunizieren über digitale Technologie ist zwar aufgrund von Skype und GoTo Meeting schon um einiges einfacher als früher, als über Telefon, aber trotzdem, es ist etwas anderes als wenn die Person im selben Raum ist.“ [Delta #10 - Team Leader B]

Delta #11 - Team Leader A „Was man vermisst sind diese grundlegenden Sachen, ob dein Kollege noch in der Arbeit ist oder nicht. Wenn ihr im gleichen Raum sitzt schaust du einfach auf seinen Tisch, fragst nach - okay er ist nach Hause gegangen. Wenn er in Linz sitzt, weißt du gar nichts; sein Skype läuft noch und rufst an und bekommst keine Antwort und fragst dich warum und kommst nach einer halben Stunde drauf das er nach Hause gegangen ist. Das sind grundlegende kommunikative Missverständnisse.“ [Delta #11 - Team Leader A]

Delta #12 - Team Leader A „Da geht einfach viel Zeit bei verteilten Teams drauf, das ist einer der größten Nachteile. Das alles einfach langsamer funktioniert und träger ist. Dadurch sinkt auch die Akzeptanz: Jetzt hab ich schon wieder ein Meeting, und den versteh ich mein gegenüber ja eh nicht, warum machen wir es dann überhaupt?“ [Delta #12 - Team Leader A]

Delta #13 - Team Leader A „Multikulturell finde ich nie schlecht, man lernt andere Sachen, man sieht auch andere Ausbildungen auf Universitäten. Also ich find das nicht schlecht, wenn man von unterschiedlichen Nationalitäten Leute zusammen mischt die Wissen zusammenführen.“ [Delta #13 - Team Leader A]

Delta #14 - Team Leader A „Aber wir haben auch immer: Welches Mikrofon nehmen wir, dann reisst auf einmal das WLAN wieder ab, dann ist ein Kabel wiedermal kaputt; das sind schon so viele Sachen wo viel Zeit drauf geht. Wo ich sage ich kann jetzt nicht in ein Meeting gehen und bau meinen Laptop auf und sehe alle, sondern da muss ich zuerst Skype aufmachen, dann anrufen, das sind auch jedesmal ein bis zwei Minuten und wenn du dann mal vier Meetings hast, hast du eigentlich schon eine Viertelstunde mit dem Starten und Verbindung aufbauen verbraucht.“ [Delta #14 - Team Leader A]

Delta #15 - Team Leader A „ Aber man sieht: es sind immer die selben Probleme, das Hauptproblem ist immer noch - und das im Jahr 2015 - die Infrastruktur. “ [Delta #15 - Team Leader A]

Delta #16 - Team Leader B „Meiner Meinung nach ist es sehr wichtig und wir versuchen diesen Punkt genau abzudecken, wir haben immer Winter-Week und Summer-Week wo wir die ganze

Firma zusammenbringen in Linz in der Zentrale. Wir treffen einander und wir sprechen miteinander direkt und es bringt uns immer auch sozial näher. Bis jetzt hat das nur positive Wirkung gebracht. “ [Delta #16 - Team Leader B]

A.2.5 Epsilon

Epsilon #1 - Department Manager „Wir haben auch Wissenstransfer Meetings die ich einberufe, auch online, um aus den verschiedenen Projekten die Informationen zusammentragen, wo jeder mal einen Vortrag macht - auch einen online - zu dem was gerade jetzt wichtig ist, das kann alles mögliche sein.Das funktioniert mit standard Tools wenn die Verbindung passt.“ [Epsilon #1 - Department Manager]

Epsilon #2 - Department Manager „Was wir noch verwenden ist der Sharepoint, da haben wir unsere eigene Seite, das muss gut strukturiert sein das man sich wiederfindet. Das man auch eben den Wissenstransfer so gestalten kann das die Leute darauf zugreifen können und sehen können was sie brauchen.“ [Epsilon #2 - Department Manager]

Epsilon #3 - Department Manager „Es war mal so, dass wir am Anfang sofort mit Scrum starten wollten und dort übersehen haben das wir innerhalb der Sprintdurchführung von äußeren Faktoren sowas von abhängig waren das wir es nicht mehr selber steuern konnten. Ein Scrum Team sollte in sich abgeschlossen sein, wenn noch ein anderer Anbieter im Boot ist, kann man nicht nachregeln, da hat sich das so nicht bewährt. Deswegen sind wir dann auf Kanban umgestiegen das war dann viel flexibler.“ [Epsilon #3 - Department Manager]

Epsilon #4 - Department Manager „Das ist ein ganz interessanter Punkt: Es hat sich gezeigt das oftmals die ganz einfachsten Mittel die besten sind. Ganz viel Kommunikation läuft, Sie werden es nicht glauben - nicht das wir eine Kamera daneben stellen und dann die Leute die anderen über den Bildschirm sehen, nein das läuft über den Chat. Ganz normalen uralten Chat. Wo man sich sofort reinhängt und der andere antwortet und den ganzen Tag lang hat man da so einen Kommunikationsstrang. “ [Epsilon #4 - Department Manager]

Epsilon #5 - Department Manager „Das liegt daran, dass das einfach wunderbar nebenher gemacht werden kann. Video reden ist immer etwas das eine direkte Interaktion bedeutet. Das kann hinderlich sein, oder man fühlt sich beobachtet und möchte das nicht. Ein Chat hat sich so für diese schnelle, kurzzeitige Kommunikation als sehr praktikabel erwiesen.“ [Epsilon #5 - Department Manager]

Epsilon #6 - Department Manager „Was ein Tool braucht: Stabile Verbindung, leicht konfigurierbar, wenn das hakt und die Video- und Audioqualität schlecht ist fällt das durch den Rost. Bildschirm teilen muss möglich sein, das ist ganz essentiell. Wir haben zum Beispiel das zwei Leute an unterschiedlichen Standorten an der gemeinsamen Aufgabe arbeiten und da müssen die ihren Bildschirm teilen können, damit der eine sieht was der andere gerade macht. Wenn das nicht geht gibt es ein Problem. Aber die Tools sind da mächtig genug und unterstützen das, dass es da meiner Ansicht nach keine Probleme gibt.“ [Epsilon #6 - Department Manager]

Epsilon #7 - Department Manager „Man kann bei einem Scrumboard entweder eine Kamera aufstellen und dann macht man es mit Zetteln, oder man macht es online. Ich bin eher ein Verfechter der Zettelmethode, aber man kann das auch online machen, da sehe ich eigentlich keine Abweichungen aufgrund der Tatsache das es ein verteiltes team ist.“ [Epsilon #7 - Department Manager]

Epsilon #8 - Department Manager „Face to Face Meeting ist bei gewissen Gruppenaktivitäten unabdinglich, zum Beispiel wenn man Workshops zusammen macht, wenn man ein Projektreview macht. Es ist auch so, dass sich ein Team überhaupt findet und sich ein Spirit aufbaut ist Face to Face ab und zu definitiv notwendig. Sich einmal kennenlernen und mal sehen, da findet eine andere Art von Kommunikation statt, das ist einfach so.“ [Epsilon #8 - Department Manager]

Epsilon #9 - Department Manager „Das wichtigste Kommunikationsmittel ist die Sprache und wenn die nicht passt, wenn man sich dort nicht sehr gut ausdrücken kann hat man ein Problem. Wenn Sie halt länderübergreifend sind, haben sie Sprachbarrieren. Das zieht in einem verteilten Team die Effizienz massiv in den Keller, das ist einfach so. Wenn da ein Team Meeting ist und man tauscht sich aus und man versteht sich nicht richtig dann führt das zu Problemen und Missverständnissen. Da ist es wichtig, dass alle ein gleich gutes Level haben, egal welche Sprache, aber wenn es da Unterschiede gibt dann wird das schwierig und führt zu Reibungsverlusten.“ [Epsilon #9 - Department Manager]

Epsilon #10 - Department Manager „Es kommt ganz wesentlich auf die Führungskraft an, und das ist ein zwischenmenschlicher Aspekt, manche können es und manche nicht. Da helfen die besten Tools nicht. Man kann alles so machen wie ich es jetzt gesagt habe und es funktioniert aber trotzdem nicht. Diese Sensorik zu haben, ein Team relativ kurzzeitig zu besuchen, Stimmungen einzuholen und weiterzugeben zu einem gewissen Maß. Da muss man ein gewisser Multiplikator sein und das ist gerade in verteilten Teams schwierig.“ [Epsilon #10 - Department Manager]

Epsilon #11 - Department Manager „Man muss also da als Führungskraft deutlich mehr investieren als wenn man das in einem in sich geschlossenen Umfeld macht. Ich bin zum Beispiel als Führungskraft mindestens alle 14 tage an allen Standorten ständig präsent, ich bin ständig am rumreisen.“ [Epsilon #11 - Department Manager]

Epsilon #12 - Department Manager „Es gibt noch ein Impediment das bei uns als verteiltes Team eine Rolle spielt: das ist die Infrastruktur in Europa innerhalb der Mobilfunknetze. Das ist für mich als Führungskraft, als Team und als Dienstleister ein ganz wesentliches Impediment, aber sowas von! Das hängt damit zusammen, im Team sind wir sehr viel mit der Bahn unterwegs und nützen aber auch die Reisezeit als Arbeitszeit und viele Leute können und würden gerne fürs Projekt arbeiten, können aber nicht. Das ist ein absoluter Skandal.“ [Epsilon #12 - Department Manager]

Epsilon #13 - Department Manager „ Die Herausforderung darin liegt darin, dass wir sehr gute kommunikative Prozesse benötigen. Also auch Leute die bei uns anfangen müssen über gute kommunikative Skills verfügen, man muss also in der Lage sein, möglichst frühzeitig Probleme zu erkennen, man muss in der Lage sein einen Teamspirit aufzubauen, auch wenn die Leute sich nicht sehen.“ [Epsilon #13 - Department Manager]

Epsilon #14 - Department Manager „Ich glaube das hat sich mittlerweile gewandelt, die Leute kriegen alles mit weil sie ja miteinander chatten - und das ist genauso wie zurufen. Sogar besser weil man da nicht gestört wird, man kann nachfragen wie man will und nicht dann wenn man zuhören muss. Das ist definitiv ein Vorteil.“ [Epsilon #14 - Department Manager]

A.2.6 Zeta

Zeta #1 - Department Manager „Das ist auch mein Credo, gerade in den großen Unternehmen, die tendieren dann so alles über einen Kamm zu scheren auch wenn es gar keinen Sinn macht. Gerade wenn ich jetzt denke an ein Team das sich mit Wartung beschäftigt: Ich habe ein stabiles Produkt das ist beim Kunden draussen, das wird aus den verschiedensten Gründen nicht mehr weiterentwickelt, da findet keine Produktentwicklung mehr statt sondern da gehts nur mehr um Wartung. Bugs beheben, minimalste Kundenanforderungen umzusetzen,... aber in Wahrheit steht das Produkt. Da setze ich sicher nicht Scrum ein, da setze ich viel eher Kanban ein. Weil der Overhead einer Produktentwicklung den ich bei Scrum voraussetze der ist da nicht mehr gegeben beziehungsweise da ist dann soviel Overhead da, das brauch ich dann nicht.“ [Zeta #1 - Department Manager]

Zeta #2 - Department Manager „Gerade bei agilen Methoden ist es einfacher weil ich die Kontrolle schneller habe. Weil ich mein Daily Standup habe und dadurch das ich die Arbeit in kleine Teile herunter breche sehe ich den Fortschritt viel schneller. Also von einem Kontrollaspekt her, ich habe auch die Rolle als Projektleiter gehabt wo ich sieben bis acht Teams koordiniert habe, dort war es in der agilen Welt sehr einfach weil ich sehr schnell den Fortschritt sehe, wenn sich die Stories bewegen durch den einzelnen Status durch.“ [Zeta #2 - Department Manager]

Zeta #3 - Department Manager „Ein team ein Backlog, das ist eine Grundregel. Pro Team ein Scrum Master, es gab aber wenn es sehr stabile Teams sind die schon lange zusammen sind brauchen die nicht einen full time Scrum Master, die organisieren sich dann selber sehr sehr viel. Bei uns gerade im Backend Bereich da gabs Teams - eine sehr erfahrene Scrum Masterin hatte drei Teams, unter anderem auch zwei davon in Polen, die waren alle sehr erfahren und sehr eingespielt. Da reicht ein Product Owner für drei Teams. “ [Zeta #3 - Department Manager]

Zeta #4 - Department Manager „Vor allem wenn es in Richtung Build Status geht, ’build is broken’, und da leuchtet es rot, sowas gibts immer mehr, das würde ich nicht unbedingt auf verteilte Teams reduzieren. Das ist auch sehr sehr hilfreich generell wenn das Team wie in einem Raum wie hier zusammensitzt. Weil es ein bisschen das Teamgefüge fördert, nach dem Motto wir können es uns nicht leisten das es da rot blinkt. In verteilten Teams kann ich mir das aber auch sehr gut vorstellen.“ [Zeta #4 - Department Manager]

Zeta #5 - Department Manager „Wenn ich sage ich hab einen Teil des Teams in Wien und einen Teil des Teams in Polen, dann ist es okay. Verteile ich das über mehr als zwei Standorte dann ist der Kommunikationsoverhead so riesengroß, dass mir die ganze Verteilung nichts bringt. Das müsste mir dann wirklich einer vorrechnen was es dann an Kostenersparnis wieder bringen würde weil die Kommunikation so an Überhand gewinnt - also der Overhead der dadurch entsteht - das es nichts bringt.“ [Zeta #5 - Department Manager]

Zeta #6 - Department Manager „Das waren die verschiedensten Kanäle, das hat angefangen die Daily Standups über Lync zu machen, also den Communicator von Microsoft. Dann gabs für

die größeren Abstimmungsmeetings, wie eine Sprintplanung oder Reviews oder Retros gabs dann Videokonferenzräume und wenn das zum Beispiel mal was ausgefallen ist gabs auch das gute alte Telefon. Und für gewisse Zwecke gab es noch immer auch das Reisen. “ [Zeta #6 - Department Manager]

Zeta #7 - Department Manager „Sehen ist immer besser als nur Hören. Weil über das Gehörte kommt bei weitem nicht soviel rüber als über das Gesehene. Wenn ich jemanden sehe, wie der dasitzt, wie dem seine Körperhaltung ist, wie der drauf ist, dann ist das etwas was keiner verbergen kann. Wenn ich den nur über Telefon oder Audio höre ist das ganz was anderes.“ [Zeta #7 - Department Manager]

Zeta #8 - Department Manager „Darum ist es mir auch ganz wichtig bei den Daily Standups das Bild zu sehen bei Videokonferenzen und nicht nur die Audio Spur, weil auch da kann ich sehr viel ablesen.“ [Zeta #8 - Department Manager]

Zeta #9 - Department Manager „Ich denke es ist ganz ganz wichtig das sich die Leute auch kennen lernen - Face to Face - und sich wirklich die Hand schütteln. Das ist essentiell denk ich, gerade wenn die Zusammenarbeit funktionieren soll und wenn ein Teamgefüge entstehen soll.“ [Zeta #9 - Department Manager]

Zeta #10 - Department Manager „Wenn die Zusammenarbeit gestartet hat, hat man versucht die Leute mal generell für eine gewisse Zeit nach Wien zu holen. Das ist egal ob da in Polen ein Team aufgebaut wurde, in der Slowakei oder in Indien. Man hat einmal versucht das Kernteam nach Wien zu holen, das dort aufgebaut werden soll und die geben das Wissen dann in der Organisation lokal weiter. Das waren große Handover Sessions.“ [Zeta #10 - Department Manager]

Zeta #11 - Department Manager „Dort wars so das man einmal im Monat für drei bis vier Tage das Team aus Polen oder die Teams aus Polen eingeflogen hat. Dann wurde gemacht: Sprint Review, Retro, und am nächsten Tag dann Sprint Planning und am nächsten Tag dann Sprint Planning zwei, so hat man es auf zwei bis drei Tage dann aufgeteilt, also für den nächsten Sprint dann, und hat dann gleichzeitig die Tage und Abende genutzt für soziale Events. Es gab zum Beispiel die Brettspiel Events am Abend, da hat man sich zusammengesetzt in der Firma, irgendjemand hat ein Brettspiel organisiert, Bier und Pizza kommen lassen um so ein bisschen ein Socializing zu erzeugen, um ein Teamgefüge zu erzeugen. Das hat so ganz gut funktioniert. “ [Zeta #11 - Department Manager]

Zeta #12 - Department Manager „Auf der anderen Seite ist es so, das es schon modelle der zusammenarbeit in verteilten teams gibt die funktionieren können, gerade wenn ich jetzt denke an länder wie polen, an länder wie die slowakei, die uns vom kulturellen aspekt her sehr ähnlich sind. Dort hat meiner Erfahrung nach die Zusammenarbeit immer am besten funktioniert. Weil gerade der kulturelle Aspekt ein sehr wesentlicher Aspekt ist in der Zusammenarbeit und Agilität schreibt sich auf die Fahnen, in einem team cross funktional zu funktionieren, und ein Team funktioniert nur dann wenn ich auch die kulturellen Aspekte beachte.“ [Zeta #12 - Department Manager]

Zeta #13 - Department Manager „Die Technik ist mittlerweile schon so ausgereift das es eine technische Hürde für mich nicht gibt, Videokonferenz ist billig, jeder hat Skype, gerade innerhalb

europa, ein Flug nach polen, nach Deutschland ist super billig, das ist nicht das Problem.“ [Zeta #13 - Department Manager]

Zeta #14 - Department Manager „Wenn ich Wertschätzung entgegenbringe, mal dorthin reise, mir anschaue wo arbeiten die, wie arbeiten die, in ein lokal auf ein bier gehen dann ist das ganz was anderes und führt zu besserer Arbeit auch in verteilten Teams.“ [Zeta #14 - Department Manager]

Zeta #15 - Department Manager „ Kommunikation ist schon ein essenzieller Part, sehr introvertierte Menschen tun sich in einem verteilten Team noch schwerer. Ich denke schon, dass es notwendig ist, das es mindestens ein zwei Personen auf jeder Seite des Teams gibt die kommunizieren können, wenn ich da lauter introvertierte Menschen hab dann wird es noch schwieriger. Ich sag mal vom technischen Skillset her ist es irrelevant, ein guter Entwickler ist ein guter Entwickler, da gibts keine speziellen Anforderungen vom technischen Standpunkt her. Eher die Kommunikation ist wichtig, und da sollte man schauen das man kommunikativere Leute hat gerade wenns in verteilte Teams geht. und die auch nicht davor scheuen auch mal zu verreisen.“ [Zeta #15 - Department Manager]

A.2.7 Eta

Eta #1- Scrum Master „Agile Vorgehensweisen jetzt verteilt zu machen ist es immer noch ein riesen riesen Vorteil gegenüber klassischen Vorgehensweisen. Ich hab auch sehr viele erfolgreiche Wasserfall Projekte gemacht, aber Dinge kommen einfach bei der agilen Vorgehensweise viel viel schneller hoch, auch wenn man verteilt arbeitet. “ [Eta #1- Scrum Master]

Eta #2- Scrum Master „Wir arbeiten hauptsächlich mit drei Wochen sprints, auch teilweise mit zwei Wochen Sprints. Meine Erfahrung ist, zwei Wochen Sprints in verteilten Teams sind manchmal eine kommerzielle Hürde. Muss man einfach sagen, wenn Leute dann viel reisen müssen, wenn man Leute auf drei Standorten sitzen hat. Auch wenn man sagt zwei Wochen Sprints, es reichen sechs Stunden für den Sprintwechsel, ein Tag ist ’gone’ sag ich mal. Darum hat sich bei uns der drei Wochen Sprint sehr gut eingeschwungen.“ [Eta #2- Scrum Master]

Eta #3- Scrum Master „Auf der einen Seite ist unser Ziel natürlich schon Experten und Topleute zu haben, aber unser Ziel ist es in einem Team nicht einen Experten für das und einen Experten für das zu haben, sondern die Teams sollen in der Lage sein, überall hin zu greifen damit man keine Flaschenhälse erzeugt. Deswegen ist das Thema Pair Programming ganz ganz wichtig, ist aber ein Thema das in der Eigenverantwortung des Teams liegt.“ [Eta #3- Scrum Master]

Eta #4- Scrum Master „Unsere Herangehensweise ist ganz einfach: wenn Dokumentation wichtig ist dann muss es Teil der User Story sein.“ [Eta #4- Scrum Master]

Eta #5- Scrum Master „ es hat sich gezeigt, dass durch diese 15 Minuten und sozusagen das Tooling das wir da verwenden wo jeder auch am Monitor sein Whiteboard sieht hat sich das bei den meisten - nicht bei allen Projekte - bei den meisten Projekten so eingebürgert, dass man eine Telefonkonferenz macht und das vom Platz aus macht. Es gibt ein paar Teams bei uns die machen es auch auf Video, aber der größere Teil macht nur auf Audio. “ [Eta #5- Scrum Master]

Eta #6- Scrum Master „Groomings - das sind die Meetings wo es darum geht Vertiefungen, vorbereitende Klärungen, für den nächsten Sprint zu machen, die versuchen wir wenns möglich ist über Video zu machen. Darum haben wir einerseits auf allen unseren Standorten ein sehr gutes Video Konferenzsystem wo Bild und Ton sehr gut sind, wo Screensharing möglich ist, und in den meisten Fällen haben wir das auch beim Kunden. Das heißt auch wenn es drei Standorte sind, das zusammenschalten per Video kein Thema ist. “ [Eta #6- Scrum Master]

Eta #7- Scrum Master „Die Retrospektive ist ein ganz ganz wichtiger Teil bei uns der auch sehr ernstgenommen wird, weil auch das Verbesserungen anspricht. Retros verteilt machen wir auch manchmal, aber v.a. bei der Retrospektive ist es einfach ganz ganz gut wenn man Face to Face sitzt.“ [Eta #7- Scrum Master]

Eta #8- Scrum Master „Beim Sprintwechsel selbst, das ist üblicherweise bei uns ein Tag wo man sich zusammenfindet, ist das Ziel das immer Face to Face zu machen, das alle Leute an einem Standort zusammenkommen. Ein Weg ist da das wir das rotieren auf den unterschiedlichen Standorten, einmal beim Kunden, einmal bei uns bei dem Standort, einmal bei dem Standort. Wichtig ist, wenns möglich ist, es face to face zu machen, klappt natürlich nicht immer.“ [Eta #8- Scrum Master]

Eta #9- Scrum Master „Das ist das Thema Projektkommunikation, wie klar werden Dinge kommuniziert, oder sind sie mangelhaft aufgrund mangelhafter Sprachkenntnisse, das kommt absolut vor. Wenn eine Drittsprache wie Englisch als Kommunikationssprache verwendet wird klappt es sehr gut wenn der Level bei allen Kollegen sehr hoch ist, aber das kann man fast nicht gewährleisten.“ [Eta #9- Scrum Master]

Eta #10- Scrum Master „Scrum funktioniert sehr viel aus dem Teamgedanken und aus dem Teamformungsprozess heraus. Natürlich wenn man neue Teams zusammenstellt wenn die verteilt sind dauert es einfach länger bis die Teamformungsprozesse da durchlaufen. Die andere Geschichte ist da die Fähigkeit Probleme zu lösen, im Team zu lösen, ist im Verteilten einfach schwieriger.“ [Eta #10- Scrum Master]

Eta #11- Scrum Master „Wenn Skype schlecht ist einem Daily zu folgen, wenn man vorm Monitor sitzt und der Ton is schlecht bei zwei oder drei Kollegen dann driftet man ab, man kann nicht folgen. Deswegen ist saubere Infrastruktur ganz ganz wichtig wenn man sich für sowas entscheidet und man darf daran nicht sparen. Und generell viele Möglichkeiten zu bieten.“ [Eta #11- Scrum Master]

Eta #12- Scrum Master „Viel wichtiger für uns ist, Kollegen und Kolleginnen zu bekommen die bereit sind in Teams zu arbeiten, die bereit sind wissen zu Sharen eben das Thema Pair Programming. Was auch nicht selbstverständlich ist, bereit sind in Retros Dinge anzusprechen und oft auch Dinge zu ertragen und Feedback zu bekommen was Verbesserung betrifft. Das ist für uns ganz eine wichtige Geschichte. Aber es gibt keine Anforderungen die jetzt sagen weil wir verteilt sind müssen wir auf was spezielles schauen.“ [Eta #12- Scrum Master]

Eta #13- Scrum Master „Auf der einen Seite bin ich bei größeren Projekten sowas wie Scrum Master - in Richtung Kunde vielleicht Projektmanager - aber in der agilen Sprache ist es Scrum

Master, vielleicht in etwas abgeänderter Eigenschaft. Und ich mache sehr viel an der Kundenschnittstelle. Mein Job da ist Customer Relationship Manager.“ [Eta #13- Scrum Master]

Eta #14- Scrum Master „Unterwegs wähle ich mich in kein Meeting ein! Vielleicht kurz wenn mich wer anruft, aber generell, man weiß auf der Wegstrecke fällt zwei bis drei mal das Internet aus, man kann nicht vernünftig teilnehmen. “ [Eta #14- Scrum Master]

Eta #15- Scrum Master „Wichtig ist, wenn ich den Optimalzustand nicht habe, wie ein gemeinsamer Raum und die sehen sich immer, dann muss ich das was ich nicht habe wenn Leute verteilt sitzen durch andere Dinge versuchen wieder zu lösen. Wo wir sagen: Wenn verteilt, dann Sprintwechsel zusammen.“ [Eta #15- Scrum Master]

A.2.8 Theta

Theta #1 - Agile Coach „Mit Kanban brauchst du viel öfter einen Sync, mit Scrum kannst du das machen das du sagt: ’Ok, jetzt setzen wir uns alle zwei Wochen wirklich physisch zusammen und diskutieren das aus.’ Sonst machst du halt die täglichen Syncs und man muss nur dann eine dedizierte session machen wenn etwas aufkommt.“ [Theta #1 - Agile Coach]

Theta #2 - Agile Coach „Wie ich zur Firma gekommen bin hatten sie schon diese Cycle, und ja, man hat natürlich schon immer noch was verbessern können aber es war auch sozusagen vorgegeben und wir haben gute Erfahrungen damit gemacht. Wir machen kein Scrum out of the Book, ich denke das macht keiner, und für mich ist es auch keine Bedingung. Wir fokussieren darauf das wir agil sind, wir nutzen einige Tools von Scrum, wir nutzen aber auch einige Tools von Kanban, und wir streben aber auch in die Richtung automatisierter Rollout Prozess und haben Continuous Delivery anvisiert, aber sowas passiert natürlich nicht über Nacht. Generell kommen wir mit Scrum gut zurecht.“ [Theta #2 - Agile Coach]

Theta #3 - Agile Coach „Wenn andere mit einem Stück Code fertig sind dann erstellt man ein Code Review mit zum Beispiel Fisheye, das ist so ein Tool von Atlassian. Oder man macht einen Pull Request und setzt die anderen drauf und sie bekommen eine Benachrichtigung und reviewen den Code und machen Kommentare oder rufen die Person an. “ [Theta #3 - Agile Coach]

Theta #4 - Agile Coach „Wir haben nach einer Zeit bewusst darauf geschaut, dass das Code Review von unterschiedlichen Standorten gemacht wird. Vor allem wenn wir Kollegen in Russland gehabt haben, das war eine ziemlich komplexe Herausforderung, hat aber ziemlich gut funktioniert. “ [Theta #4 - Agile Coach]

Theta #5 - Agile Coach „Deswegen haben wir dort eingeführt, wenn etwas im Code geändert wird, muss immer wenigstens einer vom anderen Team beim Code Review dabei sein. Damit sie es zumindest passiv mitbekommt was sich geändert hat. “ [Theta #5 - Agile Coach]

Theta #6 - Agile Coach „Für die dynamischeren Sachen nutzen wir einfach Jira. Wir schauen das wir alle unsere Aufgaben grundsätzlich dort abbilden, weil sonst hast du halt extra Aufwand das alles zu synchronisieren. Also jedes Team hat einen Teamchat, das zählt vielleicht auch als Radiator.“ [Theta #6 - Agile Coach]

Theta #7 - Agile Coach „Die meisten Best Practices von Scrum haben wir irgendwie gemeistert das sie funktionieren, auch wenn wir manchmal gesagt haben das geht nicht ausser wenn die Leute zusammensitzen. Das einzige ist was ein bisschen mühsam ist, wir machen in einem Team die Estimations mit Planning Poker, da ist es halt ein bisschen komisch wo die Leute halt dann den Wert eintippen, aber das ist jetzt einfach so, das ist jetzt kein Blocker.“ [Theta #7 - Agile Coach]

Theta #8 - Agile Coach „Wir haben ein Gentlemans Agreement und zwar jeder schaut grundsätzlich alle halbe bis Dreiviertelstunde ins Skype damit es nicht ständig zu Unterbrechungen kommt aber auch nichts untergeht. “ [Theta #8 - Agile Coach]

Theta #9 - Agile Coach „Die Leute sitzen nebeneinander wenn auch nur virtuell. Also wenn sie miteinander aktiv arbeiten, würdest du jetzt mit einem Kollegen aus Budapest arbeiten, ist meistens ein Chat offen oder Hangout oder ein Call. Wir nutzen das wirklich extensive, die Leute hängen fast die ganze Zeit im Chat. “ [Theta #9 - Agile Coach]

Theta #10 - Agile Coach „Es passiert immer wieder, das Leute parallel Diskussionen führen und wenn du jetzt in einem Meeting bist wo du was ausdiskutierst wird das nicht stören. Aber mit einem online Call hast du die anderen gleich verloren. Das ist immer eine spannende Balance zu halten wie sehr schaut man auf die anderen damit sie nicht ganz abgeschnitten werden aber wieviel Zeit will man damit verschwenden diese Diskussion absichtlich auseinander zu halten.“ [Theta #10 - Agile Coach]

Theta #11 - Agile Coach „Wir haben jeden zweiten Montag Sprintplanning und da fahren die Leute dann meistens nach Wien. Das ist einfacher, das ist historisch so gewachsen, früher wie das Team mehr ausbalanciert war haben wir es immer abwechselnd gemacht, dass das Team nach Linz gefahren ist. Wie lange sie bleiben hängt halt davon ab ob es vom Sprint unabhängige Architektur Diskussion gibt, das dort auch ausdiskutiert dann.“ [Theta #11 - Agile Coach]

Theta #12 - Agile Coach „Meine Präferenz ist, und soweit ich das gesehen hab ist das auch bei vielen anderen so, das komplexere Themen gerne Face to Face ausdiskutiert werden. Vielleicht gibts auch schon gute Tools, nur haben wir die noch nicht gefunden beziehungsweise haben auch noch nicht danach gesucht. Aber ich höre auch von anderen Kollegen oft: 'Ja das besprechen wir dann lieber in zwei Tagen wenn du da bist'. Viele Kollegen machen es dann auch so, dass sie ein oder zwei Tage in der Woche fix an den anderen Standorten sind.“ [Theta #12 - Agile Coach]

Theta #13 - Agile Coach „Ich hab schon mal versucht remote Planning Sessions zu machen, das war nicht hier sondern in meiner früheren Firma und der Aufwand ist nicht vertretbar. Du hast einfach soviel Kommunikationsoverhead, so viele Missverständnisse und Rückfragen, die Effektivität ist nicht vergleichbar. Also einfach so stupide Aufgaben das man den Backlog durchgeht und sagt: 'Ok, wo ist der Scope?', allein schon diese Diskussion wenn es ein bisschen ausufert, dann brauchst du drei mal soviel Zeit.“ [Theta #13 - Agile Coach]

Theta #14 - Agile Coach „Es ist sehr wichtig, dass die Leute ein ähnliches Niveau von Englisch sprechen. Weil auch wenn du jetzt Kollegen hast mit denen du generell gut auskommst, im Sinne von du bist nicht mehr dominant, ich bin nicht mehr dominant und du kannst mit dem gut diskutieren, aber wenn das jetzt plötzlich Englisch wird und eine Person kann sich viel besser ausdrücken, führt das einfach zu einer unbalancierten Situation. Bei Face to Face ist das nicht

so schlimm, weil du erkennst den Gesichtsausdruck. Du hast die Geduld: 'Ok jetzt sitzt er mir gegenüber und er braucht 10 Sekunden den Satz zusammen zu setzen und ich warte es ab'. Aber wenn das remote ist und der andere die Wörter sucht siehst du ja nicht unbedingt sein Gesicht." [Theta #14 - Agile Coach]

Theta #15 - Agile Coach „Grundsätzlich was ein unterlegendes Problem ist, ist die Kommunikation von tagtäglicher Arbeit aber auch von zwischenmenschlichen Beziehungen. Ich würde nicht sagen Vertrauensproblem weil das stimmt nicht, weil man arbeitet miteinander jahrelang zusammen dann ist das Vertrauen da, aber es ist schon ein bisschen anders." [Theta #15 - Agile Coach]

Theta #16 - Agile Coach „ Wenn du eine Person hast und der Rest des Teams denkt: 'Ok wir sind mehr oder weniger vollzählig hier im Raum' und es gibt eine Diskussion wo es ziemlich schnell hin und her geht, und die andere Person online kommt nichtmal zum Wort. Wenn die Hälfte da und die andere da ist, wird wirklich dann gesagt: 'Ok das ist unsere Meinung, was denkt ihr?', dann ist es nicht so fließend wie eine Roundtable Diskussion, aber trotzdem, es wird mehr acht darauf gegeben." [Theta #16 - Agile Coach]

Theta #17 - Agile Coach „Inzwischen macht das keine großen Probleme mehr. ich kann mich erinnern in der früheren Zeit hat es manchmal gewackelt und manchmal haben wir 10 bis 15 Minuten warten müssen bis wir irgendwie eine Connection aufbauen haben können. Das war ja der ganze Zeitraum von den Daily Meetings dann war es ziemlich irritierend. In letzter Zeit ist das wirklich schon die Ausnahme." [Theta #17 - Agile Coach]

Theta #18 - Agile Coach „Jetzt machen wir zum Beispiel auch so Teambuilding Workshops wo genau das Kommunikationsziel das ich vorher angesprochen habe adressiert wird und dort holen wir einfach die Leute gemeinsam zu einem externen Standort. " [Theta #18 - Agile Coach]

Theta #19 - Agile Coach „Das sie halt kommunikativ sind, oder wenigstens offen kommunizieren wollen und damit hat sich die Sache eigentlich schon. Weil wir haben schon von unseren Kollegen in Budapest als Feedback bekommen: 'Leute ihr müsst einfach an eurer Kultur arbeiten beim Teleconferencing'. Und ja, du musst außer Englisch und programmieren können nur ein netter Mensch sein." [Theta #19 - Agile Coach]

Theta #20 - Agile Coach „Der Punkt warum ich denke das agile Methoden sehr hilfreich sind in verteilten Teams ist, dass der Fokus grundsätzlich auf einer offenen kurzfristigen Kommunikation liegt, also Kommunikation mit kurzen Schleifen. Was ich gesehen habe in verteilten Teams ohne kurze Kommunikations-Iterationen ist, dass du viel leichter voneinander abweichst." [Theta #20 - Agile Coach]

Theta #21 - Agile Coach „Ich bin in der Firma seit Beginn als Scrum Master tätig, in letzter Zeit hat sich die Rolle so als Agile Coach entwickelt. Vorher war ich schon auch Scrum Master und ich arbeite nun fast mit allen Teams die Software entwickeln." [Theta #21 - Agile Coach]

A.2.9 Iota

Iota #1 - Team Leader „Es gibt einen ganz klar allgemeinen Trend hin zu agilen Methoden und es gibt einen Trend hin zu flexiblerer Arbeitsgestaltung und remote Arbeit. Diese Entwicklung hat aber in den letzten 10 Jahren parallel stattgefunden. Aus meiner Sicht gibt es da keinen konkreten Grund warum man das eine nicht mit dem anderen kombinieren könnte.“ [Iota #1 - Team Leader]

Iota #2 - Team Leader „ Bei meinem Kanban ist es so, es gibt ein Kanban Board, es gibt ganz einfache Priorisierung und wir haben den Softwareentwicklungsprozess parallel zum Management, ich unterscheide da noch zwischen dem Softwaremanagement oder Produktmanagement Prozess, also derjenige der sich darum kümmert alle beteiligten oder alle stakeholder irgendwie auf einen Wissensstand zu bringen: Was ist gerade wichtig, was wird getan, wo stehen wir. Dazu läuft eben parallel die Implementierung und beides ist insofern nur gekoppelt das einfach klar wird was wird als nächstes erledigt. Aber wir können jeden tag releasen, wir können jeden Tag Bugfixes rausbringen, und das ist für mich wichtiger diese Geschwindigkeit und ein bisschen diese losere Kopplung als diese harten Check-Ins und Abnahmen und so weiter.“ [Iota #2 - Team Leader]

Iota #3 - Team Leader „Da gibt es praktisch eine Produktretrospektive - einmal die Woche - und ich mach eine größere Retrospektive für die IT an sich so alle zwei Monate in so Workshops wo dann alle Entwickler zusammenkommen. Wir reden da dann generell mehr über den Prozess als ganzes, wo fühlen wir uns wohl, wo sehen wir Probleme, wo sind einfach Dinge bei denen mehrere Leute das Gefühl haben da müssten wir ein größeres Refactoring angehen. Weil wenn man sich nur auf Features und so fokussiert dann ist es klar das man oft so technische Schulden mitbekommt aber jeder trifft das sich so für sich selbst, und wenn man sich ein bisschen Zeit nimmt und länger über das Thema spricht dann entstehen auch neue Erkenntnisse in der Gruppe dadrüber welche Themen tatsächlich irgendwie kritisch sein könnten.“ [Iota #3 - Team Leader]

Iota #4 - Team Leader „Code Reviews benutzen wir immer, nämlich um einfach eine gewisse Form von Qualität sicher zu stellen und auch sicher zu stellen das es einen Wissenstransfer zwischen den Mitarbeitern gibt. Immer wenn jemand ein Feature einbaut gibt es Pull Requests in der Quellcodeverwaltung und da muss ein anderer Mitarbeiter drüberschauen, macht ein Coderview und macht dann einen gemeinsamen Videocall und dann sprechen sie über den Pull Request, geben sich ein bisschen Feedback und dann wird das gefixt und dann wird das von der anderen Person gemerged.“ [Iota #4 - Team Leader]

Iota #5 - Team Leader „Auch die Verantwortung wird da aufgeteilt, es gibt jemand der schreibt den Code und derjenige der den Pull Request reviewed ist derjenige der ihn merged und dadurch gibts auch niemanden der den Code besitzt. Für mich ist wichtig das es das Code Stewardship gibt, das heißt alle im Team sind für den kompletten Code verantwortlich. “ [Iota #5 - Team Leader]

Iota #6 - Team Leader „Generell ist es so, dass wir immer über einen so genannten Tech Channel den wir im Gruppenchat haben im Austausch stehen und wenn jemand was neues entdeckt oder was neues implementiert oder sich was ansieht das dann immer geteilt wird. Immer das Wissen was man gerade entdeckt das den anderen Kollegen mitteilt. Eine andere Form ist natürlich auch einen Blog Artikel zu schreiben, eine Präsentation zu machen, darüber zu berichten auch wenn man auf einer Konferenz war. Dafür reservier ich auch immer Zeit, das man auch über diese Themen redet.“ [Iota #6 - Team Leader]

Iota #7 - Team Leader „Wenn das ganze Team in einem Raum sitzt gibts einfach so eine Blase an Informationen die ständig aktualisiert wird, das hängt so im Raum das bekommt man so mit.

(...) Wenn man ein verteiltes Team hat dann passiert das halt nicht. Das heißt man muss diese Information auch aktiv verteilen, man muss sich darum kümmern das es alle mitbekommen und von daher ist es natürlich wichtiger von da drauf zu achten.“ [Iota #7 - Team Leader]

Iota #8 - Team Leader „Für mich sind digitale Tools einfach die natürliche Arbeitsweise, weil die Vorteile sind einfach für mich schlagend. Ich hab es erstmal überall, egal wo ich bin, auch unterwegs kann ich drauf zugreifen, es ist durchsuchbar, ich kann es massenweise editieren, ich kann bestimmte Dinge neu gruppieren. Also diese Möglichkeiten wie ich diese Daten manipulieren kann ist einfach viel viel komplexer und viel vielfältiger als wenn ich ein Board habe. Ich mag zum Beispiel auch virtuelle Kanban Boards, die genauso funktionieren wie das physische Board, also ich hab drei Spalten weil ich gemerkt habe es hängt eine zusätzliche Information da dran wo eine Karte genau hängt, das heißt ich kann es mir besser merken und ich hab einen besseren Überblick anstatt wenn ich nur eine Liste von Issues habe wie in klassischen Bugtrackern.“ [Iota #8 - Team Leader]

Iota #9 - Team Leader „Also Kommunikation ist das A und O und ich meine Softwareentwicklung ist ja auch nichts anderes als wir sprechen irgendwas und dann packen wir es in Worte, also Software sind ja auch Worte, ist auch wieder herunter geschriebene Kommunikation. Daran muss man einfach arbeiten.“ [Iota #9 - Team Leader]

Iota #10 - Team Leader „ Chat ist sehr unverbindlich einfach bezüglich wann kommt eine Nachricht an und wann wird sie gelesen. Das heißt es braucht immer ein weiteres Kommunikationswerkzeug in dem tatsächlich verbindlich festgelegt wird was wann wie gemacht wird. Das heißt da braucht es immer einen - ich nenn es mal Task Tracker ganz allgemein. Also irgendeine Form wo ich sagen kann da gibts eine Aufgabe, ich kann die irgendwem zuweisen, der andere kann sehen was ihm für Aufgaben zugewiesen sind und er kann dort den Status aktualisieren.“ [Iota #10 - Team Leader]

Iota #11 - Team Leader „Bei Text Kommunikation fehlt jede Menge an Zusatzinformation, die Bandbreite ist gering, also braucht man irgendeinen Sprachchat, also entweder Google Hangouts oder sowas, wo man Audio übertragen kann. Das ist extrem wichtig und dann braucht man noch eine Kamera, also das ist nicht entscheidend aber es ist extrem hilfreich in Diskussionen, auch um zu sehen wenn eine Telko stattfindet dann kann man sehen wenn jemand sprechen möchte und wenn jemand das Wort ergreifen möchte. Das funktioniert halt nicht wenn man die anderen nicht sieht. Das hab ich bisher in jeder Firma gesehen: also Chat, Tasktracker und irgendwas für Sprache. Früher war es viel Telefon, heute ist es vermehrter Videochat und so, Skype ist eigentlich die Regel.“ [Iota #11 - Team Leader]

Iota #12 - Team Leader „Es ist so, das du dein Gegenüber und die Motivation verstehen musst, auch seine Ängste kennen musst um seine Entscheidungen nachvollziehen zu können. Dieses Wissen darüber was das Gegenüber für Ängste und Bedürfnisse hat das ist etwas das kann man nur lernen wenn man sich regelmäßig in vielfältigen Begegnungen mit seinen Kollegen begibt. Und die außerhalb auch von einem Arbeits- und Auftragskontext sind. Das man zusammen auch essen geht und auch streitet und Probleme wälzt, und das funktioniert am besten eben physisch.“ [Iota #12 - Team Leader]

Iota #13 - Team Leader „Ich glaube das sozusagen die remote Arbeit mehr Vorteile bietet als Nachteile. Aber das es eben eine neue Kategorie von Problemen eröffnet und das die eigentliche

Schwierigkeit ist das zu erkennen. Das ist nicht einfach nur die Mitarbeiter sitzen woanders. Sondern das genau da andere Dinge passieren die einem erstmal per-se nicht so bewusst sind.“ [Iota #13 - Team Leader]

Iota #14 - Team Leader „ Was ich aber als vorteilhaft empfinde ist eben doch Leute aus anderen Kulturkreisen, aus anderen politischen Systemen, aus anderen wirtschaftlichen Systemen mit dazu zu bekommen weil die einfach komplett neue Aspekte mit rein bringen. Wie sie aufgewachsen sind, was für sie wichtig ist, wie sie Dinge empfinden, wie sie Text wahrnehmen, wie sie UI wahrnehmen. Diese ganzen Themen erfordern ein höheres Maß an Sensibilität im Umgang, wir Deutschen untereinander sind sehr direkt was in anderen Kulturkreisen gar nicht funktioniert, aber das schadet einfach nicht das man das sozusagen lernt. Und auch kulturelle Themen reinzubringen kann einfach einen zusätzlichen Blickpunkt auf Themen bringen die unter Umständen sehr wertvoll sind, was ich mir eigentlich Wünsche.“ [Iota #14 - Team Leader]

Iota #15 - Team Leader „Also das sie wirklich zwei bis drei Wochen, je nachdem wie komplex die Aufgaben sind, gemeinsam mit den Kollegen vor Ort sind. Das schafft schon extrem viel Nähe und viel Wissen über den anderen. Es ist eine wichtige Basis um dann später auch zu wissen wie reagiert er gerade oder warum macht er das gerade.“ [Iota #15 - Team Leader]

Iota #16 - Team Leader „Nicht jeder Mitarbeiter ist dafür geeignet. Es ist wichtig das man erstens ein hohes Maß an Selbstorganisation an den Tag legt, das man sich auch aktiv darum kümmert das man arbeiten kann ohne ständig auf Hilfe von Kollegen zurückzugreifen denn die sind ja auch nicht ständig verfügbar. Und zum anderen brauche ich einfach Mitarbeiter die das Ziel nicht aus den Augen verlieren, wo man konstant daneben stehen muss und sagen muss: geht das noch in die richtige Richtung, hast du daran und daran und daran gedacht. Das sind zwei Dinge die nicht jeder mitbringt.“ [Iota #16 - Team Leader]

Iota #17 - Team Leader „Ich habe die Erfahrung gemacht wenn es zwei Sprachen gibt über die übersetzt werden muss, zum Beispiel von Deutsch ins Englische ins Ukrainische, und dann wieder zurück, das ist sehr anstrengend. Weil es gibt zu beiden Sprachen immer unterschiedliche Bedeutungen von Wörtern und dann bringt man noch eine dritte Sprache ins Spiel und das macht es extrem aufwändig. Wenn man sowas hat würde ich drauf achten das etwas nicht zweimal übersetzt wird.“ [Iota #17 - Team Leader]

Iota #18 - Team Leader „Rein von den anderen Werkzeugen und Tools die wir haben um unsere Arbeit zu unterstützen sehe ich kein Problem. Die Auswahl ist riesig, die Entwicklung geht immer weiter, es gibt immer neue Tools. Da ist sicher noch Luft nach oben aber so für meinen Alltagsbedarf gibt es relativ wenig Probleme die irgendwie nicht gelöst sind.“ [Iota #18 - Team Leader]

Iota #19 - Team Leader „Das ist gerade sehr wichtig die Leute da zusammenzuhaben. Die Diskussionen sind ein bisschen länger, und man braucht mal ein Whiteboard und muss mal was aufmalen und muss mal anderen Leuten was erklären. Das das über Remote zu machen ist zu schwierig, das funktioniert nicht.“ [Iota #19 - Team Leader]