Tomi Makkonen

SUCCESS FACTORS IN DISTRIBUTED AGILE DEVELOPMENT: CASE STUDY

JYVÄSKYLÄN YLIOPISTO

TIETOJENKÄSITTELYTIETEIDEN LAITOS

2018

ABSTRACT

Makkonen, Tomi
Success factors in Distributed Agile Development
University of Jyväskylä, 2018, 56p.
Information Systems Science, Master's thesis
Supervisor(s): Kollanus, Sami

This thesis aimed to figure out success factors in distributed software development conducting literature review and empirical research. The motivation for this research rose from practical work experience and the notion that usage of agile development has increased same time as global distributed software development has become more common. The research question formed to investigate the topic was: "What are success factors in distributed agile development and what experiences about this combination already exists?" To answer research question, there were conducted literature review of existing literature and empirical case study research executed using theme interviews. In literature review used keywords were distributed development, agile development and distributed agile development. These keywords were covered in success and challenge perspectives and literature related to distributed development and agile development were used to support review because the lack of references related only to success factors of distributed agile development. In both, distributed development and agile development success factors, raised two similar factors. The first common factor is about teams and team members. Based on agile development team members should be competent and team should be high-caliber team and in distributed development the team integration and spirit were highlighted. The second factor is related to communication and effective communication technologies and tools. Based on studied empirical case, communication, trainings and team cohesion were agreed as success factors in distributed agile development. Also, correct roles for team members and technical tools were mentioned.

Keywords: Distributed development, Agile development, Distributed agile development, success factor

TIIVISTELMÄ

Makkonen, Tomi
Menestystekijät hajautetussa ketterässä ohjelmistokehityksessä
Jyväskylän yliopisto, 2018, 56s.
Tietojärjestelmätiede, Pro gradu
Ohjaaja(t): Kollanus, Sami

Tämän tutkimuksen tavoitteena oli selvittää mitkä ovat menestystekijöitä hajautetussa ketterässä ohjelmistokehityksessä, kirjallisuuskatsauksen ja empiirisen tutkimuksen avulla. Mielenkiinto aihealueeseen heräsi käytännön työkokemuksesta ja siitä että ketterän ohjelmistokehityksen käyttäminen on lisääntynyt samalla kun kansainvälinen hajautettu ohjelmistokehittäminen on myös yleistynyt. Tutkimuskysymys joka muodostui aihealueen tutkimiseen: ”Mitkä ovat hajautetun ketterän ohjelmistokehittämisen menestystekijöitä ja mitä kokemuksia tästä yhdistelmästä on?” Vastatakseni tutkimuskysymykseen, suoritettiin kirjallisuuskatsaus tehdyistä tutkimuksista ja kirjallisuudesta sekä empiirinen tapaustutkimus käyttäen teemahaastatteluja. Kirjallisuuskatsauksessa käytettiin avainsanoja kuten hajautettu kehittäminen, ketterä kehittäminen ja hajautettu ketterä kehittäminen. Näitä avainsanoja tutkittiin yhdistäen ne menestykseen ja haasteisiin ja kirjallisuuteen. Kirjallisuutta liittyen hajautettuun kehittämiseen ja ketterään kehittämiseen käytettiin tukemaan katsausta koska yhdistelmästä hajautettua ja ketterää kehittämistä ja menestystekijöitä ei ollut tarpeeksi olemassa olevaa tutkimusta. Molemmissa hajautetussa ja ketterässä kehittämisessä menestystekijöihin liittyen tuli ilmi kahdenlaisia menestystekijöitä. Ensimmäiset tekijät liittyivät tiimiin ja tiimin jäseniin. Perustuen kirjallisuuteen ketterästä ohjelmistokehityksestä tiimin jäsenten pitäisi olla pätevä ja korkealuokkainen tiimi ja hajautetussa ohjelmistokehityksessä tiimin integroitumista ja tiimin yhteishenkeä korostettiin. Toinen tekijä liittyy kommunikaatioon ja tehokkaisiin kommunikointiteknologioihin ja työkaluihin. Empiirisen tutkimuksen mukaan kommunikaatio, koulutukset ja tiimin yhtenäisyys olivat menestystekijöitä hajautetussa ketterässä kehityksessä. Myöskin oikea roolitus tiimin jäsenille sekä oikeat teknologiset työkalut mainittiin.

Avainsanat: Hajautettu kehittäminen, ketterä kehittäminen, hajautettu ketterä kehittäminen, menestystekjiä

FIGURES

TABLES

TABLE OF CONTENTS

1 INTRODUCTION

Year 2010, Freudenberg and Sharp (2010) summarized list of burning questions related to software development, based on feedback received from experts and practioners. Among other topics, there were listed distributed agile, large agile projects and self-organizing teams. In Agile2011 workshop covered question about which topics should be researched further in near future and again distributed agile was chosen as important subject. Researchers Jalali and Wohlin (2012) also stated that need for new research data about combining agile development and global software development is emergent.

Researchers and practioners keep subject important, business is more diverse than ever and software development is expected to be more responsive and agile than before. Combining agile and global development can be an answer, but there are also new challenges related to this combination. Key principles of agile software development are co-located, self-organizing small development team of experts, who communicate effectively with each other. Same time, globally distributed software team can be located in different countries and time zones. Hence, combining them is a challenge and in distributed agile development these two conflicting ways of working should be combined successfully.

I conducted systematic literature review to investigate and evaluate this topic in success factors' point of view. In systematic literature review idea is to identify, evaluate and present all available literature relevant to a specific research question. Aim is to provide background information to summarizing existing literature and finding gaps from existing research. The process of searching relevant literature starts with defining research question. After research question is defined relevant literature can be filtered. Keywords are distributed software development, agile software development and distributed agile development. I searched literature using one keyword at a time and in the beginning focused to find article which is quite recent and relevant as possible. Then if article was relevant also after reading it, I concentrated its references and tried to find other relevant articles based on titles and abstracts. I went all keywords through combining those with success factors and challenges too. Then I had literature divided into four categories; distributed development, agile development distributed development and success factors. Used databases were Google Scholar and AISeL.

This thesis aims to figure out success factors in distributed agile development project. This paper answers to research question: What are success factors in distributed agile development and what experiences about this combination already exists? I am going to answer to this question by doing systematic literature review using articles, handbooks and conference proceedings related to topics distributed software development, global software development, agile software development and distributed agile development. When researching literature about success factors in software development, most of the results consider success factors in software development generally. When searching specifically success factors in global software development or agile development literature can be found, but when searching literature about success factors in distributed agile development there is less literature available. Reason for this result is that combining agile development and distributed development is recently aroused way of working. Nevertheless, many global software development companies are moving towards agile development and at same time developing software geographically distributed, so future research data is needed.

This thesis has the following structure: in the introduction research process, topic and motivation for this study is presented. Second chapter is literature review about distributed development, agile development, distributed agile development in perspective of success factors and challenges in those. Third chapter is about empirical research, used research methodology, data gathering and data analysis. Fourth chapter is about results of the studied case and it combines distributed and agile development and presents experiences from the practice and produce table of the key success factors in distributed agile development. Fifth chapter is discussion where results of literature review and empirical research are compared and in the end, sixth chapter is about summary, conclusions and future research topics.

2 Literature review

This chapter is about distributed software development. In the beginning of the chapter I present citations which tell about the change that software development has faced. Second section is for identifying global software development, offshoring, outsourcing and distributed software development. Then idea is to research why usage of distributed software development has increased and then the last section focuses on challenges and success in distributed software projects.

2.1 Distributed software development

Globalization affects in many areas of living and information system development makes no difference. Boland and Fitzgerald (2004) states that: "Global software development has become an extremely important issue for organizations at present in the climate of increasing tendency towards globalization and global outsourcing." Also, other authors notice that companies all over the world are interested in global software development (Holmstrom et al, 2006). Damian, Lanubile, Hargreaves and Chisan (2004) present also challenges and states that increased popularity of global software development creates software engineering challenges impacting temporal, geographical and cultural differences. Also, Ågerfalk and others (2005) agree that distributed development is an issue of increasing significance for organizations nowadays, especially considering the current trend towards outsourcing and globalization. So, the change in business is changing software development too. Software companies try to respond to competition and develop new ways of working. To sum up, globalization is present also in software development and it is important to understand distances, differences and challenges to complete distributed software development projects successfully.

2.1.1 Definitions and background for distributed software development

In this section idea is to cover definitions for mostly used terms like distributed development, global software development and offshoring. The aim is also study why companies change to distributed software development. Following Table 1 gives understanding about words and definitions which have quite similar meanings regardless of the used word.

Table 1 - Definitions for distributed development and related terms

AUTHOR (S)	DEFINITION
Grover, Cheon & Teng (1996)	**IS Outsourcing** is the practice of turning over part or all of an organization's IS functions to external service provider(s).
Carmel (1999)	**Globally distributed IS development** projects are projects that consist of two or more teams working together to accomplish project goals from different geographical locations.
Heeks et al (2001)	**Global software outsourcing** is the outsourcing of software development to subcontractors outside the client organization’s home country.
Zheng Yan (2004)	**Offshore software development** generally means that software is developed through collaboration of a team in an emerging country.
Agerfalk et al (2005)	Intuitively, classifying a project or development team as distributed means the team members are not co-located, but geographically spread out; we may thus say that there is a geographical distance between actors in a **distributed development** setting.
Phalnikar et al (2009)	**Distributed development:** This involves co-operation between several teams located at different sites. It will comprise efforts where the bulk of the work is outsourced to developing countries with a small team of consultants working on site with the business stakeholders.

In distributed software development, teams are not physically co-located therefore face-to-face communication and meetings in same room are rare. Global software development is special case of distributed software development and the difference is that in global development, dispersion of the teams is across national borders, and in distributed development the dispersion can be anything from opposite buildings to different continents (Layman et al, 2006). In outsourcing, external company is responsible for providing software development for the client company. When both companies are located in same country it is called onshore outsourcing and when companies are located in different countries it is called offshore outsourcing. Offshoring means that a company creates own software development centers which are located in different countries. Other differences are that in offshoring other parts of the project is mostly done from emerging countries and in distributed development other area can be also near or development can be done decentralized inside same country (Jalali & Wohlin, 2012). Term nearshoring is used when development is transferred to geographically closer countries, decreasing cultural and time differences (Jiménez et al, 2009). No matter which definition is used, general principle is that project development is divided into two or more locations.

In addition to geographical distance there are also possibility to socio-cultural and temporal distances and cultural and language barriers can be present in distributed development. Even though different distances challenge distributed development, between years 2000 and 2010 many organizations and industries have shaped their business towards globalization. Because of this change, globally distributed development and virtual teams have become increasingly popular in many areas as product development and information systems development (Sarker and Sahay, 2004).

Companies are willing to face these difficulties for different reasons. Carmel and Agarwal (2001) give two obvious reasons for moving offshore development: a larger labor pool and cost advantages. Larger labor pool and broader skillset, cost advantages and possibility to around the clock working are mentioned too (Agerfalk et al, 2005). So, technology makes remote working easier and more efficient and same time enables outsourcing to low-cost countries. Distributed development can be relevant also if local expertise is needed to satisfy local demands. Per Herbsleb and Moitra (2001), factors which have accelerated the change towards global software development are; the need to access global and cost-competitively resource pool wherever located, nearness to the market, the fast formation of virtual teams and pressure to using time zone differences in round-the-clock software development.

2.1.2 Challenges and success factors in distributed development

In this section aim is to cover challenges which come with dividing software development. After figuring out challenges I present possible solutions to these challenges and in the end, summarize success factors in distributed development.

Following listings are challenges that distributed development has faced. According to Herbsleb and Moitra (2001) all levels from engineers to executives face challenges on many levels from technical and social to cultural level. Agerfalk and others (2005) list geographical dispersion, time zone differences, cultural differences, language barriers, national traditions and norms of behavior. Delone and others (2005) have identified time separation, cultural differences and geographic distance as barriers to project success. Per Kiel and Eng (2003) main themes which cause challenges in distributed development are time, language, culture, power and trust. Following section gives broader understanding about different challenges. After challenges are covered, solutions for those challenges are managed.

Many different factors, such as cultural and organizational differences, geographic distance and time zones affect to the overall project atmosphere and play important role in successful communication when developing software in a geographically distributed environment (Šmite, 2006). Carmel (1999), takes challenges one level deeper defining what consequences these differences and dispersions cause. In his study, he defines factors that challenges global development teams and affect performance of the teams. One of the factors is geographic dispersion, what is the main reason for three other challenging factors: loss of communication richness, loss of teamness and coordination breakdown. The fifth factor is cultural differences as can be investigated from figure below.

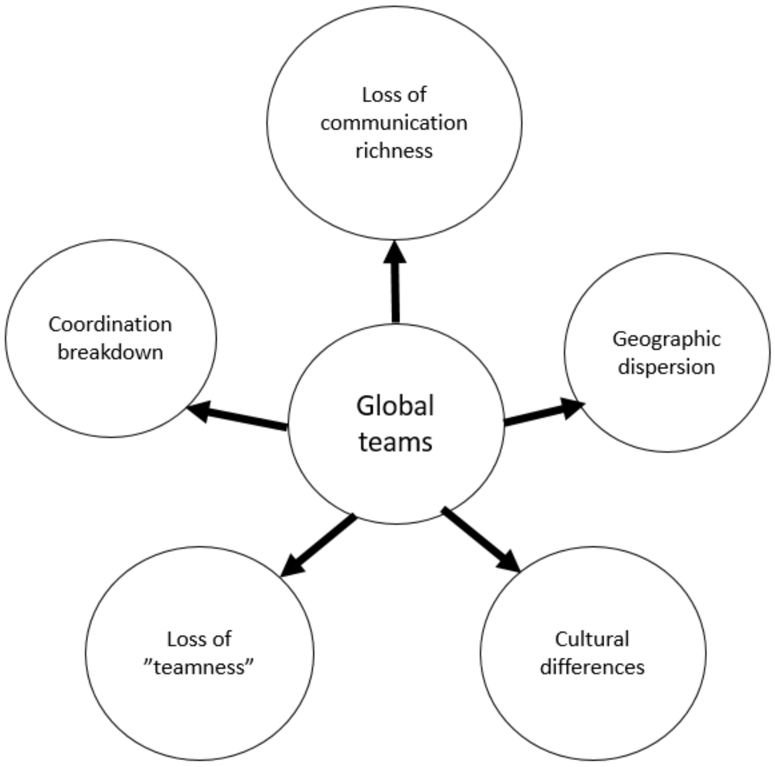

Figure 1 - Challenges in global software development (Carmel, 1999)

Geographic distance means that teams are not co-located. Team separation causes increased coordination challenges, more communication problems and delays. Differences in feedback cycle, misunderstandings and communicating contextual information are practical examples of challenges due to geographical distance (Delone et al, 2005).

Differences in time zones are most of the time present as geographical distance is too. So, temporal distance means that teams are separated by time because of different working hours and time zones. This reduces overlapping working time which makes communication more challenging. Project management and coordination is also challenged by temporal distance because of different schedules and pressure coming from different business sides. These issues with coordination combined with communication problems make prioritization and timing of activities challenging (Delone et al., 2005).

Cultural differences or socio-cultural distance can cause difficulties in collaboration and communication. Deeper reason for these cultural challenges can be divergent values which challenge team members trust to one another (Delone et al, 2005). According to Agerfalk et al (2005), socio-cultural distance means that different team members give different meanings to different situations based on their cultural background, so culture can affect how team members react to certain situation. Herbsleb and Moitra (2001) go even further dividing cultural dimensions to the need for structure, attitude toward hierarchy, sense of time and communication styles.

Carmel (1999) presents six solution factors which unite global teams and make teams more effective. First of the solutions is telecommunication infrastructure which enables software development in global teams. Second solution is collaborative technologies and other solutions are development methodology, product architecture, team building and managerial techniques. These possible solutions can be examined from following figure.

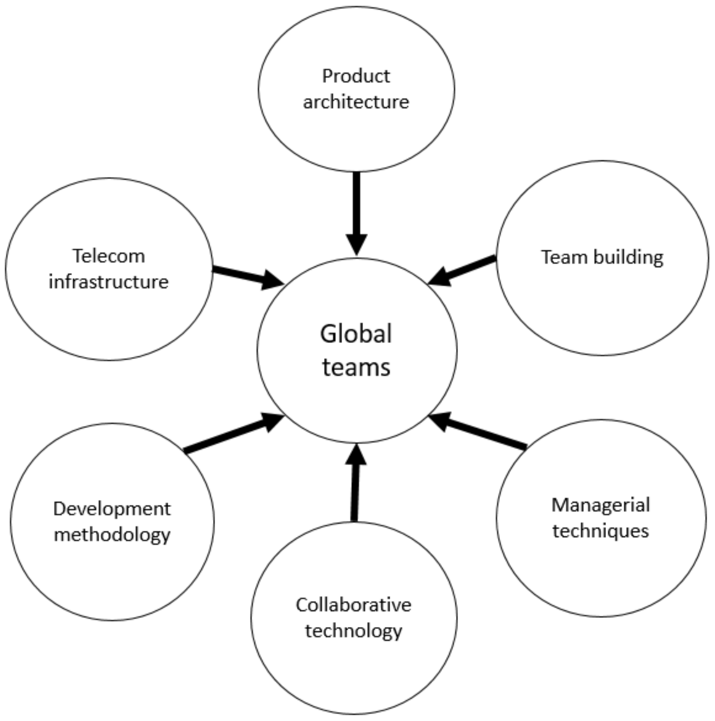

Figure 2 - Solutions for challenges in global software development (Carmel, 1999)

Agerfalk et al. (2005) present that travelling and creating trust and team spirit are the best solutions to overcome geographical distances. Trust and team spirit create "teamness" among distributed team members which reduces geographical distance. Another solution mentioned for geographical distance is nearshoring in which nearshore team is reducing coordination and communication challenges decreasing geographical and temporal distances.

Temporal challenges can be faced different ways. In one case study case company decided to work divided between only two sites to make time zone differences manageable and communication easier. In same study, another case company decided not to make temporal differences smaller, but to focus on cooperation between locations and distributed virtual teams. Whether teams are truly distributed or not, communication technologies enable distributed work (Agerfalk et al., 2005).

Solving culture issues are challenging because those differences are all about individuals and their political or religious values. Best way to face these differences is to create informal and formal communication and knowledge sharing between team members, so team members get know other members and their cultures (Agerfalk et al, 2005).

Prikladnicki, Audy and Evaristo (2003) present three different dimensions of critical success factors in global software development: technical, nontechnical and combination of these called hybrid. Technical factors are related to technical expertise and nontechnical factors constructs of dimensions such as social, cultural, languages and behavior. Based on authors case study technical success factors are software development process and infrastructure. Nontechnical success factors are team integration, communication and feedback. Success factors which go to hybrid category are training, planning and engagement.

2.1.3 Conclusion

When considering global software development and success, most of the times focus is on solving challenges that distributed way of working creates. Addition to earlier presented solutions there are also articles and reviews stating success factors which are not connected to challenges. For example, Evaristo and others (2004) summarize that critical factors to project success are software development process, training of the team, team integration, communication, engagement and IT-infrastructure. Ebert and De Neve (2001) highlight significance of effective tools and work environment for successful global software development. In the following table is summarized success factors in global software development based on reviewed literature.

Table 2 - Success factors in global software development

AUTHOR(S)	SUCCESS FACTORS
Carmel (1999) Ebert & De Neve (2001) Prikladnicki, Audy & Evaristo (2003) Evaristo et al. (2004) Ågerfalk et al. (2005)	Collaborative technology Effective & tools Technical infrastructure IT-Infrastructure Communication technologies
Prikladnicki, Audy & Evaristo (2003) Evaristo et al. (2004)	Trainings Training of the team
Carmel (1999) Prikladnicki, Audy & Evaristo (2003) Evaristo et al. (2004) Ågerfalk et al. (2005)	Team building Team integration Team integration Teamness & team spirit
Carmel (1999) Evaristo et al. (2004)	Development methodology Software development process

2.2 Agile software development

This chapter gives background knowledge about agile software development and the reasons why usage of agile development has increased. In the beginning of the chapter is discussed about different definitions for agile, agility and agile software development. After definitions are managed, I cover challenges and success factors in agile development and conclude with critical success factors for agile development.

Whole software development has faced change where development is done more customer-centric than earlier. The reason for this change is that also business and services have changed and software development must answer faster for changes in business side. Agile software development tries to manage this shift and that is the reason why agile software development has continued increasing its share from year 2001 till nowadays (Dingsøyr et al., 2012).

In year 2001 software practioners introduced "Agile Manifesto" which constructs of four key instructions and twelve principles. Based on these four instructions individuals and interactions should be valued more than processes and tools, working software more than comprehensive documentation, customer collaboration more than contract negotiations and responding to change more than following a plan. So, manifesto directs developers to trust their technical professionality and simplified plans to create value for users deploying efficient software for customer iteratively. These values have been ground for many software development methodologies and methods which can be called agile. Key features in agile development are continuous requirements gathering which needs frequent face-to-face communication, pair-programming, refactoring, continuous integration, early expert customer's feedback and minimal documentation. The most widely used and known agile methodologies are Scrum, Extreme Programming, Feature-Driven Development, Dynamic System Development Method, Adaptive Software Development and Lean Development (Chow & Cao, 2008; Jalali & Wohlin, 2012).

Sheffield and Lemétayer compare agile methods to traditional development and point out the difference between detailed level planning to iterative and adaptive planning where tasks are scheduled and executed only at time when needed. This shift to iterative development creates flexibility and ability to adapt changes to customer requirements. Jalali and Wohlin (2012) agrees that when comparing traditional software development to agile methods, agile development is more flexible in requirements changes. Also, extensive collaboration between customers and developers and encouraging towards self-organized co-located teams are mentioned as characteristics of agile development.

2.2.1 Definitions for agility and agile software development

First thoughts of agile development are tracked to article by Nonaka and Takeuchi from year 1986, but not until conference held back in year 1995 agile methods was mentioned. Still the actual born of agile development took several years and finally year 2001 The Agile Manifesto was published by software development practioners as mentioned. Agile software development is created by practioners and consultants to answer the change in business world and background theory is tested and defined based on practioners experiences in practice. Now as agile development has become more and more used it has come also more and more studied, but most of the studies are still based on industry-driven researchers without conceptual studies of agile software development (Conboy, 2009).

Since the manifesto was articulated, practitioners and researchers have been trying to define agility and its different angles. Agile software development has been in use already almost 20 years now, but still definition for agility and agile way of developing software can be unclear, so next I go through few definitions by different researchers. Fowler and Highsmith (2001) summarize agile manifesto's idea about appreciating more ability to respond to unexpected changes than ability to plan them ahead. Conboy (2009) instead focuses more on the word agile and its meaning and he differentiates agility, flexibility and leanness but still explains that agile development is sum of all these three. According to Conboy, flexibility is the ability of creating change or embracing change and leanness then is about creating needed value for customer measured with economy, quality and simplicity. All in all, most of the definitions promote respond for surrounding changes and needs and speed of the respond. Examples of agile and agility definitions can be observed more in the following Table 3.

Table 3 - Definitions for agile and agility

AUTHOR (S)	DEFINITION FOR AGILE
Fowler & Highsmith (2001)	**Agility** is all about trusting in one's ability to respond to unpredictable events more than trusting in one's ability to plan ahead for them.
Highsmith (2002)	**Agility** is the ability to both create and respond to change in order to profit in a turbulent business environment.
Erickson, Lyytinen & Siau (2005)	**Agility** is often associated with such related concepts as nimbleness, suppleness, quickness, dexterity, liveliness, or alertness. At its core, **agility** means to strip away as much of the heaviness, commonly associated with traditional software-development methodologies, as possible to promote quick response to changing environments, changes in user requirements, accelerated project deadlines, and the like.
Conboy (2009)	Final Definition of **Agility**: The continual readiness of an ISD method to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality, and simplicity), through its collective components and relationships with its environment.
Lee & Xia (2010)	At the heart of **agile** development approaches is the notion of software development **agility**, which is defined in this research as a software team's ability to efficiently and effectively respond to user requirement changes.

Lui and Piccoli (2007) present another way of defining agility and their argument is that information systems agility is composed of technological agility, process agility, people agility and structure agility. Technological ability is about flexibility of information technology and possibility to rapid adjustments if needed. Other features of technological agility are scalability, versatility and adjustability. Process agility is about flexibility of company's business processes and readiness to respond changes in business. People agility is more about individuals and their knowledge and skills. People agility can be measured using training level and job rotation. Structure agility is about organizational level flexibility. Characteristics of that are empowerment, distributed decision-making and lower hierarchies. Researchers also states that agility is not some state that organization is or is not, but rather agility should be measured on a continuum and each of the components mentioned above can be measured separately.

Based on Sarker and Sarker (2009) research the definition for agility is multidimensional and it consists of resource agility, process agility and linkage agility. These categories have also subcategories and for example resource category is divided into people-based agility and technology-based agility whereas process agility consists of methodology-based, environmental-based and temporal bridge-based agility. Parts of linkage agility are cultural-based and communication-based agility.

Highsmith (2002) defines that agile development's typical characteristics are strategic capability, a capability to create and respond to change and a capability to balance flexibility and structure. In addition to earlier ones, also a capability to draw creativity and innovation out of a development team and a capability to lead teams and organizations through unstable times and uncertainty. Dingsøyr and others (2012) summarize key principles of agile development as follows: Empowered and motivated developers, who are relying on technical expertise and simple designs to create business value to users, delivering working software at short, regular intervals. Lee and Xia (2010) defines that agile development is development team's readiness for efficient and relevant response to customer's requirement changes during development project's lifecycle.

2.2.2 Challenges and success factors in agile software development

In this section, aim is to identify what success factors can be found from agile software development literature reviews and case studies. Before covering success factors, I go through challenges in agile development and present solutions to those challenges. In the end of the section I summarize found success factors and solutions in agile software development.

When searching literature about challenges and success in agile development, most of the researches are done considering the change of working method from traditional development to agile development, so it is important to separate these two different angles of research. In this paper focus is on the successes and challenges faced during agile software development project.

Chow and Cao (2008) construct a list of challenges and failure factors in agile development into four categories: organizational, people, process and technical. In organizational dimension, there are for example factors like lack of management commitment and too traditional organizational culture. In people dimension example factors are lack of team work and lack of necessary skill-set. In process dimension, there are factors such as lack of customer presence and ill-defined project scope. In technical dimension, there are factors: lack of complete set of correct agile practices and inappropriateness of technology and tools.

Chow and Cao (2008) describe agile on words flexible and responsive and agile methods as ability to manage atmosphere where changes are constant and emerge with success. According them, agile methods and ability to use those methods efficiently for response needed changes from customers directs to success. One way of identifying success factors is to use Critical Success Factor approach. According to Bullen and Rockart (1981): "Critical Success Factors (CSF) are the limited number of areas in which satisfactory results will ensure successful competitive performance for the individual, department or organization." In other words, critical success factors are key areas where project must success so, that business can be successful and goals can be achieved.

In agile software development, critical success factors can be considered as factors which have to be concentrated on during software development project that it will be successful. After defining critical factors have to point out that there have not been many formal studies on critical success factors in agile development point of view and because of that I concentrate on reviewing case studies and other researches which covers successes or failures in agile software development projects. Reviewing both successes and challenges is beneficial when identifying success factors because from failures and problems is possible to learn how to avoid serious challenges and pitfalls that could be critical to the project success (Chow & Cao, 2008).

Chow and Cao (2008) divide success factors to following five categories: Organizational factors, people factors, process factors, technical factors and project factors. They investigated in their research that which of these categories and sub-categories have biggest impact on project success in terms of quality, scope, time and cost. These attributes of success describe overall success of a project. Quality means that delivered product is working well and scope means that the product meets all customer's requirements. Time means obviously that product is delivered on time and last success attribute is cost which means that product is delivered within the planned budget. Organizational factors category consists of management commitment, organizational environment and team environment. Team capability and customer involvement construct people factors and parts of process factors are project management process and project definition process. Technical factors consist of agile software techniques and delivery strategy. Process factors are formed by project nature, project type and schedule. Categories, sub-categories and success attributes can be found in the following figure.

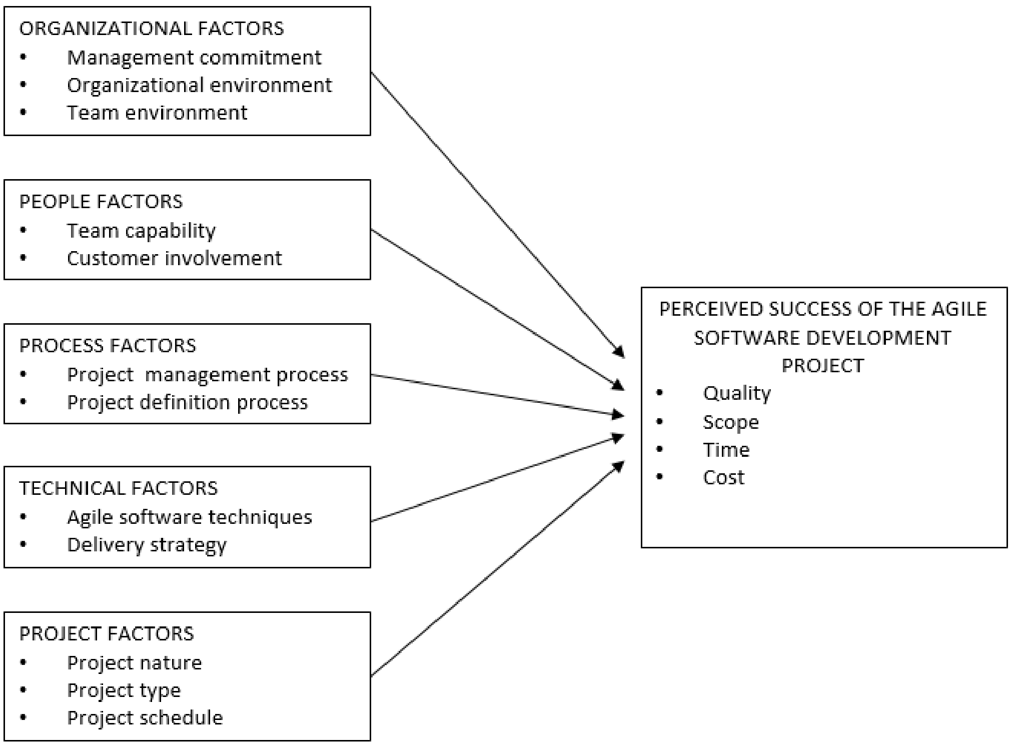

Figure 3 - Research model for success factors in agile development (Chow & Cao, 2008)

Based on research findings of Chow and Cao (2008) critical success factors in agile software development project are, correct delivery strategy, proper practice of agile software engineering techniques and high-caliber team. There are also other success factors which are critical to certain success dimensions but not all. Those factors are good agile project management process, agile-friendly team environment and strong customer involvement. Researchers summarize success factors as follows:" As long as the Agile project picks a high-caliber team, practices rigorous Agile software engineering techniques and executes a correct agile-style delivery strategy, the project could be likely to be successful."

Cockburn (2001) has defined five sweet spots which are ideal surroundings for agile development. First guideline is to have two to eight people in one room working and communicating together. The advantage of this setup is that information moves faster and feedback is easily available. Second perception is on-site usage experts. Because experts are on-site and all times available feedback is short as possible. This rapid feedback supports the development team to get understanding of the customer's needs and habits of the end-users. Third observation is one-month increment. Reason for this is that incremental development provides feedback points and it enable changes and repairs. Next perception is fully automated regression tests. This improves the system quality and easies programmers work. Last sweet spot is experienced developers. Ideal situation would be that team consists only of expert developers.

Cohen, Lindvall and Costa (2004) list three most important success factors identified among early users of agile methods and those are culture, people and communication. All projects have these three, but culture have to be right for agile, people have to be competent enough and communication have to be rapid. In addition, Cohen and other add that close interaction with customer and quickest possible feedback is a critical success factor too.

Misra, Kumar and Kumar (2009) researched which factors have a clear connection with success when starting new agile project and result was list of nine factors: Customer satisfaction, customer collaboration, customer commitment, decision time, corporate culture, personal characteristics, societal culture, and training and learning. So, according to this research all these factors had clear relationship with project success. Factors are divided into different dimensions and for example customer satisfaction, customer collaboration and customer commitment are organizational factors and more specifically customer centric issues. Decision time and corporate culture are also related to organization. Personal characteristics, societal culture, training and learning are all related to people, so those are people related factors.

In the study done by Sheffield and Lemétayer (2013), success factors are more like advices that help to succeed. First advice is to broaden agility level to include factors in both project environment and project. Second principle is not to be blinded by some particular project management methodology from practicing the most appropriate agility level on software development. Third one is about common understanding of software development agility. When project team, project management, top management and customer disagree on agility, project success is not likely. Fourth continues where the third ends, so team, management and customer have to be more adaptable and flexible to project management and software development. Fifth is all about tailoring the development process to fit project's needs and sixth observation is that some of the important success factors to the project can differ in long term when compared to principles of agile methodologies. Researchers concludes success as follows: "In summary, the current study demonstrates that in successful projects software development agility is aligned with factors in the project and project environment. In other words, a one size- fits-all approach to software development agility is most inappropriate."

2.2.3 Conclusion

Four success factors are mentioned several times by different authors, so based on literature these four factors are success factors for agile development: Experienced team, customer involvement and commitment, possibility to get feedback and culture that supports agile development. In the following table, there is summary based on reviewed literature. These four factors can be categorized to people factors and organizational factors. Team expertise and customer involvement are clearly people factors and agile-friendly culture is organizational factor which supports communication and rapid feedback-cycle. Addition to these success factors also communication in general and correct use of agile methods and techniques are mentioned as factors that might affect positively to project success.

Table 4 - Success factors in agile development

AUTHOR(S)	**SUCCESS FACTORS**
Cockburn (2001) Cohen, Lindvall and Costa (2004) Chow & Cao (2008) Misra, Kumar and Kumar (2009)	Experienced developers Personnel characteristics High-caliber team Competent people
Cockburn (2001) Cohen, Lindvall and Costa (2004) Chow & Cao (2008) Misra, Kumar and Kumar (2009)	On-site working experts Close interaction with customer Strong customer involvement Customer commitment
Cockburn (2001) Cohen, Lindvall and Costa (2004) Chow & Cao (2008) Misra, Kumar and Kumar (2009)	Rapid feedback Quickest possible feedback All times available feedback Customer collaboration
Cohen, Lindvall and Costa (2004) Chow & Cao (2008) Misra, Kumar and Kumar (2009) Sheffield and Lemétayer (2013)	Culture have to be right for agile Agile-friendly team environment Corporate culture Project environment

2.3 Distributed agile development

This chapter aims to answer questions such as why companies combine distributed software development and agile development, what are possible challenges in this combination and based on earlier experiences which are success factors to focus on. Benefits and challenges are collected mostly from empirical reviews and earlier experiences because topic is still quite recent and there is not so many existing theories or textbooks. Therefore, solutions to challenges are managed also as success factors.

According to year 2008 State of agile development survey revealed that already 57% of participants were working in distributed teams. Also 47% of survey participants stated that they are combining agile with distributed development (Shrivastava & Date, 2010). There are couple reasons why practioners are combining these two. Firstly, business world is more global than ever and distributed software development has become a major trend lately. Secondly, business requires organizations to develop and evolve software faster and that has made agile development another major trend. As distributed software development requires formal processes executed by traditional plan-driven style, quickly changing environments are key factors for the use of agile development. This conflict creates tension in distributed agile development (Ramesh, Mohan, Cao 2012).

2.3.1 Challenges and success factors in distributed agile development

Lee and others (2006) define agility in globally distributed software development as the capability of distributed teams rapidly implement and deploy software by using IT resources and expertise, to make use of emerging business chances at geographically distributed locations. Sarker and Sarker (2009) define that agility in distributed development is capability of distributed team to quickly complete development tasks and adapt to changing conditions rapidly.

Per Phalnikar, Deshpande and Joshi (2009) the most significant challenges in distributed agile are decreased communication bandwidth, decreased visibility into project status, configuration management and disconnection on project estimations. Ramesh and others (2006) process challenges in distributed development and how some characteristics of agile development work as solution for those. Then they summarize what new challenges are formed in a result of this combination of agile and distributed development.

First challenge is conflict between communication need and communication impedance. In distributed development, formal communication mechanisms are trusted and in agile informal interactions are more trusted. Other clear difference is between fixed and changing quality requirements. Distributed development relies on fixed, upfront requirements and on the contrary agile development trusts in evolving and ongoing negotiations. Third challenge is in different valuing between people and process-oriented control. Distributed development relies control achieved by formal processes and in agile development control is more people-oriented and achieved by informal processes. Conflict between formal and informal is present also in agreements. Agile contracts can be informal and loosely defined and on the contrary in distributed development agreements are explicit. Fifth challenge is about team cohesion. In distributed development teams, can be located on the other side of the world, so these different locations do not feel the teamness and cohesion when compared to agile developments co-located team members. The challenges and their relations to practices, to solve challenges, are presented in following figure 4 (Ramesh et al. 2006).

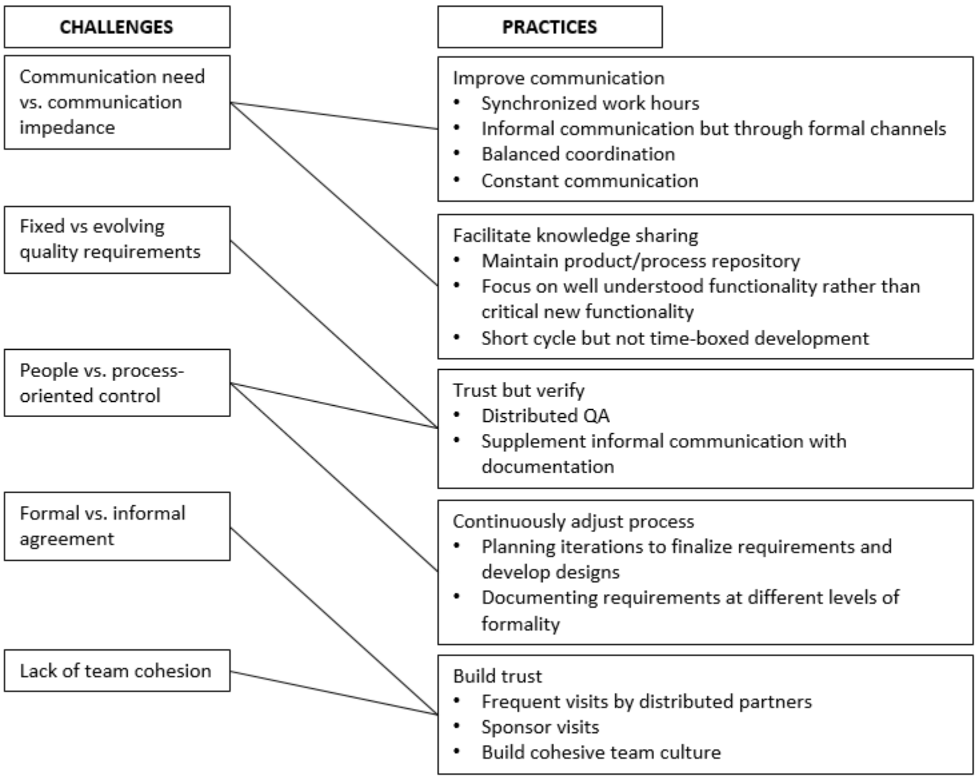
CHALLENGES
PRACTICES
Communication need vs. communication impedance
Fixed vs evolving quality requirements
People vs. process-oriented control
Formal vs. informal agreement
Lack of team cohesion
Improve communication
• Synchronized work hours
• Informal communication but through formal channels
• Balanced coordination
• Constant communication
Facilitate knowledge sharing
• Maintain product/process repository
• Focus on well understood functionality rather than critical new functionality
• Short cycle but not time-boxed development
Trust but verify
• Distributed QA
• Supplement informal communication with documentation
Continuously adjust process
• Planning iterations to finalize requirements and develop designs
• Documenting requirements at different levels of formality
Build trust
• Frequent visits by distributed partners
• Sponsor visits
• Build cohesive team culture

Figure 4 - Mapping between challenges and practices (Ramesh et al. 2006)

Based on the case study conducted by Ramesh and others, there are practices which help to manage challenges in distributed agile development. These practices are categorized to five groups: Improve communication, facilitate knowledge sharing, trust but verify, continuously adjust process and build trust. Practices to improve communication are synchronized work hours, informal communication, balanced coordination and constant communication. Ways to facilitate knowledge sharing are maintaining repository, focus on well understood functionality and short cycled but not time-boxed development. These two practices try to decrease challenges related to communication.

Third practice, trust but verify consists of distributed quality assurance and supplementing informal communication with documentation. It decreases challenges in quality requirements and controlling people and processes. Fourth practice is continuously adjusting process, which includes iteration planning and documenting in different level of formality and it decreases challenges related to controlling people and processes too. Last practice is build trust. It consists of frequent visits by distributed partners, sponsor visits and building cohesive team culture. With these practices, aim is to reduce challenges in agreement formality and lacking team cohesion. Practices to reduce impact of challenges are presented in figure 4 (Ramesh et al. 2006).

2.4 Literature review summary

This literature review examined success factors in distributed agile development. Success factors in distributed and agile development were investigated because both distributed and agile development have been increasing last years as the way of developing software. Also, the combination of distributed agile have grown its attention. Since there were not much practical experience and empirical research related to distributed agile development, in this research agile and distributed development was examined too. So, success factors where gathered using perspectives of challenges and solutions because when challenges were known and solutions were known, possibility to succeed were higher.

Success factors in distributed development were training of the team, team integration and spirit, development process and communication tools and techniques. Addition for that, in agile development, success factors were professional team, customer involvement, collaboration and development culture. So, what are success factors in distributed agile development and what experiences about this combination already exists?

Based on these experiences in distributed and agile development, and literature about distributed agile development success factors constructed of communication, knowledge sharing, trusting but verifying, adjusting continuously process and building trust. First factor was communication which played critical role both in distributed and agile development and also in distributed agile development. Second factor was sum of continuous process, trainings and competent team. Third success factor area was trust. It included building trust between distributed teams, internally inside development teams and with other stakeholders.

Future research areas could take deeper investigation about success factor areas for example informal and formal communication in distributed agile development or some more specified case situation for example time-zone distributed scrum project and success factors in it. Also, challenges and conflicts in combining distributed and agile development should be researched more thoroughly.

3 Empirical research

This chapter is about conducted empirical research. In the beginning of the chapter the aim of the research is presented. After that is presented used research methods and in the end, is implementation of the study including data gathering and data analysis.

3.1 The aim of empirical research

The goal of the research is to find out success factors on distributed and agile software development project. This goal is achieved based comparison of literature review and case study results. The result of this comparison is needed because in the future distributed agile development is increasing and success factors are important to be known. To achieve this aim, following research question is defined:

> What are success factors in distributed agile development and what experiences about this combination already exists?

Following sections presents used research methods and how the empirical research for the case study was implemented.

3.2 Research methods and implementation of the study

Selected research method for this thesis is qualitative research. This selection is made based on comparison of qualitative and quantitative research methods.

Qualitative research is related to specific time and location. The idea of qualitative research is to get realistic, new information about topic which is researched. Many times, qualitative research has inductive analysis, which means that research tries to find out unexpected observations. In comparison quantitative research has usually deductive analysis, which means that research is expected to verify already known truth and use that truth base for further generalizations. Qualitative research method is preferred in situations when the aim of the research is to get information about practical real-world phenomenon which is not researched heavily earlier. (Hirsjärvi ym, 2007).

For data gathering in studies where aim is to get information about practical issues interviews are used. Interviews offers possibility to get more personalized point of views and possibility to ask follow-up questions when needed.

Even in research questions is included part "—experiences about this combination already exists." and that highlights the aim of getting opinion of interviewed person in that specific time and location. Case study is selected because distributed agile development projects are not studied for long, so opportunity to study project which is distributed to different locations and has started implementing agile methodology recently is interesting opportunity. The research is limited to cover one development project because covering agility of the whole development organization would have been too broad subject. The result of this limitation is that interviewees are working in same development project and knows each other.

3.2.1 Background information of interviewees

The research case company is big technology consulting company which is focused on package and custom software development with customers. Case project is part of this kind of customership where consulting company is developing software with customer and end users of developed software are company and household users.

The case study project consists of two scrum teams which are both distributed to two different countries. Managers, product owners, and part of the testers are located in Finland on customer site and Scrum Masters, developers and part of the testers are located offsite in Latvia. Following figure demonstrates organization structure of the case project organization.

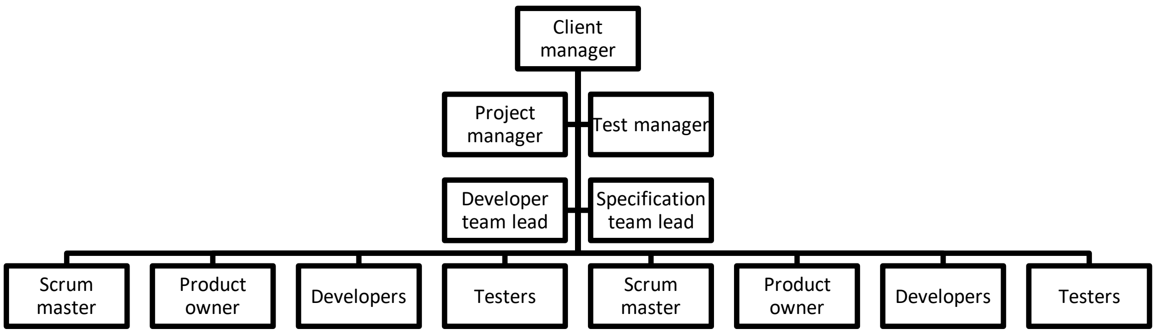

Figure 5 – Case project organization chart

3.2.2 Data Gathering

The research material is gathered by having interviews with selected team member who have proper knowledge and key role as participating distributed agile development project. All the team members have also prior experience working in distributed waterfall setup. The aim of the interviews is to get insights and experiences from distributed agile project and investigate greatest challenges and possible solutions for those and list major success factors for distributed agile project. The questions for the interview where constructed to answer for research question and to cover same areas as conducted literature review. That enables discussion between literature and practical experiences received from the interviews.

The case company was contacted in the beginning of 2017 and interviews were scheduled to be held in February and March and more specific dates and times were agreed through email. As planned interviews were started in February with first test interview, where idea was to verify that interview structure and questions were providing enough and proper data. After analyzing this first interview the rest of interviews were scheduled and held. Two interviews with interviewees who were located in Finland were face-to-face and four other interviews were held using Skype for business -software. All interviews were held in English, recorded and transcribed.

The interview structure was planned to guide in conducting interviews and questions were presented from broader to more detailed. Idea behind this structure was not to guide interviewees when answering and decrease the effect of interviewer for the answers. The interviews followed themes such as agility, distribution, challenges, solutions and success factors. The structure of the interview is presented in APPENDIX 1 and Finnish translations are presented in APPENDIX 2. Both teams have their own scrum masters, developers, testers and product owners. Following table clarifies interviewees roles and locations in this case study.

Table 5 - Roles of Interviewees

Interviewee	Team	Role	Location	Expertise
Team member 1	Team 1	Product Owner	Helsinki	Business
Team member 2	Team 1	Scrum Master	Riga	Technical
Team member 3	Team 1	Developer	Riga	Technical
Team member 4	Team 2	Product Owner	Helsinki	Business
Team member 5	Team 2	Scrum Master	Riga	Technical
Team member 6	Team 2	Developer	Riga	Technical

3.2.3 Data analysis

As described earlier, selected research method is qualitative methods because that is more suitable for investigating topic where is not much earlier researches. Data were gathered using recordings and then transcribed. After transcription data was divided based on themes and team members. Used tool for this categorization was Microsoft Excel. After this categorization interviews were read through and themed for own groups of themes. Background for these themes were mostly same that were in interview questions, so for example if searched "agility" related answers first I went through answers for the questions related to agility and detailed questions related to people, process and structure agility. One example of sentence related to status of agility in case project is following:

> Well, I think that it's not very agile because my understanding is that what we have kinda waterfall-agile style.

After transcribed data was categorized using themes and citations were categorized using identifying data of interviewees. For example, previous citation about agility were from team member 3, whose locations is Latvia and role is technical. Other possible combinations were for example team member 1, Finland, business. Data was analyzed and categorized and following section of this thesis presents results based on categorized themes.

4 Results

In this chapter I am going to answer for the research question: "What are success factors in distributed agile development and what experiences about this combination already exists?" based on results of empirical research and conducted literature review. Discussion about found observations is followed in chapter 5.

4.1 Experiences in agile development

Practitioners and researchers have been trying to define agility and its different angles. Agile software development has been in use already almost 20 years now, but still definition for agility and agile way of developing software can be unclear. Whether project is agile or not, is hard to specify explicitly and in this case project, this same unconsciousness was also present. All team members stated that project is agile at least in some degree, but how agile, differed between individuals. Team member 1 described case project's agility as follows:

> My opinion is maybe that our framework how we operate that's towards agile, we have the scrum teams working based on agile principles but overall, I would say the project governance or the management of the project, that's more waterfall.

Team member 3 agreed that agile and waterfall both are still present in development, and it's not very agile cause waterfall is still present as series of handovers. Team member 2 highlighted that team and whole project has become more and more agile, but because the lack of previous experience of agile projects, current agility is only medium agility. Team member 5 had noticed also that first steps towards agility is taken, but without previous experience there is still long way to go. Team member 6 summarizes the current state of agility:

> The whole way of working is not implementing full agile, as we do not have ready release at the end of the sprint, but in comparison to waterfall this is more transparent for the developers, meaning there is less management and there is more planning it yourself.

Some parts and phases of the project is still from waterfall, but more and more the way of working is moving towards agile. Team member 4 brings a little bit different angle to the agility, saying that agility varies so much on the context and it depends also level to which try to reach. Based on earlier experiences, team member 4, had never seen full agility and interviewee states also that there is always some component of something else involved in development as well.

In literature, many researchers have suggested that agility is a binary condition that company or team either has achieved or has not. Lui and Piccoli (2007) questioned this notion and suggested that company can be characterized on an agility continuum and contend that organizations normally achieve different degrees of agility. They introduced degrees of process agility, people agility and structure agility. Because all interviewees agreed that there is some level of agility present, I wanted to clarify which part of case project is agile and which degree not. In interview, after general question about agility, I specified these degrees and asked one by one, how specific agility is present in case project. Based on the answers I summarized following table 6 of team members' answers. Process agility is seen the most agile degree of project, people agility also for some extent but structure agility not. Differences in people agility can be related to team member's roles because developers felt way of working at least quite agile (team members 2,3,5,6). Instead product owners' opinion was that people are not very agile, but improving from previous.

Table 6 - Team members' degrees of agility

Team member	**Process agility**	**People agility**	**Structure agility**
Team member 1	Quite agile	Towards agile	Not very agile
Team member 2	Partly agile	Limited agile	Not very agile
Team member 3	Agile	Agile	Not very agile
Team member 4	Agile	Not very agile	Not very agile
Team member 5	Towards agile	Limited agile	Partially agile
Team member 6	Agile	Quite agile	Not very agile

Case project is agile in process degree which means that project has flexibility of company's business processes and readiness to respond changes in business. People agility has more separation inside teams and result is that project is not agile as it could be but still towards agile. People agility is about individuals and their knowledge and skills. People agility can be measured using training level and job rotation. So, probably some of team members have had more trainings and possibility for example job rotation and working in different roles and some others not. About third degree of agility all interviewees are agreeing, in degree of structure case project is not agile. Structure agility is about organizational level flexibility. Characteristics of that are empowerment, distributed decision-making and lower hierarchies.

One factor which might explain lack of empowerment and power for decisions is centered decision-making. Team members 1 and 4 in product owner roles open the decision-making process:

> We do not have much power in the teams yet I think. Because for instance the prioritization and such is very strict and deadlines from this more like "waterfallish" approach to release schedules and UAT schedules and so on. Those are very constricting, so I think the structural kind of like the decision making has not changed from the previous yet.

> I would say that power for decision making that's more or less given, so we don't have so free hands to do the decisions, so for example on my perspective as a product owner I see that I'm more kind of proxy for the team and I represent the client but basically the client is the one who makes the decision and I should work alongside what they are willing to have and how they are prioritizing issues.

To sum up in this case project, project setup works so that client orders and plans what they want and then in the planning and design phase our product owners are involved. This causes the situation that decision about ordered project is already done and decision related to every project, needs to go through client's representative person. To make this degree more agile, success factors areas to focus on are people factors and organizational factors. Especially customer involvement and agile-friendly culture are factors which support communication and rapid feedback cycle, which makes empowerment and divided decision-making possible.

4.2 Experiences in distributed development

The case project is at least partially agile but it is also distributed. Most of the times distribution is considered as two or more different locations where team members work, but because these different locations can be at different city, country or even continent, there are possibility for temporal and cultural differences present. As structured in interview structure, first I asked general question of how distributed case project is and then specified geographical, temporal and cultural distances of distribution. Team member 1 sums up case project's distribution:

> We have two locations, so basically this on-site where we are facing the client and working with the client that's of course the one part and then it's the distributed part, so part of the development and testing is located in Riga, so we have basically two locations where we operate and that's like the level of distribution, so two venues where the team is located.

All team members agreed that there is geographical distribution and there is not temporal distribution. That is quite expected because development center is in Riga and on-site team is located in Helsinki, so both locations have same time zone. Term nearshoring is used when development is transferred to geographically closer countries, decreasing cultural and time differences (Jiménez et al, 2009). Team member 5 summed level of distribution in case project:

> We have same time-zone and of course culture wise I am pretty sure that there are some differences between Finnish people and Latvian but I do not see that there are major differences, so I think we are quite close culturally also.

Team member 3 pointed out interesting view about development team not being distributed because all the developers are in Riga and that the team member did not felt much distribution due to the co-location of developers. Following table 6 presents team members' thoughts of distribution.

Table 7 - Team members' distances of distribution

Team member	Geographical	Temporal	Cultural
Team member 1	Distributed	Not distributed	No big differences
Team member 2	Distributed	Not distributed	No significant differences
Team member 3	Distributed, developers not distributed	Not distributed	No differences
Team member 4	Distributed	Not distributed	Silos of individual people
Team member 5	Distributed	Not distributed	No major differences
Team member 6	Distributed	Not distributed	No differences

All of the team members agreed that there were not major differences in the cultures of two locations, but few team members told that working language inside teams is English and working language of customer is Finnish, so there is slight possibility for language barriers. Team member 1 explained the language difference:

> Client's official language is Finnish, so that's also one aspect, so there is also the language difference between the team members, but I haven't seen that the distribution hasn't caused any challenges.

Team member 4 touched the same issue as team member 3 earlier about teams not so distributed as such, but the point of view was different. Team member 4 stated that the challenge is not distribution of teams but more distribution of individuals.

> I think it's the fact that even in Riga delivery center and here people are working from home a lot, so I mean it does not matter if we are distributed in a sense that we are in different countries, I think that the thing that affects most is that we are missing some of the face-to-face communication that there should be and I think that sort of like sometimes makes this too serious.

As said geographical distance means that teams are not co-located. Team separation causes increased coordination challenges, more communication problems and delays. Differences in feedback cycle, misunderstandings and communicating contextual information are practical examples of challenges due to geographical distance (Delone et al, 2005). Next chapter contains challenges related to distributed and agile experiences that members of the case project have faced.

4.3 Challenges in distributed agile development

Based on Ramesh and others (2006) main challenges in distributed agile development are related to communication, requirement changes, people and process, formality and informality and also team cohesion. Phalnikar and others (2009) highlights decreased communication bandwidth, decreased visibility, configuration management and disconnection on project estimations as biggest challenges in distributed agile development.

Case project interviewees raise five different challenges faced during distributed and agile way of working. First and most common challenge raised is communication. Second challenge area is lack of agile mindset. Third challenge is about estimations in agile and fourth one is lack of earlier experience in agile projects. Fifth challenge is about roles and team structure.

Researchers construct a list of challenges and failure factors in agile development into four categories: organizational, people, process and technical (Chow & Cao ,2008). On the other hand, Carmel (1999), defines factors that challenges global development teams and affect performance of the teams. One of the factors is geographic dispersion, what is the root for three other challenging factors: loss of communication richness, loss of teamness and coordination breakdown. The fifth factor is cultural differences. Phalnikar and others (2009) point that the most significant challenges in distributed agile are decreased communication bandwidth, decreased visibility into project status, configuration management and disconnection on project estimations. First team members 1,2, 3 and 4 summarize challenges concerning communication:

> Maybe one thing is the communication and the transparency in general, I would say it would be beneficial if we would have this kind of bigger venues where we go through the new items what are planned for the sprints and that everybody would have clear understanding of what's happening.

Team member 1 pointed out that one part where communication is lacking is big picture. Reason and solution could be that there would be arranged bigger venues and meetings where all stakeholders are present. In such meeting, also larger vision and goals could be communicated.

> From agile trainings, there suppose to be discussions together with client, so in theory developers should also see clients and talk together but currently communication with client is handled by other team members, I do not actually know who is talking with client.

Team member 2 highlighted communication between developer and business person. In case project that happens seldom because product owner is the messenger in between of client's business persons and developers. Many times, also language changes in a way that when client representatives discuss with product owners, language is Finnish and then when same issue is managed together with developer, language is English.

> I think that some information is not shared and people are not very good, I'm kind of like team lead there so I have much more knowledge about what is happening in the project but they know that most of the emails and other communication channels that I'm having it's not distributed to other team members and even to the one that want to know what is happening.

Team member 3 is concerned about overall information sharing throughout the team and especially that sometimes relevant person does not have access to relevant information.

> I think that the thing that affects most is that we are missing some of the face-to-face communication that there should be and I think that sort of like sometimes makes this too serious.

Team member 4 raises up face-to-face communication and formality of conversations. These both have critical role in agile development and same time are quite hard to maintain in distributed development. As presented previously, communication is clearly one area where challenges are met both in literature and in case project. In case project communication challenges relates to knowledge sharing, informal face-to-face communication and misunderstandings due to distribution. The second challenge, which touches communication closely and is present in case project is lack of mindset for agile development. Team member 4 opens what about agile mindset is missing and what is already there.

> We have the tools, we just need the attitude.

Case project has the tools for working agile but the challenge is in attitude. Team members 3 and 5 presents the challenging attitude as follows.

> I think that also the challenge is that like not every team member is motivated to being agile and some people.. in agile environment and in agile project every team member should be aware of what is happening in the project and we don't have this and even worse that some people just don't care about it.

> I do not know. I think actually basic question for me is do we really want to become really agile, I think that is the main question.

Based on previous citations mind shift and attitude is missing in case project teams at least partially and team member 4 has already noticed that there is still change that is needed.

> I think that for many of us we need a mind shift. I mean we kind of for instance I would really want our scrum master to be like, well I cannot change people's personalities but I would need them to be more enthusiastic and more kind of leading and kind of

> like, how should I say, they should be this "esikuva" whatever that is for other people, they should show the way, they should be more pro-agile and that is really big mind shift for other people as well. We should be more willing to experiment and like I said not so serious.

Third challenge is planning and estimations. Chow and Cao (2008) present that in process dimension, there are factors such as lack of customer presence and ill-defined project scope. In technical dimension, there are factors: lack of complete set of correct agile practices and inappropriateness of technology and tools. As presented previously tools as such are appropriate and basics of agile practices are present, but because of missing mindset and attitude, also some of the practices are not fully working, for example planning and estimations. Following citations describes status of planning and estimation in case project.

> One challenge is that, our burndown chart doesn't look quite good or even worse there are deviations so there is something wrong with estimates or scope of sprints. Either we are unable to estimate tasks good enough or sometimes something unplanned becomes more important, so we prioritize for example bug fixing.

> One thing is from the planning perspective, so there is some rough-level estimation of the items before we take those in the development and testing and also before the specification, so we do the rough-level estimation in a very beginning but since the specification-process is kind of agile, so the rough-level estimates and the first idea what could be the feature and what it contains that might differ quite a lot what it comes then we really start to development and testing-processes, so also the estimation is not so accurate since we are doing the estimation so early phase, so I think that's also caused some issues in our project.

Roles are also mentioned as challenging when working in distributed agile development. This is legacy of waterfall development, when team members had tight roles and areas of expertise and they focused only on that. Then again in agile development ideal situation is that every team member is able to work in various roles. Researchers Chow and Cao (2008) described challenges in people dimension with example factors like lack of team work and lack of necessary skill-set and this lack of skills is linked straight to capability to change roles inside teams. Team member 4 opens the challenge with members being experts and still being able to change roles.

> I think one of the things is like what we spoke about earlier already is that people are so experts, some much experts in just particular areas, that's a challenge and I have not much ideas on how we can really change that. That's like very tough, I mean we can always teach people. They can go to trainings or they can have colleague mentoring them on how to learn to do other stuff as well but then on the other hand I think there is also kind of certain level that which we need these people to be experts, so that is a balancing.

Team member 2 brings up the challenge about teams divided based on the area of expertise and that resulting as silos.

> I think also this agility is limited because people are working on their areas and we cannot put, or get one guy working in some different area because mostly one, well not one but several areas which are areas of competence for that particular team member. So, I think this makes some kind of silo mentality or whatever it is called properly.

Team member 6 agrees about specialization affecting agility.

> It's quite hard like use the full agile where everyone would be all-around specialists and would be able to take every task. Like, just to take any task of the backlog and do it from beginning to the end. It's like, It's less possible here. Yes, and in this way we are still not actually agile.

Fifth challenge in case project is lack of experience in agile development. This is probably one of the key challenge because it affects for the attitude, communication, roles and practices related to agile method. Team members 3 and 5 describe current state of experience in agile.

> First challenge is that we are all or almost all new to agile projects, so we all are kind of learning.

> I think especially like our Riga team we have, we don't have people that actually have worked in agile projects and nobody knows what is this and how it should be done and also our team is very young.

> There are things that we are kind of learning and for most of team members this is first agile project and probably we do not know exactly how to work probably, do not spend enough time to reflect this.

To sum up challenges communication cannot be highlighted enough but also lack of experience can be reason for most of the challenges in case project. Distribution has major effect for communication and to increase communication tools, attitude and mindset must be present. To get those things working properly experience in agile development is needed, and also trainings for team members to understand in practice what for example agile mindset means in context of this case project.

4.4 Success factors in distributed agile development

In previous chapter there is covered what distribution, agility and challenges with distributed agile development are in this case project. Next is covered success factors for the case project. When discussing about success factor it is important to understand that success itself is hard to measure and different stakeholders can define success differently. Based on literature, critical success factors where defined as follows: "Critical Success Factors (CSF) are the limited number of areas in which satisfactory results will ensure successful competitive performance for the individual, department or organization (Bullen & Rockart, 1981)."

Based on literature review three critical success factors where identified: communication, continuous learning and trust between all stakeholders. In case study, interviewees answered question: "What are success factors in distributed agile development?" explanation for success factors were based on Bullen and Rockart (1981) definition that critical success factors are key areas that project members need to focus on to get successful results. Following section presents team members' observations of success factors.

> I would say the most important one that we have good co-operation between both locations and co-operation and communication maybe those are hand-to-hand so both of those, and I would say also the transparency is one that makes it successful that everybody knows the big picture and knows their role and how they are contributing to the overall project, so that's also one.

Team member 1 highlights communication and information sharing, so that all team members despite location or role could know bigger roadmap and direction. Based on that information sharing transparency and feeling of belonging and teamness would also improve. He points out the fact that when bigger picture is understood and team members knows their roles then contribution for overall project is better too.

Team member 2 agrees importance of communication but he points out especially in person meetings and getting together as a whole team. So, in the other words it is communication in face-to-face manner. This kind of communication is limited because of different locations, but arrangeable if needed.

> I think it would be good to have in person meetings and getting together as whole team.

Team member 3 raised communication also as success factor but her emphasis is more about getting feedback about done work. This kind of communication is about project's information flow, so how information flows from customer to product owner and from product owner to the development team. One part of feedback loop is also between developers and testers. Another view for communication as success factor is escalating issues and getting solutions for identified challenges. This is known responsibility area of scrum master in agile development.

> Since this is distributed project the most important thing for us is communication. Faster the better, because if has someone somewhere some issue it should be escalated immediately, so it will be solved earlier and there won't be kind of too many issues collected and project will be successful.

Team member 4 emphasizes communication and correct roles inside team. Addition to those she also lists right technical tools and platforms for one possible success factor. Correct roles inside team means that right people are doing right work, so expertise and work tasks are matching and all team members creates value in their own role.

> Probably I cannot emphasize communication enough, and I think also the sort of having the right people doing the right work.
>
> Communication and the right people doing right things, I mean doing right things like having the right priority for the things we are doing and also doing this stuff in the right ways, because I mean let's say for instance currently we are in situation sometimes where if we made some technological advances, let's say we had a couple of enablers in bottom we would be able to do the work that we are doing are producing whatever feature we are working on, we could actually do that in a better way, so I think that could be also critical success factor that we would have the right technical platform and tools.

Team member 5 lists communication as biggest challenge for distributed teams and same time overcoming this challenge is also critical success factor. Another mentioned success factor is agile trainings to get whole team more experienced and more unified. Now when only part of team has been trained some ways of working feels given, but after trainings common understanding would be better.

> I think in distributed teams the biggest challenge is communication. If teams are co-located, then communication much easier.
>
> We have agile training for scrum masters and scrum master substitutes but of course it could be good that whole team could have some training.

Team member 6 is having same thoughts as team member 3 about feedback loop and also similar approach for face-to-face communication's importance as team member 2. His argument on behalf of eye-to-eye communication is that in such communication team members are able to feel whether other member understands meaning of discussion or not. It's easier to be sure that there is common understanding present.

> Communication. If you have questions, call. If you are not sure about something, call. Just talking eye-to-eye also is very important, so calling is like, it's solve most of communication issues. Maybe because when you talk to each other eye-to-eye you can, you can feel that other person doesn't understand something or you don't have, you are not on the same page with the other person.

Results based on interviews are presented in following table 8. After table comes next chapter, where results are discussed and reasons behind differences and similarities in results are considered.

Table 8 - Success factors in distributed agile development

Team member	**Success factors**	**Action to success**	**Results**
Team member 1	Communication	Close co-opera-tion	Transparency
Team member 1	Roles	Team members are in correct roles	Efficiency
Team member 2	Communication	Face-to-face	Common understanding
Team member 2	Team cohesion	Team meetings	Team spirit
Team member 3	Communication	Team work	Efficiency
Team member 4	Communication	Face-to-face	Team spirit, informality
Team member 4	Roles	Team members are in correct roles	Efficiency
Team member 4	Tools	Technical tools and platforms	Efficiency
Team member 5	Communication	Co-location	Ease
Team member 5	Training	Agile trainings	Agility
Team member 6	Communication	Eye-to-eye	Common understanding

5 Discussion

In this chapter aim is to look behind results and discuss what is causing differences and similarities in literature review results and case study results. For example, how team members' roles or locations affects to the results. First section is answering for the research question: "What are success factors in distributed agile development and what experiences about this combination already exists?" Second section compares results and in the end of the chapter also limitations of this research are presented.

5.1 What are success factors in distributed agile development and what experiences about this combination already exists?

Based on empirical research the most critical success factors in distributed agile development are different levels of communication, people working in correct roles, team spirit, technical tools and trainings.

Communication is mentioned by all of team members, so it is clearly seen as critical factor in distributed agile development. Most of the team members highlighted especially face-to-face communication as the most important factor. Other levels of communication mentioned are communication inside development team and also communication in whole flow from business expert to developer. Those different levels have also different results as face-to-face communication increases efficiency, informality and common understanding, communication inside team increases team spirit and informality and communication through whole project gives transparency and broader understanding.

Both product owners located in Finland mention that one success factor is that correct team members are working in correct roles. Result from team members working in correct roles are working efficiency and increased interest and meaning for own working. This factor is linked to the change from component team towards more agile team, where is new team member roles like scrum master.

All of interviewees have participated in agile trainings but one of the team members raises the fact that part of developers has not participated in any agile trainings and case project is their first touchpoint for agile software development. Trainings could improve understanding for roles, ceremonies and culture of agile development and in that way increase efficiency and overall agility.

In software development technical tools are always part of the working but in distributed agile development tools affects to communication because without relevant tools different levels of communication is challenging to secure.

Tools are emphasized thus without efficient and easy ways to communicate between different locations, agility and informality would be challenging to reach.

Team cohesion is pointed as success factor and team meetings are mentioned as action to improve it, so it's one level of communication, especially to decrease feeling of distribution. Team cohesion and team spirit is maybe more of a consequence of communication and culture inside the project than success factor itself.

Based on literature there are three broader success factors in distributed agile software development. First factor is communication and knowledge sharing. Level of communication is team's internal knowledge sharing to reach efficient information flow. In literature especially when discussing about agile development also informal face-to-face is raised as important, but as same time distribution challenges that. One solution for more face-to-face experience is use telecommunication tools with video to get better notion of team member's reactions and feelings.

Second factor is sum of continuously adjusting processes, trainings and competent team. Continuous improvement and process for it has raised interest during last years when terms like "devops" and "agile" has gotten more known. This continuous improvement is about fast cycle from idea and plan to working software, and with fast cycle it is also possible to get rapid feedback and using this continuous cycle improve end-product. Getting this continuous process working, team members need to be in correct roles and that is secured using trainings.

Third success factor area is trust. It includes building trust between distributed teams, internally inside development teams and with other stakeholders. Trust is big word and it is hard to measure, but in project teams it can be observed from informality and transparency. For example, when team members have informal conversations about each other's lives bond and trust develops. Transparency is result of members trusting each other. There is no need to hide challenges, because when all stakeholders are on the same line and going towards same direction, this kind of communication creates efficiency but also trust.

5.2 Comparison of research results

Based on literature review and empirical research part of the success factors are same, but some differences are present too. Communication, trainings and team cohesion are mentioned in both, roles and tools are only mentioned in empirical research. Continuous improvement, competent team and trust are mentioned only in literature review. Following table describes similarities and differences related to success factors in distributed agile development.

Table 9 - Comparison of success factors

Success factors	In literature review	In empirical research
Communication	x	x
Trainings	x	x
Roles		x
Technical tools		x
Team cohesion	x	x
Continuous improvement	x	
Competent team	x	
Trust	x	

5.2.1 Communication

According to previous studies in literature communication is highlighted and areas to concentrate on are synchronized working hours, informal communication through formal channels, balanced coordination and constant communication. In the research synchronized working hours are not mentioned, but that is because in case project there is no temporal distribution. Research supports informal and constant communication highlighting especially face-to-face communication.

Balanced coordination is not mentioned in research results as such but both product owners want more co-operation and transparency, so partly those cover each other. Notable about case project is the fact that part of teams is located with customer and others in different country so equal communication is challenging, but balancing and having development side joining the meetings remotely supports balanced coordination.

Constant communication is the level of communication what occurs inside development teams. In practice constant communication in case team is limited because of distribution, but for example teleconferences can be harnessed to get as close as possible for constant communication. The main idea is to encourage for telecalls, videocalls and chat instead of using email and other slower ways of communicating. Findings from empirical research supports the significance of constant communication as success factor.

5.2.2 Team

According to literature team cohesion is great challenge for distributed agile development and actions to ease the challenge are frequent visits by distributed partners, sponsor visits and building cohesive team culture. According to empirical research team cohesion is factor that affects success and team meetings, meaning visits between both locations, is definitely one possibility to increase teamness and feeling of togetherness. Sponsor visits were not mentioned separately by team members, but overall transparency and broader communication is supporting this factor partially.

In literature, building trust is linked to team cohesion but in research that does not get any mentions. One reason for this might be the fact that building trust is one general cornerstone in team work and most of the interviewed persons have been working together two years minimum, so this kind of fundamental has forgotten.

Building cohesive culture does not raise up in interviews as success factor but as current challenges in agility there are mentions that there in the case project should be more non-formal communication between people, which is something that is needed to create the culture, the agile culture that would need more informal working together. So, building this culture is not mentioned as success factor, but importance of agile culture with informal communication is noticed.

In literature review the importance of correct roles for team members is not noted as success factor, but according to case study correct roles play important role in team's efficiency. Roles are linked to agile quite tightly and when team moves from waterfall development towards agile development, conflicts old ways of working with new ones.

5.2.3 Continuous improvement

According to literature review continuous improvement consists of competent team, trainings and continuously adjusting process. In other words, to succeed, competent team is needed and to get competent team, there needs to be trainings and processes adjusted time to time. Competent team members, high caliber team, experienced developers and personnel characteristics are already mentioned in agile developments success factors but expertise and capability of distributed teams are mentioned in success factors in distributed agile development. This success factor is not supported based on empirical research but it's noteworthy that interviewees are team members so it might be challenging to see this kind of success factor from the executing level.

Based on the literature review trainings are mentioned as success factors in distributed development and that is success factor in distributed agile development too. Empirical research supports this finding and trainings have important role towards competent team. Trainings give new point of views, ways of working and possibility to reflect current practices with someone outside of executing team. This makes continuous improvement possible and ongoing.

5.2.4 Prerequisites

There are couple of important factors which are mentioned only in empirical research or background parts about agile development or distributed development, but are still worth of mentioning because as mentioned earlier team members are working in executing level and some of the success factors are taken as given.

First prerequisite for successful project work in distributed development is communication tools. Without proper communication tools like chat, videocall or teleconference, decreasing the communication bandwidth is challenging. In literature, collaborative technology and telecom infrastructure are mentioned as success factors for distributed development, so earlier studies support the importance of tools for communication. Product owner of other team raises technical platforms and tools as one notable success factor. Having technical platform and tools are fundamental preconditions for any software development project and those emphasizes when development teams are working distributed. One example in literature, is about product repository which supports teams' knowledge sharing.

According to review of agile development success factors many different studies emphasize importance of culture. There are mentions that culture have to be right for agile, there needs to be agile-friendly environment, corporate culture and project environment which supports agility. In distributed agile development culture is challenging, because of two or more different locations are present and two or more working cultures are present too. Empirical research touches on agile culture, but it is mentioned as current challenge in agile development and one key thing to improve. The challenge in case project is that many team members have worked years in same teams and everyone has become expert in some area of development, so culture where team members would move into areas beyond their current roles is challenging to reach.

To sum up comparison of results, there is strong agreement between literature review and empirical research about importance of communication to succeed in distributed agile development. Team cohesion is mentioned as success factor in both studies and trainings are also highlighted. Differences occur when dealing with competent team, roles, technical tools, continuous improvement and trust. To clarify, mentioned differences are partially covered as for example trust is result of team cohesion and informal communication and competent team is result of trainings and correct roles.

5.3 Existing experiences in distributed agile development

The case study group consists of six team members who have worked in distributed development over two years and in agile development at least one year. As stated in results sections for most of team members this is first agile development project and from Latvian part of the team only interviewed persons have had agile training. Said that, there have been over a year distributed agile development and next I am going to go through experiences based on interviews. Based on the literature review there were not much experiences, so discussion is mostly done based on the case study.

Agility and distribution are both challenging terms, cause different persons experience level of agility and level of distribution differently. To ensure that experiences are related to distributed agile development, both terms where defined and separated to three different levels. According to interviews, the case study project is agile in processes and partially in people, but in structure agility is still not achieved. The reason behind state of structure agility is lack of empowerment and power for decision-making. Product owners state that external prioritization and given deadlines are more waterfall development than agile, so inside team agility has increased but from customer side, there occurs still constricting structures.

Distribution was divided to geographical, temporal and cultural levels. According to interviews, team is geographically distributed, but there is no temporal difference and only a little cultural difference. The fact that teams are not distributed temporally, gives possibility for real-time communication and eases distribution, because there is no information handover needed. One developer points out that all developers are in Latvia, so in sense they are not even distributed. Still, customer and customer representatives are in different country so distribution is present. One of team members raises issue about remote working and silos of individual people. Previously, in waterfall development challenge has been component teams, where developers do only one thing they are experts and that resulting as silos of experts. In agile development as said earlier, communication and especially face-to-face communication has significant role and when team members are working mostly remotely they become silos of individuals.

Team members point out few ways to improve current development based on experience gathered in distributed agile development. First area of improvement is structure of development teams. Currently, same developer might work in two to four different components and developer is challenged because different components are made using different technologies and that causes jumping from different kind of technology to other. So, developers could be divided more strictly to component teams. Same time approaching agile development, roles should not be as tight as those have been and all team members should know about others working. Developers should know how testing is done and testers should know how development is done.

Apart from team structure and specialization, development could be improved changing whole project's mindset. Part of team members do not want to share information and other members do not want to receive information. So, knowledge sharing could be improved. Partially related to knowledge sharing, also client organization should be change and the change towards scaled agile framework causes structure which could enable development team to get the support from the management and other vendors and in that way improve information and knowledge flow.

Last thing which is related to mindset of development teams is that do teams and team members really want to become agile. Because team members have used to work in some area of specialization and this change towards agile broadens working also for other areas of software development. The fact is that if team members do not want to become more agile, team does not become more agile.

5.4 Limitations of the research

In this research, there are limitations related to interviews, literature review and researcher, which might affect to the results and generalization of the results. The research group was quite small and all interviewees are working in same development project, so it might decrease reliability of this research. In same time, this research is qualitative interview research and aim is to get new information of topic which is still not researched thoroughly. So, based on that fact the small group of interviewees is not limiting this research.

Limitation related to literature review is low amount of references from previous studies. The fact that distributed and agile development has increased is not yet covered by literature. To get understanding of current findings in literature, literature was reviewed using previous findings from agile development and distributed development separately, and then by combining results from these areas to cover success factors in distributed agile development.

First limitation in empirical research is when interviewees are selected, because when study is about distributed and agile development project, interviewees need to be working in this kind of setup and also have enough knowledge about both issues. The definition for having enough knowledge about distribution and agility is made based on work experience and trainings. Interviewed persons were selected because all of them had participated agile trainings. Other limitation is that team members are divided by two locations and emphasis on technical and business sides. To decrease effect of these differences, interviewed persons were selected so that all project roles and locations are presented.

Second noteworthy limitation for generalization is that all interviewees are working in same project, so there might be client specific, person specific and process specific challenges or success factors which are not relevant for distributed agile development, but are relevant in this case project. Aim is that when all interviewees are from same project, differences related to client, organization or processes are not drawing attention from more general findings.

Third limitation is definition for distributed and agile development. Because of different state of agility and different state of distribution, findings from this case study may not be relevant for example in distributed agile development project where is big temporal distribution. So, it is important to understand current levels of project's agility and distribution. In this empirical research this challenge was managed defining distribution and agility of the current project, using terms people agility, process agility, structure agility, geographical distribution, temporal distribution and cultural distribution.

Fourth limitation is related to conducting interviews. There is always possibility for human error and misunderstandings. This empirical research is first that researcher has done and there is possibility that his competence and experience is affecting for the research. The researcher is part of the case study project and working with interviewees almost daily, so there is possibility that this personal relationship affects for the research especially because interview is conversational situation. On the other hand, the fact that researcher knows backgrounds of the case project, there is possibility that common understanding is easier to get with research group. The fact that researcher knows interviewees might relax the atmosphere in the interview-session and in that way, there is possibility that this atmosphere raises trust during sessions.

6 Summary and conclusions

This chapter summarizes whole research, conclusions of the research based on the literature review and empirical case study. After conclusions is covered the importance of this research and in the end, topics for future research.

6.1 Summary

The goal of this research was to find out success factors in distributed and agile software development. The topic is interesting because there is not much previous research done, but the use of distributed and agile software development has increased significantly lately, especially when considered distributed teams that work near each other without temporal difference. The literature review is presented in second chapter. Third chapter considered research methods which were used to conduct empirical case study and fourth chapter was about results of the empirical case study. Chapter number five discussed about gathered results and compared results between previous study and this empirical research. In the last chapter, summary, conclusions and future research ideas were presented.

The topic was approached using literature review and empirical case study research. The topic was addressed in literature from previous studies related to distributed development, agile development and combination of those two. Studies about distributed and agile development were used to get better support to create theoretical basement for the research. After literature review, case study was conducted, results presented and conclusions summarized.

6.2 Conclusions

This section is about conclusions made based on conducted research. This research aimed to answer what are the success factors in software development project which is distributed and agile. The research question was:

> What are success factors in distributed agile development and what experiences about this combination already exists?

The literature review summarized what is already researched about distributed agile development and success factors in it. Literature was reviewed, to get results and existing experience about success factors in distributed agile development. Based on earlier studies, success factors constructed of communication, continuous process, competent team and building trust. Communication played critical role both in distributed and agile development and also in distributed agile development. Second factor was sum of continuous process, trainings and competent team. Third success factor area was trust. It included building trust between distributed teams, internally inside development teams and with other stakeholders.

Empirical research agreed partially with results of the literature review. Communication, trainings and team cohesion were mentioned in both, roles and tools were mentioned only in empirical research. Continuous process, competent team and trust were mentioned only in literature review. So, based on empirical research most critical success factors in distributed agile development were different levels of communication, people working in correct roles, team spirit, technical tools and trainings.

Most of the interviewees highlighted face-to-face communication as the most important success factor. Face-to-face communication resulted as more efficient and informal communication which gave better common understanding throughout the team. Second important communication way was communication inside development team. This communication inside development team resulted as transparency and broader understanding about whole development project. Especially, product owners raised the factor that correct team members are working in correct roles, which means that every team member is having enough challenge in everyday tasks and enough competence to succeed in those tasks. This success factor is close to competent team factor, which was highlighted in the results of the literature review. Technical tools were mentioned as success factor and that is more of an enabling factor, because without proper technical tools distributed and agile development work could not be efficient. So, based on this thesis success factors in distributed agile development were communication, trainings, team cohesion, competent team, team members roles, technical tools and continuous process. In next section is covered how these factors can be established in practice.

6.3 The results in practice

In this section is presented how results of the research affects for development teams, team members and software development organizations lives in practice. Fact is that distributed and agile development has increased during past years, so it is important for organizations and teams to understand where to concentrate on achieving success.

From organizations point of view, it is important to understand that some of the success factors results from some other and in a way factors are connected for each other. For example, technical tools enable distributed team to communicate effectively and through that communication team members are able to create trust between each other and that results in team cohesion. On the other hand, trainings support culture where communication is highlighted, agile roles are known and team pursue for continuous improvement resulting competent team where team members are in correct roles.

This thesis supports observations of communication, trainings and team cohesion as success factors and provides correct roles for team members and technical tools as new success factors for distributed agile development. Continuous process, competent team and trust were not agreed as such in empirical research, but competent team can be result of trainings and correct team members working in correct roles. Trust can be result of informal communication and if team cohesion is high also trust is present. Continuous process is not mentioned in empirical research but interviewees were team members so it might be challenging to see this kind of success factor from the executing level. Next section presents relevant topics for future research based on this research.

6.4 Topics for further research

This section states interesting research topics which were not covered in this thesis but are related to this area of research. The topics for future research are based on the observations and results done.

Based on the interviews one main topic that should be investigated more thoroughly is different levels of agility and different levels of distribution and how those different levels affect towards each other. In this thesis cultural difference and distribution was quite small, but would be interesting to research how much bigger cultural difference would affect agility. Quite similar issue is temporal difference, because that challenge communication and so can make instant communication almost impossible.

One more specific research topic relates to informality felt inside development team. Because in this research was mentioned that informal communication creates trust and increases team cohesion, so would be interesting to understand how to increase this informal communication and how big is the impact for the success.

Last topic for the future research would be related to connections between success factors in distributed and agile development. Because based on the conducted research seems that some success factors are more of enablers and other ones are resulting from some other success factors. Research should concentrate on factors that make then other success factors possible, for example technical tools make efficient communication between distributed locations possible, but it does not ensure that communication is informal which effect on the level of the team cohesion.

REFERENCES

Agerfalk, P. J., Fitzgerald, B., Holmstrom Olsson, H., Lings, B., Lundell, B., & Ó Conchúir, E. (2005). A framework for considering opportunities and threats in distributed software development.

Bullen, C. V., & Rockart, J. F. (1981). A primer on critical success factors.

Carmel, E. (1999). Global software teams: collaborating across borders and time zones. Prentice Hall PTR.

Chow, T., & Cao, D. B. (2008). A survey study of critical success factors in agile software projects. Journal of Systems and Software, 81(6), 961-971.

Cohen, D., Lindvall, M., & Costa, P. (2004). An introduction to agile methods. Advances in computers, 62, 1-66.

Conboy, K. (2009). Agility from first principles: reconstructing the concept of agility in information systems development. Information Systems Research, 20(3), 329-354.

Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. Journal of Systems and Software, 85(6), 1213-1221.

Ebert, C., & De Neve, P. (2001). Surviving global software development. *IEEE software*, *18*(2), 62-69.

Erickson, J., Lyytinen, K., & Siau, K. (2005). Agile modeling, agile software development, and extreme programming: the state of research. Journal of database Management, 16(4), 88.

Evaristo, J. R., Scudder, R., Desouza, K. C., & Sato, O. (2004). A dimensional analysis of geographically distributed project teams: a case study. *Journal of Engineering and technology Management*, *21*(3), 175-189.

Fowler, M., & Highsmith, J. (2001). The agile manifesto. Software Development, 9(8), 28-35.

Freudenberg, S., & Sharp, H. (2010). The top 10 burning research questions from practitioners. *Ieee Software*, *27*(5), 8-9.

Grover, V., Cheon, M. J., & Teng, J. T. (1996). The effect of service quality and partnership on the outsourcing of information systems functions. Journal of Management Information Systems, 12(4), 89-116.

Highsmith, J. A. (2002). Agile software development ecosystems (Vol. 13). Addison-Wesley Professional.

Hirsjärvi, S., Remes, P., & Sajavaara, P.(2007). *Tutki ja kirjoita* (13. p.). Tammi.

Jiménez, M., Piattini, M., & Vizcaíno, A. (2009). Challenges and improvements in distributed software development: a systematic review. *Advances in Software Engineering*, *2009*, 3.

Khan, S. U., Niazi, M., & Ahmad, R. (2009, July). Critical success factors for offshore software development outsourcing vendors: A systematic literature review. In *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on* (pp. 207-216). IEEE.

Kiel, L., & Eng, P. (2003, May). Experiences in distributed development: a case study. In *International Workshop on Global Software Development at ICSE* (pp. 44-47).

Lee, G., & Xia, W. (2010). Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility. Mis Quarterly, 34(1), 87-114.

Lui, T. W., & Piccoli, G. (2007). Degrees of agility: implications for information systems design and firm strategy. Agile information systems: Conceptual-ization, construction, and management, 122-133.

Misra, S. C., Kumar, V., & Kumar, U. (2009). Identifying some important success factors in adopting agile software development practices. Journal of Systems and Software, 82(11), 1869-1890.

Phalnikar, R., Deshpande, V. S., & Joshi, S. D. (2009, January). Applying agile principles for distributed software development. In Advanced Computer Control, 2009. ICACC'09. International Conference on (pp. 535-539). IEEE.

Prikladnicki, R., Nicolas Audy, J. L., & Evaristo, R. (2003). Global software development in practice lessons learned. *Software Process: Improvement and Practice*, *8*(4), 267-281.

Ramesh, B., Cao, L., Mohan, K., & Xu, P. (2006). Can distributed software development be agile?. *Communications of the ACM*, *49*(10), 41-46.

Sheffield, J., & Lemétayer, J. (2013). Factors associated with the software development agility of successful projects. International Journal of Project Management, 31(3), 459-472.

Shrivastava, S. V. (2010). Distributed agile software development: A review

Siau, K., Long, Y., & Ling, M. (2010). Toward a unified model of information systems development success. Journal of Database Management, 21(1), 80-101.

Šmite, D. (2006). Global software development projects in one of the biggest companies in Latvia: is geographical distribution a problem? Software Process: Improvement and Practice, 11(1), 61-76.

Takeuchi, H., & Nonaka, I. (1986). The new new product development

game. Harvard business review, 64(1), 137-146

APPENDIX 1 INTERVIEW STRUCTURE

Definition for project to be understood as one release which lasts 3-4 months.

Questions:

1. How agile this project is?
    - How agile this project is in process, people and structure agility?
2. What restricts this project to be more agile?
3. How distributed this project is?
    - How distributed this project is as geographically, culturally and temporarily?
4. What challenges you have faced during this project?
    - Are those challenges related to distribution or agility?
5. When project is successful?
6. What are success factors in distributed agile development?
7. How would you improve this project?

APPENDIX 2 HAASTATTELURUNKO

Projektin määrittely yhden julkaisun, mikä kestää noin 3-4 kuukautta, mittaiseksi.

Kysymykset:

1. Kuinka ketterä tämä projekti on?
    - Kuinka ketterä tämä projekti on prosessiketteryyden, ihmisketteryyden ja rakenneketteryyden näkökulmasta?
2. Mikä rajoittaa tätä projektia olemasta vielä ketterämpi?
3. Kuinka hajautunut tämä projekti on?
    - Kuinka hajautunut tämä projekti on maantieteellisesti, kulttuurillisesti ja ajallisesti?
4. Mitä haasteita olette kohdanneet tämän projektin aikana?
    - Ovatko ne liittyneet hajautumiseen tai ketteryyteen?
5. Milloin projekti on onnistunut/menestynyt?
6. Mitkä ovat hajautetun ketterän kehittämisen menestystekijöitä?
7. Kuinka kehittäisit tätä projektia?