TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

TIMO NIKKILÄ

AGILE IN PUBLIC RESEARCH PROJECTS

Master of Science Thesis

Examiner: Professor Kari Systä
Examiner and topic approved by the Council of the Faculty of Computing and Electrical Engineering on 9 April 2014

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Information Technology
NIKKILÄ, TIMO: Agile in Public Research Projects.
Master of Science Thesis, 80 pages, 0 Appendix pages
May 2014
Major: Software Engineering
Examiner: Professor Kari Systä
Keywords: agile, team, public research, project management

Public research relies on projects. Knowledge and awareness of project management have increased in software development in last few decades. At the same time project management has become one success factor in enterprise world. It is reasonable to believe that project management is similarly important in public research projects. Agile is a project management and teamwork methodology used in software development world. This thesis examines public research projects and the way they are already align with agile values and the way agile values and techniques might help projects and teams to improve.

In high level the thesis is divided into two parts. In the literature chapters study agile values and most general techniques are explored. In the project part a few public research projects are studied by interviews and free-form talks and by observing how projects are run.

PREFACE

Professor Kari Systä examined this work.
I thank all the participating projects at Tampere University of Technology and Tampere University. Furthermore I thank my good friend Heini Kallio who helped me to find out what I want to study for thesis. I also thank Olli Sorje and Valtteri Pirttilä and all the other collages who have been teaching me how healthy and fruitful group work is done in action.

Terms and Definitions

epic	Collection of multiple features under the same theme.
on-site customer	Customer representative co-located with the team.
product owner	Vision owner and customer representative.
time-boxed	Allocating fixed time period for planned activity.
TUT	Tampere University of Technology
UTA	University of Tampere
VTT	Technical Research Centre of Finland
iteration	The act of repeating a process with the aim of approaching a desired goal.
sprint iteration	Agile practice called iteration.
practice	Single practice, such as daily stand-up in agile.
methodology	Used collection of practices, such as SCRUM in use.
framework	Bare bone set of practices that can be used to implement methodology. For example SCRUM is a framework.
Team	The team means the same as the people working in a project.

Contents

1 Introduction

Traditionally project work has been done through hierarchical command chains, where project manager is responsible for the whole and assigns tasks to team members. Waterfall model is often thought to be a traditional software development model. Waterfall is a sequential design process, in which development is done through phases like conception, initialization, design, construction, testing, acceptance testing and maintenance. This approach might work in some environments such as building houses or working in assembly line. However, it does not work in inherently complex industries, such as software development. The more complex your projects are the more professionalism it takes to build and the more dependent you are on motivation, on judgment of many and on reasons. The more complex your environment is the more learning and innovation is required. Interactions drive innovations and learning [2]. Agile has emerged from this ground. Many software teams have succeeded by using agile. Even more often teams have failed their transition to agile. Most of failed projects have not been catastrophes, they may even have had some benefits of using agile practices. However, agile is more values lived well than a collection of principles. Agile transformation failures are the motivation of this thesis.

Agile was born in a meeting in the year 2001. Representatives from XP, Scrum, DSDM and Adaptive Software Development in agile held a meeting. They discussed and examined common ground in all the light weight methodologies used in software development. Then they labeled those principles as agile. Agile, therefore, is a collection of often used practices, values and frameworks that have been proven to be successful with multiple teams and multiple projects. [43] Even though agile was established over a decade ago, it has numerous definitions [42]. Jim Highsmith describes an idea and value of light weight methodology: "*Both scientific and management researches have shown that a simple set of rules can generate complex behaviors and outcomes whether in ant colonies or project teams. Complex rules on the other hand, often become bureaucratic.*" [2]

Agile emerged from the software industry. Software aspects can be seen in agile principles, manifesto and techniques developed under agile. Still, agile is mostly about human interactions and human attitudes needed for working in a complex environment. The research world relays heavily on professionalism and intelligent people. Research is also complex by its nature. You need to be innovative to build theories and you need to test them and try to learn along the way. There is a good reason to believe that agile would work in the research world as well. Agile has even been used in research before. Additionally, there is a Linked In group called *Agile Research*, where people discuss

agile usage in research. Regardless, agile has had niche role in research so far. This thesis is about to study public research projects: how they are already working aligned with agile values and how could they improve by using agile.

Frameworks, such as Scrum, XP or Crystal Clear are not described or recommended by the thesis. Even though it would be much easier to start with a framework, there are reasons to avoid that approach as well. First, if a group is lacking some traits needed for agile, such as self-organization, then there is a big risk that installing agile would lead to chaos [36]. Second, a good coach is extremely important for successful agile transformation. It is unlikely that public research teams would find and hire a good coach. Third, the research world is a somewhat different in nature to software development. The research world also has strong conventions, such as many one person projects. Agile is not designed for one person projects, but for teams. In this kind of environment it works better to try to understand the core values of agile and have repeated baby-step improvements. Many small steps combined make great change. Some authors warn from making your own agile practices, when you are not familiar with agile [8].

The thesis is divided into five parts. First part describes agile on a theoretical level. Second part describes thesis's viewpoint to agile. This part is very psychological and it highly respects the fact that agile is mainly about people. Third part describes some agile practices. It is not a comprehensive list of practices, but a few practices that are referenced later on. Fourth part describes the bare bones of what it takes to be successful with agile. Fifth part introduces few public research projects and their project management.

2 Theoretical Background to Agile

Compared to waterfall, agile is a light weight methodology for software development. It is a transformation from a process based on anticipation, such as waterfall, to a one based on adaptation [2]. Working with agile is feeling different when there are 500 people doing it, than when there are 5 people doing it. Bigger projects need more structure [2]. **Only small teams are dealt with in this thesis, since the studied public research projects are all relatively small.**

The traditional way of creating software is to have a plan and three constraints: cost, schedule and scope. If you are able to build accordingly to the plan whiting constraints, then your project was successful. Otherwise it was not successful. Traditional way to do software projects is to plan and build. The aim of overwhelming planning is to be more accurate and to reduce uncertainty and so forth making an accurate budget, scope and schedule. However, benefits for extra planning start to diminish the more you plan. Figure 1 illustrates this [10].

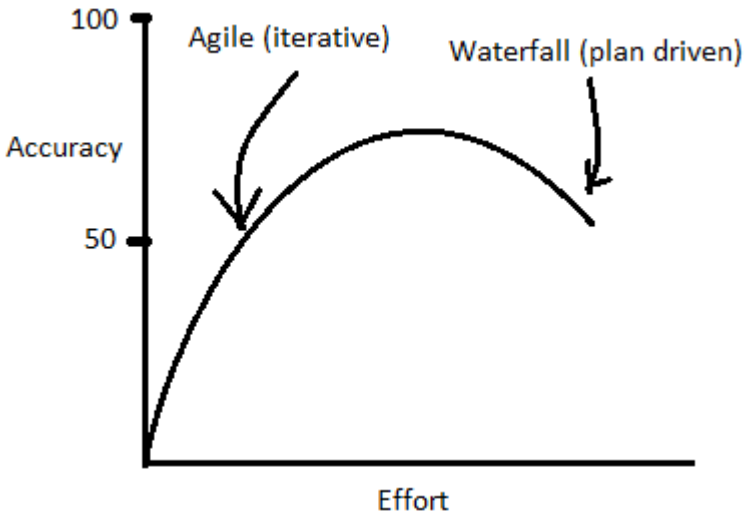

Figure 1. Additional estimation effort yields very little value beyond certain point. [10]

The agile way of doing software is to speculate and adapt. Plans are speculations, which are tested by frequent deployment and feedback. Plans are updated when something new has been learned about market needs or project progress. Updating plans based on new information is called adapting. Agile replaces three traditional success factor constraints (cost, schedule and scope) with value, quality and constraints. Figure 2 shows the classic constraints triangle and how it changes when agile is used. Agile replaces following a plan and correcting to a plan, with focusing on business value. The plan is corrected whenever new information is learned about business value. If project plan is massive and goes into the details, then correcting a plan gets very burdensome. Agile recognizes this by minimizing upfront planning. The client buys a team to develop software, because the client has some needs. Agile does not require the client to know his every need at the beginning of the project. Even with traditional development, all the real needs cannot be recognized upfront. So creating long and detailed list of needs upfront is wishful thinking at best. More often they are waste of time and source of contract negotiation and arguing that leads to missing the business value. Simple design means valuing adaptation over anticipation [2]. Needs are learned along the way as the product is developed and tested. Some questions need to be answered beforehand, such as the feasibility of the project. Agile does not remove this type of planning. However, agile accepts that even project feasibility decisions can change, when more is learned during the project. The agile principle of failing fast works here as well. The sooner it is learned that the project is not feasible, the better. Then sooner the project can be cancelled, the more time and resources are saved. Project cancellation can be a reason for celebration. So when traditional project management object change, agile is fostering changes for delivering business value. The attitude of doing barely enough is not used only for planning, but all over in agile.

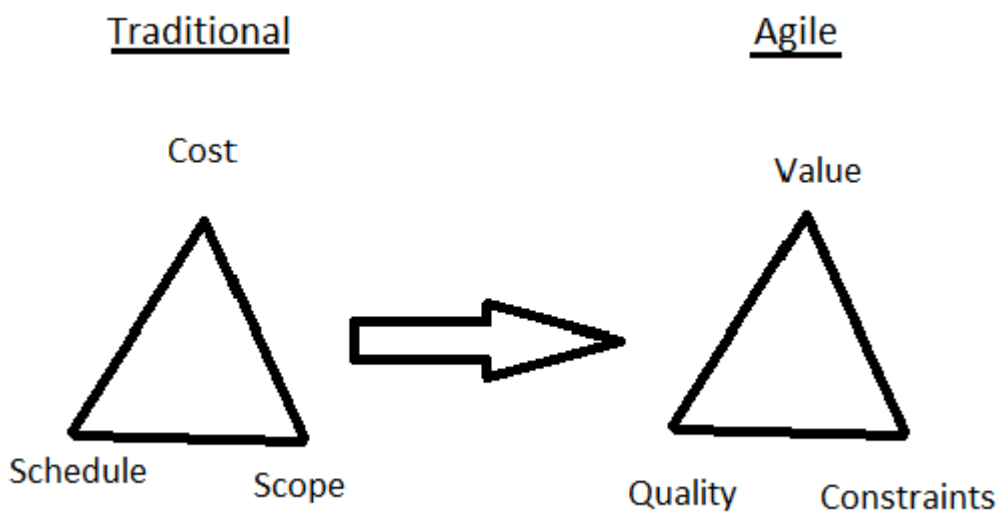

Figure 2. Traditional constrains triangle on the left side. Agile replacement for traditional triangle is on the right side. [2]

Just like any other methodology, agile has its sweet spots. When used correctly in a sweet spot, agile brings all the value one can get from the framework. The further one gets from the sweet spot, the less value is achieved. Agile sweet spot is in a complex environment [34]. Complex environment means environment with characters of both chaos and order. This kind of environment is called Chaordic. If the working environment has no chaos, then there is no need to speculate, experiment, and adapt. Optimization can be used instead. On the other hand, if there is only chaos, then anything cannot be plan, there is no time or point to speculate or make experiments. If a solution is experimented now and it works, the same solution tried again ten seconds later might not work. Future plans are not done in chaotic environment, but events are reacted to as they happen. To be in a house that is on fire is an example of chaotic environment. To simplify, there are two sources of chaos in software development: unknown technology and changes in specifications. Figure 3 illustrates agile sweet spot whiting these two variables.

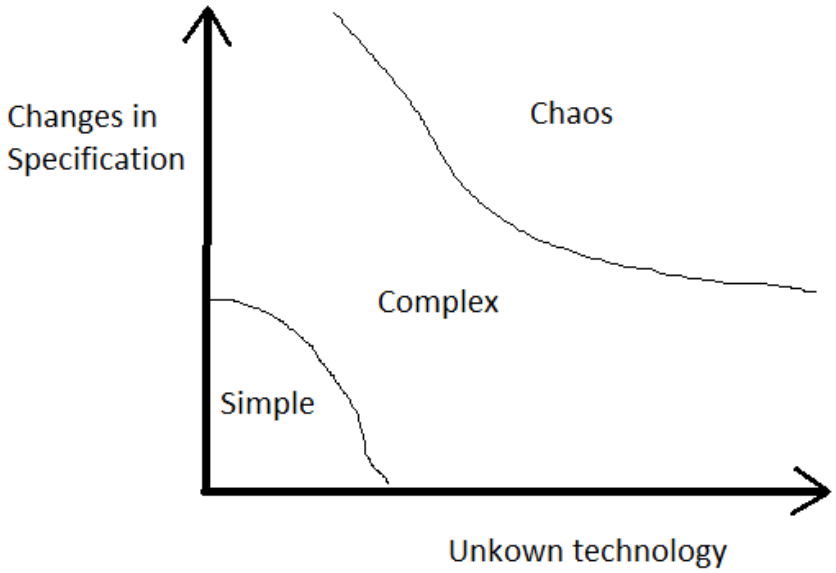

Figure 3. Agile works best in a complex environment. Complexity is found between chaos and simplicity. There are two sources of variability in the figure: Unknown technology and changes in specification.

As mentioned before, some planning is always needed. Some projects need more planning than others. For example, failing to properly design an airplane can lead to the loss of lives. Loss of lives is not acceptable. Not even when aggressive user testing improves progress. On the other hand, too much planning on a start-up web service will lead to running out of money before going live. Thus, planning is a balancing act between the damage of planning too much and the damage of not planning enough. Right amount of planning depends on the problem at hand. The more damage is caused by not enough planning, the more favorable plan-driven development is. The more

damage is caused by too much planning, the more favorable is an agile development. Figure 4 illustrates this.

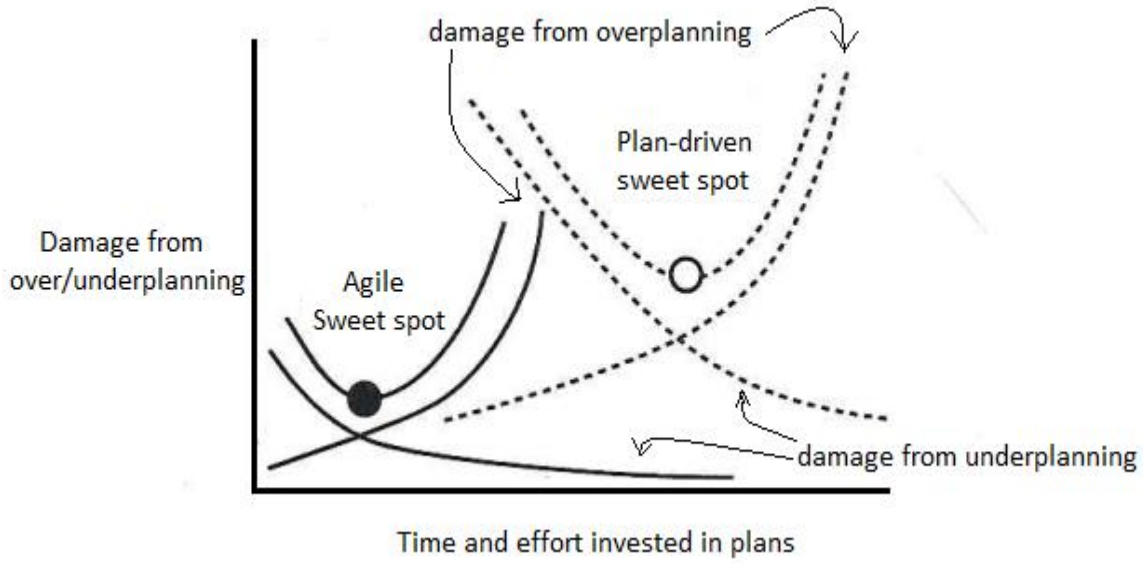

Figure 4. Different projects call for different amounts of planning. The less damage from underplanning and the less time and effort invested in plans, the more the situation favors agile. [7]

Rest of the chapters describes few key aspects of agile the reader should know before reading the rest of the thesis. List of agile aspects is not comprehensive, but adequate for the thesis. First, what are agile business objectives? Agile has no absolute value. It should be used only if it can deliver enough business value. Chapter *Agile Business Values* answer business value issue. Agile manifesto and its twelve principles are what many consider, the core of agile. The manifesto and the twelve principles are described in chapter *Agile Manifesto and Principles*. Agile is passionate about value adding. Chapters *Customer and Vision*, *Continuous Value Adding* and *Lean* all explore an angle of adding more value. Finally, the chapter *Management in Agile* answer the questions about a manager's new role in agile.

2.1 Agile Manifesto and Principles

What is agile? This chapter tries to give an answer to that question. Agile manifesto and its twelve principles were produced when agile was established. The manifesto is the heart of agile and the principles describe what is behind the manifesto. When you are new to agile, you start with an agile framework, such as Scrum. You then follow mainly the framework, not manifesto. While you are gaining deeper understanding, you should move from framework to manifesto. [31] Agile manifesto:

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

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

That is, while there is value in the items on the right, we value the items on the left more.“ [33]

Agile manifesto is rather compact and self-explanatory. Agile does give value to processes and tools, justified documentation, contracts and plans. However, human assessment and creating value for customer are valued even more in agile. So, agile highly values individuals and interactions, working software, customer collaboration and responding to change.

Agile manifesto is a high level guideline to agile. Twelve principles go into more detail and are great guidelines for project management and teamwork. Yet, the twelve principles are far less known than the manifesto. The twelve principles are only listed here. This thesis will not clarify their meaning. Reading them through is enough to understand the rest of the thesis. Twelve principles go as follow:

“*We follow these principles:*

- *Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.*
- *Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.*
- *Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.*
- *Business people and developers must work together daily throughout the project.*
- *Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.*
- *The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.*
- *Working software is the primary measure of progress.*
- *Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.*
- *Continuous attention to technical excellence and good design enhances agility.*
- *Simplicity--the art of maximizing the amount of work not done--is essential.*
- *The best architectures, requirements, and designs emerge from self-organizing teams.*
- *At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.*”[45]

There is also a draft proposal for agile manifesto and twelve principles for research. Those were written by Xavier Amatriain [5]. Reader should notice that agile manifesto and principles have been used countless times and are debated widely across the software industry. Amatriain's agile research draft is almost unknown and gone through almost without public debate. It is still an interesting draft and Amatriain's agile research manifesto is shown here:

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

- *Individuals and interactions over processes and tools*
- *Real-world working solutions over comprehensive theories*
- *Commitment and response to social needs over obtaining grants, patents, or publications*
- *Responding to change over following a plan*

That is, while there is value in the items on the right, we value the items on the left more."[5]

Amatriain has changed two agile manifesto rules in his manifesto. First, theories are equated to documentation. When it comes to explorative research, instead of building comprehensive theories we should take the most important untested part of what we want to prove and test it in the real world. For example, instead of arguing if Scrum has it place in research, we should give it a try to see how it works. Second, he claims that commitment and response to social needs should be more important than obtaining grants, patents, or publications. This can be an important notion. Patents and publications are used as important meters of reward in research. If they are not measuring how well research is serving social needs, then they are source of dysfunction.

Amatriain has also developed a Scrum-like agile scientific framework which can be found from his blog [6]. This framework is not studied in this thesis. However, it is interesting to read, if the reader is interested in using a Scrum-like framework in science.

2.2 Agile Business Objectives

Agile would not be vital methodology if it didn't bring any business value. This makes it important to sufficiently define eligible business values of agile: [2]

1. Continuous innovation
2. Product adaptability
3. Improved time to market
4. People and process adaptability
5. Reliable results

2.2.1 Continuous innovation

Developing software in today's complex world requires a mindset that fosters innovation. There are multiple sources of complexity. Here are a few sources listed: clients not knowing well enough what they want, developers not knowing well enough how to build it, changes in business environment and resources and new feedback changing the requirements. Building software is continuous cycle of learning and innovating solutions to oncoming challenges. Innovations are generated in culture based on the principles of self-organization and self-discipline.

Innovations are important to research projects. For example explorative research projects start by building a theory and then testing the theory. Building the theory is complex process. New way of thinking may be needed. New theories are often built up in human interactions [2]. First they are built and tested through rigorous thinking of many. When enough confidence has been obtained, a test plan is built, funding requested and the theory is put to test. Even if test practices may be well defined, all sorts of challenges occur. Finally, results may or may not confirm the theory. Many times results points to another direction and new theories are built. This all in mind, continuous innovation can be seen as business object also in research world.

2.2.2 Product adaptability

The future can be predicted, but it will always surprise us. For some products, changes in market, technology or specific requirements happen weekly. For other products, changes happen rarely. Still, changes happen and the product needs to be able to adapt these changes. The product needs to deliver customer value today and be able to adapt to tomorrow's needs. [2]

Public research projects rarely create a product. Some projects might create a product as a side effect of testing a theory. Do side effect products have changing business needs? Do they need to change over time? Is there a maintenance period for side effect products? Perhaps side effect products do not have to change over time in many cases. This way of thinking, product adaptability is not always a concern for the research world. If software development is thought as theory building, then tested program or product could be thought of as a tested theory [1]. With this analogy, public research projects do also produce a product. The product is a tested theory. This is an interesting point of view, but how much can or should researchers try to increase the quality of theory so that it would be adaptable in the future? Generally simpler theory is better. The need for product adaptability in research world can be argued either way.

2.2.3 Improved time to market

It is crucial to meet the market window in software development. Tough competition makes the market window short. Agile improves time to market by increasing focus, streamlining and skill development. To simplify the meaning of focus is to build the

highest business value features first and publish whenever smallest marketable value is added. Streamlining is to concentrate on value adding activities and eliminating overhead and compliance activities. Skill development is to improve time to market by choosing people with right skillset and molding them into productive team. Right skillsets and gained experience while working and healthy teamwork should improve skill development. [2]

It is inarguable that time to market is important for public research projects. There are many competitive research teams. If two teams are testing the same theory, then often the one that public first will have they publication published on better papers and get most of the honor. One might argue that the goal of publish research is not to gain honor, but to serve public. Still, this reward system is real and it must be accepted that the research game played aligns with reward rules. Time to market also has a role when it comes to serving the public. The sooner the research is ready, the sooner its results can be used. For example, the sooner a new treatment for cancer is ready, the sooner it can be used to save a life.

2.2.4 People and process adaptability

If we want adaptable products we first need to build adaptable teams that view change as part of a dynamic environment. Agile encourages teams to think of learning and adapting as an integral part of delivering value. [2]

How much do people and processes in public research world need to adapt over time? Arguably team sizes, roles, techniques, financing models, rewarding systems and so on are changing over time and new information is coming in all the time. All this affects how the work should be done. Therefore people should be adaptable. The needed rate of adaptation is ambiguous. Even if not much adapting is needed, or even accepted in some cases, better adapting can still have a competitive advantage.

2.2.5 Reliable results

Agile is an explorative process. Because of all the unknown aspects that comes during the project, agile will not deliver completely pre-specified results, but it will deliver valuable results – time after time. A repeatable process delivers the same result time after time when given the same input. A reliable process delivers value within set target boundaries regardless of the impediments thrown in the way. So agile is reliable, but not repeatable. [2]

Research is certainly explorative in its nature and obtaining reliable and valuable results is a meaningful goal for research. Some research experiments need to be more repeatable than other. For example, exploratory research does not have much value, if no one knows how the theory was tested. So, in a way it needs to be repeatable. Agile is not trying to water all repeatable down. How an experiment was done can still be described in repeatable way even if the team was solving impediments thrown in the way in unrepeatable manners.

2.2.6 Do business objectives justify further study

Even if it could be study on its own, agile business objects fit rather well for public research projects. It seems like agile could bring value to public research projects and thus further studies are legitimated. For those who have tried agile and think it does not work: it is worth remembering that agile is easily and commonly misinterpreted, and when done so, business values will not be met.

If programming is viewed as theory building, then there is a strong link between the research world and software development [1]. Big part of explorative research is building theories, testing them and then rebuilding those theories, testing them and so forth. Software development is also about building theories and testing them in a loop. Viewing programming as theory building makes agile usability studies more attractive for the research world.

2.3 Customer and Vision

Having a clear vision is crucial for a project to succeed. Creating such a vision is hard and requires leadership. The absence of clear vision will cause agile projects to spin-off into endless experimentation. The customer owns the vision and it is crucial that the vision is made clear to development team. [2] Making the vision clear is an ongoing process throughout the project. The customer is therefore part of the agile team.

Creating a shared vision at the beginning of the project might contain creating a vision box, an elevator statement, a project data sheet and a release plan together with the team. [2] Following questions should be answered during the visioning phase [2]:

- What is the customer's product vision?
- What are the key capabilities required in the project?
- What are the project's business objectives?
- What are the project's quality objectives?
- What are the project's constraints (cost, scope, schedule)?
- Who are the right participants to include in the project community?
- How will the team deliver the product (approach)?

Who is the customer? The classical way of thinking is that the one hiring the team to work for him is the customer. He has a vision he has hired the team to work on and he will be using the outcome the team is producing. Some see it differently. They claim that people are not only hired to work for the buyer. Everyone who is participating in a project should have willingness to change the world and some idea of where the world should be going. If the values of a team member and a sponsor are not aligned, then they should not work together. Agile, done well, cannot be separated form values lived well [31]. In this way of thinking, all stakeholders are customers [28].

Whoever the customer is, the vision needs to be clear. In many cases there may not be a real customer in public research. There are multiple ways to deal with such a situation. For instance there can be one individual who is ultimately responsible for the

vision. The whole team participates, but one person is ultimately responsible. In other cases there can be a customer team. A team of key participants, such as professor, a seasoned researcher and an industrial partner are together the vision owners. They make the vision clear to everyone.

2.4 Continuous Value Adding

The value created is increased through a continuous flow of value [2]. Let us expand on this, since it is one of the central ideas in agile.

Imagine a traditional way of creating software, where the whole software is delivered at once, once everything is complete. If the project lasts a year, then for the first year money is being spent and no business value is being delivered. For the first year, the risk is only getting bigger and bigger. See figure 5 - Big Bang. To improve the situation, created software can be delivered periodically or in other words incrementally. Client is receiving some value after the first period and some more value after the second period and so on. In this approach the product starts to pay itself back sooner. In addition, feedback is received earlier. See figure 5 - Big Increments. Adding value can be improved by shortening the delivery period. See the figure 5 - Small Increments. Since the value of the feature is not only proportional to the size of the feature, then even more optimization can be done. Features are prioritized. Business value is usually the main factor in prioritization. Business value consists of the financial value, the cost of development, the amount of significant learning and the amount of risk removed [10]. When highest business value features are done first, then value curve looks like in figure 5 - Highest Value First.

Yet another trick to improve value curve is learning. Iterative shortens the feedback loop and therefore improves feedback. Improved feedback improves learning. [11] The last trick to improve continuous value adding, is to stop the project when the value adding curve is too flat. Only 20% of software features are always or often used. Up to 65% of features are rarely or never used. [12]

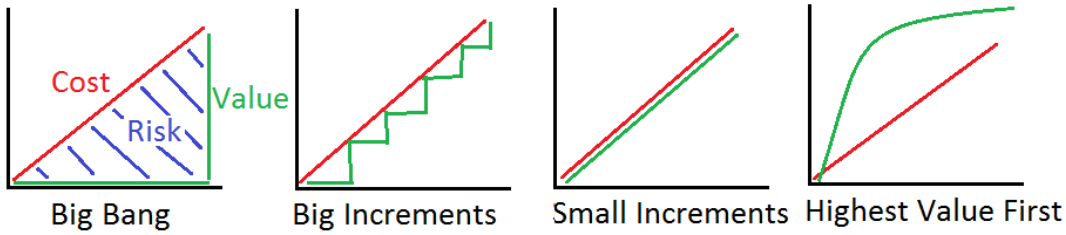

Figure 5. Improving the value curve. Delivering by small iterations and highest value first is delivers higher value and faster. [11]

2.5 Lean

Lean is not part of this thesis and is therefore described only shortly. Lean can be used to extend agile. The following lean ideas are explained since they may bring good supplementary value for public research projects: some aspects of removing waste and portfolio management. [13]

Lean has an obsession with removing waste. Any activity that does not deliver value to the customer is considered waste [2]. Storing intermediated products is waste according to lean principles. Work should be done just in time when it is needed and the product should be brought from idea to value as fast as possible [13]. This is done by removing delays from the process rather than keeping everyone as busy as possible. [13] Agile implicitly advices commitment deferring, but lean makes commitment deferring explicit. Decisions should be made at the last responsible moment, no earlier and no later. [13]

Portfolio management consists of selecting the most important features from the most important products to create and enhance [13]. In other words, project portfolios are idea inventories. The project portfolio includes everything from planning the life cycle at the very beginning, to the removal of the product at the end of the life cycle. Ideas often become less valuable over time. Consequently, the key purpose of a portfolio is to make sure that the highest business value idea is done first and conceived from idea to value as fast as possible. [13] This directs us to define as small projects as possible. [13]

Organization needs to remove delays to deliver quickly. Once major source of delay is removed, the second major source of delay becomes major source of delay and it gets the attention it deserves. This mechanism is exposing smaller and smaller delays. So organizations that deliver quickly expose delays. Delays decrease both effectiveness and efficiency. When delays are exposed, they are easier to remove. Fewer delays lead to better time to market. The portfolio minimizes the work in progress. Usually the quality of the work also improves, since shortening deployment time exposes bad quality, such as a messy manual deployment process. Shorter cycles are also a good tool for removing wasteful steps. When focus is on speed, team needs to understand what they are building and they can therefore avoid building things that are not needed. [13] The portfolio is a line of sight to business needs. It creates visible representation of business features with priority and technical effort. [13]

Portfolio management is closely bound with an organization's resource management. Some even say that product focus should be used instead of project focus. This would encourage staffing for the entire life of the product instead of staffing for the project. [13] The book Lean-Agile Software Development summaries the problem with many projects: "*When many projects are in process, the contention for resources becomes so great that projects get scheduled based on recourse available or political clout rather*

than what would contribute the most value to the business. We've seen extreme cases where it seems that teams don't even exist; rather, there are several people working on a project together while they also work on other project with other people. In organizations like this, creating a business focus is often the impetus for creating effective teams. Project thinking virtually guarantees inefficient use of personnel." [13]. Studies have shown that working on multiple projects at the same time, decreases efficiency significantly [13].

2.6 Managers in Agile

The role of management is important, but different in agile software development than in waterfall development. In agile, the project leader's style is leadership-collaboration rather than command-control. Leading and creating a collaborative work environment is more difficult and more rewarding than commanding. Agile project leaders are champions of the vision that has both customer focus and technical focus. They need to articulate the vision so that everyone understands it, and nurture it so that no one forgets it. Leaders help the team to deliver by protecting them from distractions, such as unnecessary compliance work. Leaders are responsible for staff selection, staff development and ongoing encouragement. They also help to create a safe environment where collaboration, participatory decision making, interactions, conflict resolutions, fierce debates and collegial respect can flourish. [2] Facilitation skills are one important tool for success.

Project leaders also have responsibilities outside the team. He has to meet with the customer, executives and other stake holders to set and meet their expectations and to educate them on how to participate as partners with the team. [2]

3 Agile Techniques

It is a general tendency that people have to invent a new solution rather than researching previous solutions. [7] This chapter introduces a few popular agile practices. The list is not meant to be a comprehensive collection of agile practices. It is meant to describe only a few practices that the studied research projects could benefit from implementing.

3.1 Work Flow Visualization

This chapter introduces two tools: First, backlog used for prioritizing, estimating and planning work. Second, a work flow table for tracking task flow from state Backlog to state Done.

Backlog is a commonly used term in agile. Sprint iteration has a backlog and the product has a backlog. Backlog is a list of epics, features, capabilities or tasks. It is a list of work items that should be done. Product backlog objective is to expand the product vision, through an evolutionary requirements definition process, into a product feature list. [2] Often it is a good practice to sort the table so that the features at the top of the list possess also the highest value. This way they can be easily selected to be dealt with next. Figure 6 illustrates product backlog.

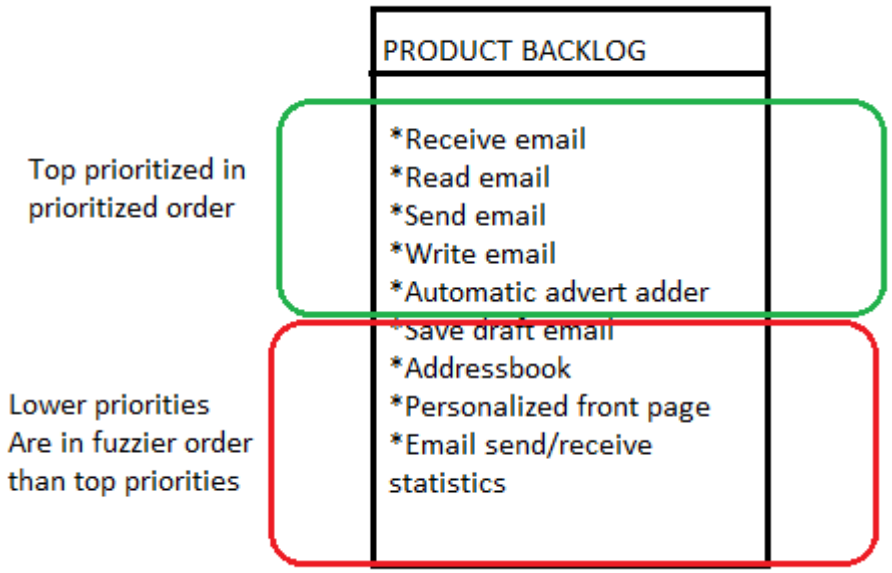

Figure 6. Product backlog. An example of a product backlog. The order of the stories shows priority. The higher in the list, the higher in priority.

Sprint iteration backlog contains work selected to be done in sprint iteration. Otherwise sprint iteration backlog and product backlog are almost equivalent. Figure 7 illustrates sprint iteration backlogs derived from product backlog at figure 6.

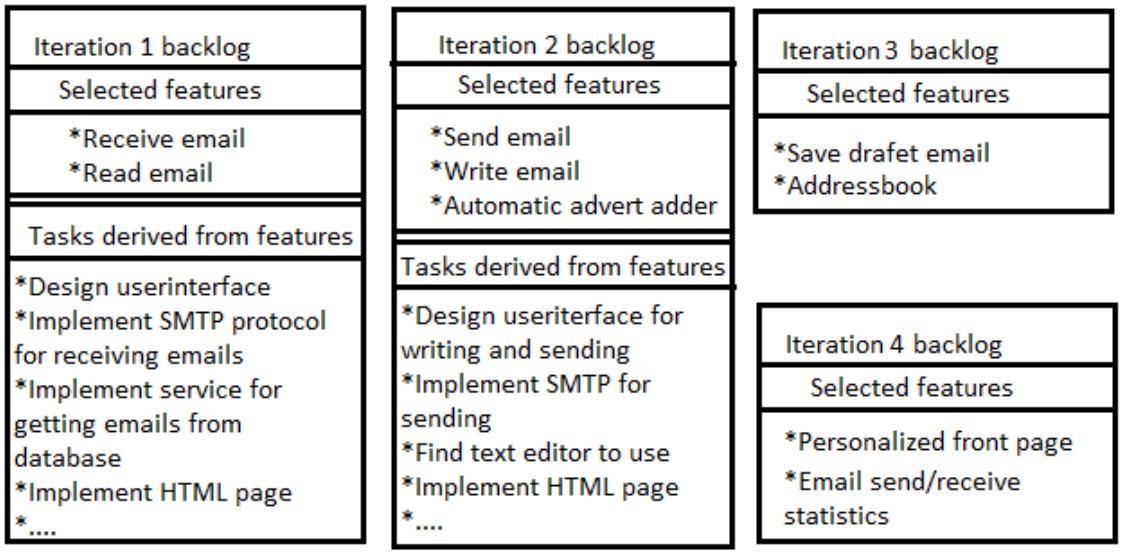

Figure 7. Sprint iteration backlog for the following four sprint iterations. A fairly detailed backlog for the next sprint iteration. The further in the future the sprint iteration is, the fuzzier the sprint iteration backlog is.

A workflow table can have many forms that serve different needs. The Kanban table is introduced here. Figure 8 shows an example of a Kanban table.

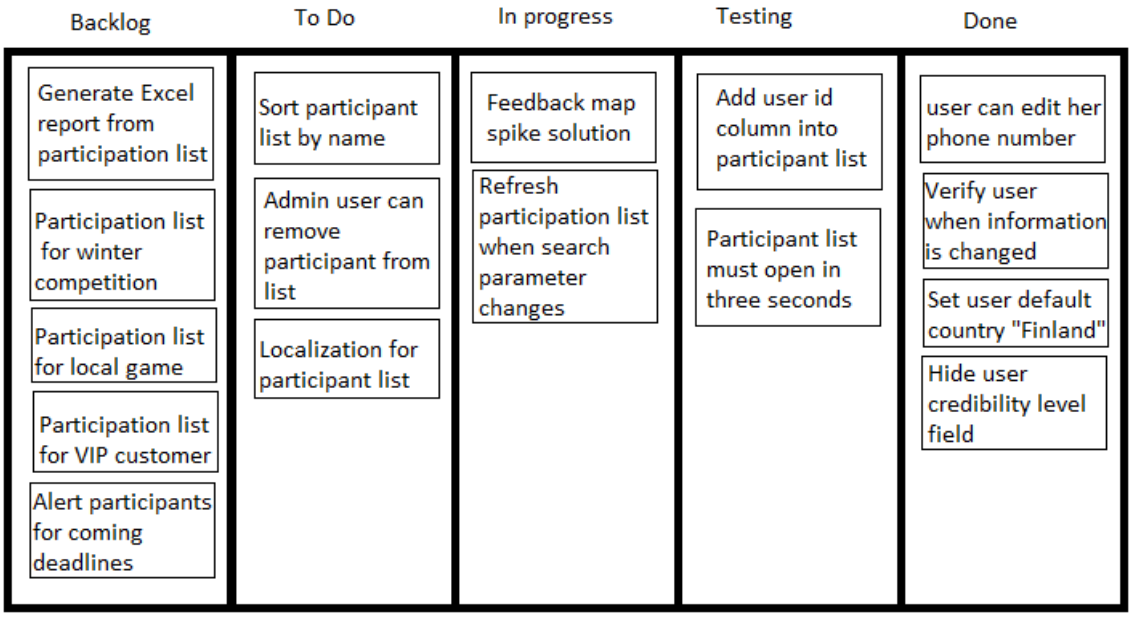

Figure 8. The backlog is an unordered list of identified features. When the team is given permission to build a feature, the feature is split into tasks and moved to the To Do state. To Do is an ordered list, so that the highest position has the highest priority. The In Progress state has work in progress limitation. One team may have a maximum of three works in progress at one time. The same goes with the Testing state. The Done

state shows features that are done and waiting to be delivered to the customer. When delivered, the task will be removed from the Kanban table.

The Kanban table limits the work in progress at any given time. Kanban does this by specifying a slot for each available type of activity and limiting the number of tasks in slots. For instance "To Do" and "In Progress" are activities in a figure 8. When a task in one slot is done, the card representing the work is moved to next slot in the workflow. For instance, a task can be moved from "In Progress" to "Testing" in a figure 8. The table always represents the current state of work. It also shows both process and status with minimal effort. [13 p98] Even if Kanban is not time-boxed, the Kanban table or Kanban table variations can be used in time-boxed development as well.

It is extremely important to realize that Kanban table should visualize process as it is. For instance, consider that task X that takes one day to implement, is in progress. Then the customer decides that feature F with tasks H, M, V, W and E should be done immediately. At least the following actions can be taken:

1. Leave task X into "In Progress" and move also some task of F to "In Progress".
2. Move X back to "To Do" and move some task of F to "In Progress"
3. Leave X into "In Progress" and make emergency swim lane for tasks of F. The emergency swim lane shows everyone that all the other tasks are on hold since the team is doing emergency work.

The author's experience suggests that, if the customer comes up with emergency work rarely, then options one and two can work nicely. If the customer puts forward emergency work all the time, then jumping between tasks messes up the workflow and waters down the whole point of work in progress limitations. In this case the Kanban table should show the process honestly as it is and some sort of emergency swim lane visualization should be used. If the Kanban table looks messy, then it is obvious that the process in use is messy and it should be improved on.

3.2 Retrospective

Retrospective is another agile practice for facing reality: How well the current process matches to the current and yet ever changing situation [8]? To retrospect is to stop, reflect last few weeks, planning what could be done better and then trying out few improvements for next few weeks. Plan of action needs to be made, or nothing will be changed. Plan of action consists of tasks. Task describes what should get done and who is going to do that. Wall of the team room is a good place to place the plan of action, so that it reminds people of what was committed. Retrospective is a tool for the team to evaluate and making appropriate adaptations in the following four areas [2]:

- Product value: Is value in the form of releasable product, being delivered?
- Product quality: Is the quality goal of building a reliable, adaptable product being met?
- Team performance: Is the team adapting effectively to changes imposed by management, customer or technology?

- Project status: Is the project progressing satisfactorily within acceptable constraints?

Retrospective is usually held between sprint iterations. If sprint iterations are longer than few weeks, then mid-sprint-iteration retrospectives should be arranged every other week. [7] Retrospectives can be used for all sorts of milestones, such as project ending [8].

Retrospective is one of the few practices that each agile team needs to implement in one way or other. Retrospective is all about to improvement. Threshold to try new ideas is smaller, since everyone knows that they are stuck with it for the following two weeks, no longer. If there is no retrospective culture, then changes may be thought to be eternal, which makes any changes to be risky.

3.3 Rhythm

Agile software development is commonly done in rhythms. There are rhythms on sprint iteration, daily stand-up meetings, interactions with customer on story detailing, releases, waves, constantly thinking, designing, building, testing and reflecting small increment of work [2]. Release, sprint iteration, and daily stand-up are time-boxed tools that make the backbone of rhythm. Figure 9 illustrates the relationship between these rhythms and the duration of rhythms and typical events that ends and starts new cycle of rhythm.

Daily stand up is commonly used agile technique. SCRUM version of daily stand ups are described next. Daily stand up is held preferably at the same time every day. Every participant is encouraged to answer three questions:

- What did you do yesterday?
- What are you planning to do today?
- What impediments are in the way?

Duration should not exceed 15 minutes. Purpose of daily stand up is to raise the visibility of each person's work and ensuring that their work is integrated. Impediments should not be solved in daily meeting. Impediments can be solved right after daily stand up. Daily stand up is for team members, not for status check. If status is checked team member feels pressure to conform to the plan rather than discuss coordination issues. Daily stand up meeting is a tool for self-organization. It is preferable participants to stand during the meeting to encourage the brevity. [2]

Sprint iteration is an important factor of rhythm. Return of investment is increased through continuous flow of value. End of sprint iteration is the beat in rhythm where business value gets delivered to customer. The length of sprint iteration is from one to four weeks. At begin of sprint iteration, customer has created prioritized list of features, estimated by the team. Then team is choosing top prioritized features, they can accomplish during next sprint iteration. Team is implementing features during the sprint iteration. Accomplished features and potentially releasable product is shown to all stake

holders at review at the end of the sprint iteration. Feedback is gathered during review. Retrospective takes place before new sprint iteration planning. [2]

Release means that product is published to real users. How often release is done, depends on when customer thinks that enough business value is created for new version. The most important features and capabilities are done in early releases. Release plan is giving overview of what will be released and when. Release plan is important for multiple reasons, such as for better understanding of project feasibility, mitigation of risk and give the team a "feel" for the entire project. [2]

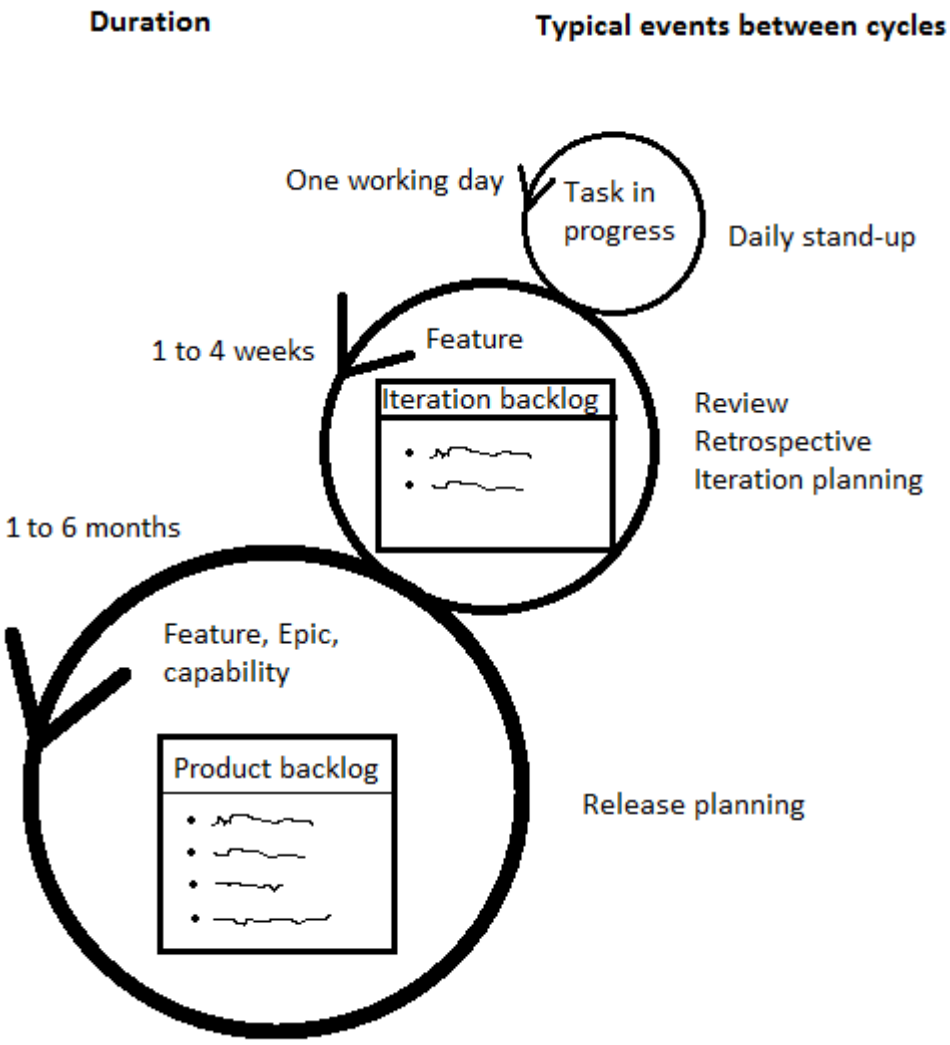

Figure 9. Agile rhythm. Release consists of several sprint iterations. Sprint iteration consists of several one day sprints. All cycles have constant length and different means of planning.

3.4 Estimation and planning

There are multiple levels in planning: day, sprint iteration, release, product, portfolio and strategy [10]. This chapter focuses mainly on sprint iteration and release planning. Estimation is kept as a one of the most difficult things programmers must do [8]. One reason is that programmers do rarely know how they will spend their time. This means to say, interruptions lengthen the time. Second, programmers do not know and cannot deal with all the specification details and techniques.

Story points are used for relative size estimations. Relative sizes are given to features instead of implementation estimation in time. Small feature is having relative size of one story point. Around three times bigger feature is having relative size of three story points. Twice as big feature as three is having either five or eight story points. All integers should not be used for estimating. With given uncertainties the differences between ten and eleven makes very little sense, whereas the difference between one and two makes all the sense. Eleven is only 10 percent greater than ten, but two is 100 percent greater than one. Fibonacci numbers, 1, 2, 3, 5, 8 and so on, could be usable sequence for estimating relative sizes [10]. The reason for story points is that it is easier to estimate relative sizes of features than it is to estimate time to build a feature [8]. For example, how long it takes to build a house? Or when you have already built a house, then how long it takes to build a house twice as big?

Velocity is how many story points are done in sprint iteration. Although estimates are almost never accurate, they are consistently inaccurate. One task takes more than was estimated and another one less than was estimated. Estimations are consistent in aggregate. Different set of interruptions occurs during different sprint iterations. Still in average interruptions tend to be consist from sprint iteration to sprint iteration. [8] Figure 10 illustrates this.

Velocity is used to make coarse grained releases plans – the average of story points delivered per sprint iteration. For sprint iteration planning some suggests that team commitment should be used instead of velocity. Team commitment planning means, that the team is choosing as many features, from prioritized list as they believe they can deliver during next sprint iteration. [10]

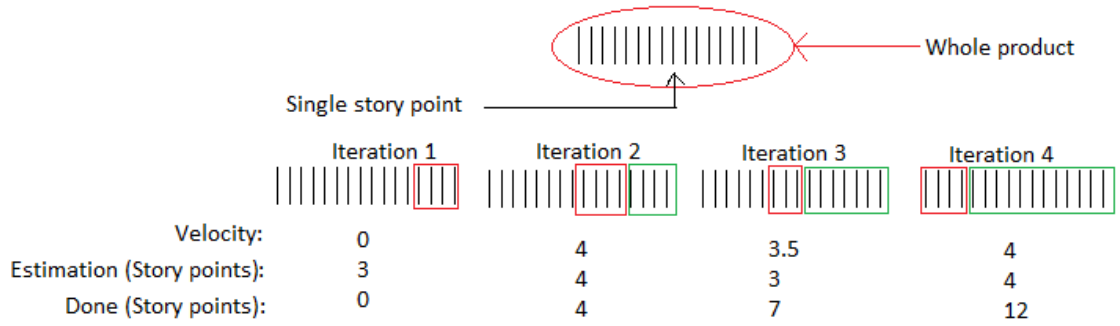

Figure 10: Estimating by velocity Deviation from sprint iteration to sprint iteration does not affect velocity much. One sprint iteration true velocity is bit higher and another sprint iteration bit lower than statistical velocity. [11]

3.5 Facilities in Agile

Facilities are a crucial part of a high performance agile team. Team work in the same room and the layout should support their work. They are working in an informative workplace, including information radiators and shared spaces, such as a whiteboards.

One example of an agile work place is a combination of caves and a commons. Caves are places for providing some privacy for phone calls, email writing and other need for separation. The commons are places where team members are intensively co-located.

Information radiators communicate information to passersby, so that they do not have to disrupt team members. Information radiators can also be aid for management to notice when the team is in need of help. [13] Some say information radiators help team to focus on the top priority issues. Two characteristics are keys to good information radiators: The information changes over time and it take very little energy to view the display. Information radiators should be placed into a densely used public space near the team. The team room is the best place for information radiators [8]. The entrance of the team room or hallway is also good. However, a web page is not, since visiting a web page requires more effort than most people is willing to expend. [7]. The information should be constantly visible from everywhere in team room [8]. Information radiators can be used to show for example work breakdown for the next sprint iteration, results of reflection workshops or user stories in development or in progress [7].

Information radiators may lead to gaming, in which people are not driven by business value, but by attempts to improve their position in information radiators. To prevent this, review the use of information radiators with the team and take old information radiators down after a sprint iteration or two. Above all, never use information radiators in performance evaluation or report them outside! [8]

Whereas information radiators are for the passerby, team members need tools to improve their communication in everyday work. Team members need to transfer knowledge and ideas from one to another. A whiteboard is an excellent tool for this. There should be plenty of whiteboards on the walls in the team room. [8]

3.6 Agile and Maintenance Work

The optimal situation would be that there is no maintenance work or emergency requests. The team is focused only on one goal. Rather often, the reality is different. This chapter discusses the question of how to deal with surprising or unscheduled maintenance work with agile.

What to do with emergency requests? Remember that the next planning is only a week or two away. Maybe the changes can wait until that. What if the changes cannot wait? Responsiveness to business needs is a core agile value. If sprint iterations are used, then as much work is taken out from sprint iteration as is added. In addition, only stories that are not started can be replaced. [8]

What if the working environment is too chaotic for sprint iterations? When a great share of tasks done in a sprint iteration are emergency tasks? When the difference between planned work and delivered work is significant iteration after iteration? Maybe a Kanban type approach could be used. The Kanban table is introduced in the chapter *Work Flow Visualization*. Priorities can be changed anytime, and whenever an old task is finished the highest priority new task will be selected.

3.7 Team Rules of Engagement

Every team should have rules of engagement. These are ground rules on how team members are supposed to treat each other. The rules of engagement should not have too many rules. Three may be too few and ten may be too much. These rules are owned by the team. The team participates in developing, adapting and enforcing these rules over time. These rules should direct conflict and contention in a positive way. [2] The SCARF model suggests that rules of engagement improve avoid-approach response by improving fairness [18].

There is no right set of rules. Each team must find what works best for them. However, to have an illustration of what these rules could be, look at table 1.

Team Rules of Engagement
Everyone has an equal voice
Attack issues, not people
Respect each other and your differences
Everyone participates into decision making.
Ask help when stuck
Offer help when asked

Table 1. Example set of rules of engagement.

3.8 Facilitation

Organizations have a lot of know-how, but they may be inefficient in using and combining all the know-how they have. Facilitation is a group process toolset for improving the way people participate and create results together in meetings. The facilitator does not take part in creating substance. Facilitation is meant to help especially experts and professionals in low hierarchical organizations. Figure 11 shows the relationship between facilitator, consultant and coach. Facilitation should answer these questions: How to start a meeting in a way that people feel safe and willing to participate? What facilitation practices to use to make different kind of people to participate and to reach the goal of a meeting? How to end in a way that people remember what was done and decided and they leave in good mood? [15]

All leaders should have facilitation skills. It would be good, if the team members also knew something about it. Facilitation is an extremely important tool. It is mentioned in this thesis to make the reader aware of its existence. Facilitation is a wide subject. The reader is advised to explore further sources to gain sufficient understanding of its usage.

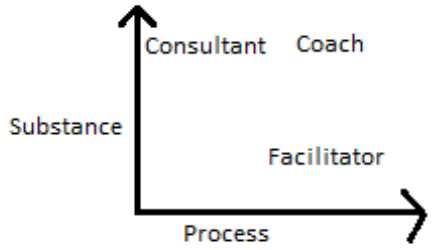

Figure 11. Relationship between a facilitator, a consultant and a coach is shown. The facilitator drives processes, but is not involved in substance. [15]

3.9 SCORE

The University of Maryland has developed a lightweight framework called SCORE for mentoring doctoral students. SCORE is based on agile. The core idea of the SCORE is to have two meetings: frequent daily stand up meetings participated by all students and on-demand research meetings. Daily stand up meetings are not actually held every day, but they are still called daily stand ups because of likeness with actual daily stand up. Daily stand ups are held in Maryland on late mornings on Tuesday, Wednesday and Friday. For more information about daily stand up, read the chapter *Rhythm*. On-demand meetings are arranged whenever there is a need for more in-depth, one-on-one meetings. Daily stand up is a good place to expose need for on-demand meeting. On-demand meetings are arranged typically on the same afternoon. Maryland University has been using SCORE since 2006. Following benefits are reported: More efficient time

use for faculty, improved productivity for students and improved group identity and shared knowledge. [14]

When asked by email if they have tried student teams, the author of SCORE paper answered as follows: "*We find that two-student projects actually work best. Or even larger projects in which there are sub-projects that students can mostly do on their own, but contribute to a larger whole. That way, they are always helping each other make progress, whether a little or a lot. However, sometimes one-student projects are all we can fund, or the students actually prefer this.*"

4 Thesis's Approach to Agile

Rest of the thesis is built upon the author's theory of how agile should be done. The rest of the thesis is not an official, well-accepted and theoretically sound approach into agile. The viewpoint to agile is defined first. Rest of the chapter describes challenges and opportunities to help succeed with given agile viewpoint. *Blub paradox* describes why it is so hard to make a human believe the benefit of any new approach, such as new viewpoint to agile. The *Rightshifting* model describes organizational evolution, which is extremely important, if continuous improvement is valued. Organization should rightshift, when they improve continuously. The chapter *Psychology of Change Resistance* concentrates on people's urge to maintain the status quo. It is not possible to improve anything without changing anything. Resistance is a natural reaction to changes. This must be taken seriously, if one wants to be successful with changes. The author's opinion is that **it is practically impossible for an individual employee to change an organization to be agile,** even if such attempts are not that uncommon. Failure risk is very high, even with managers. The chapter *Psychology of Change Resistance* is there to justify author's opinion. Chapter also gives some tools how to success, if one wants to try anyway. *Social Aspect of Learning and Innovation* defines why intensively collaborating small group is better to solve complex problems than individuals. *Competition within the Team* and *Measuring Performance* describes two common sources of dysfunction to collaboration and continuous improvement.

4.1 Thesis's Theory of Agile

Based on the author's personal experience and what the author has heard, most of agile transformations fail in becoming agile. Agile transformation failures are the main motivator to exploring new viewpoints on how to become agile.

Waterfall is a traditional project management methodology. Waterfall has many well-known structural drawbacks in complicated projects. These drawbacks are the same with all projects and teams. For example, a phase exists in which specification is created. This phase is at the beginning of the project and the client participates in this phase. Specification phase creates a document that lists all the things the client thinks he needs. After the specification phase begins the design phase. One or two software architects create the design of the up-coming software. The end product of the design phase is another document that is input to the following phase: programming. After programming, there is testing and after which the client performs acceptance testing. Accepted acceptance testing is the end of the project. One problem of waterfall is that the customer participates only at the specification and the acceptance test phases. It is very common that the program delivered does not meet customer's needs. Unmet needs

are a surprise revealed at the end of project, when everything was meant to be completed. How could one improve on this issue? If the structural problem is that the customer is too distant, then the solution is to involve customer more. This example means to claim that if waterfall has structural shortcomings, then continuous improvement and use of common sense would fix those problems. Moreover, different waterfall projects would take different paths in improving, but all projects would be fixing the same shortcomings with like-minded solutions. At the end, all projects would be similar. Not the same, but similar.

Let us recall how agile was established. There were several more or less independent teams who tried to improve their project management and teamwork in the era of waterfall. Then seventeen software professionals with a background of successfully managed projects met each other in the year 2001. Their goal was to find out what was in common with all their successful projects? Based on that, they created the agile manifesto and the twelve principles. [43] This thesis uses the theory that continuous improvement is what took teams and projects from waterfall to agile. On the other hand many traditional projects have tried to transfer to agile, by having certifications and education and installing some agile framework, such as Scrum. Often those attempts fail to be agile. Failed projects are probably not catastrophes. They may have improved their performance by using agile practices. However, they fail to have full potential of being agile. What traditional projects and teams need is an approach that changes traditional values to agile values.

Is continuous improvement enough? Distributed, unfocused teams are common in software development and even more common in the studied research projects. Some studied projects have been trying for continuous improvement. It has been very dysfunctional, since people were in simultaneously in several projects, belonging into many functional groups and project staffing was changing too often. The result was too fuzzy and complicated. If there are no teams and no isolated projects then what should be improved? Another similar issue is that many organizations and essentially all studied organizations optimize individual work. By doing this, organizations will find some local performance maximum. Learning teamwork skills will take time and during that time performance can drop. If done successfully, the performance of a team working intensively together can be significantly higher than in a case of individually optimized work. Third, if the team members are not equal, then improvements are only the opinions of a few. For example, project manager may consider that team members cannot be trusted. If the project manager is the ultimate decision maker, then improvement attempts will be biased and rest of the team may not be committed. Still, there are many evidences that team can be more than the sum of its parts. The chapter *Team's Effect on Learning and Innovation* offers more information.

Following theory is the cornerstone of the thesis:

To be agile, team needs to live out two values:

1. **Complicated problems are better solved by small and intensively collaborating group than by individuals.**
2. **Continuous improvement.**

Every advice and practice and theory written later is done with mindset aligned with this theory.

4.2 Blub Paradox

This chapter helps the reader to understand that people may not understand you when you are offering them solutions that requires a new way of thinking. Agile is a new way of thinking to most.

Theoretical average powerful programming language is called Blub. Now consider a programmer using this average language. As long as he is looking down to power continuum, he knows that he is looking down. Less powerful languages are missing some features he is used to. A C++ programmer knows that saving bits to register is less powerful than saving string to variable. When he is looking up the power continuum, he does not know he is looking up. All he sees is weird language, perhaps similar to Blub, but with some messy stuff thrown in as well. Blub is good enough for him, since he is thinking in Blub. Assembly programmer does not necessarily realize he is looking up the power continuum when he is seeing string saved to C++ variable. You need to have caution with the opinions of others, because of the Blub paradox. They will likely be happy with whatever language they happen to be using. [29]

The author of the thesis is a somewhat good programmer. He has rather good overall comprehension of a few programming languages. He also has a rather good understanding of the most common best practices. While communicating with people with different backgrounds, the author's experiences are aligned with the Blub paradox. It is apparent that people do not have intuitive skill to recognize the benefits of a new approach. Instead, people have an intuitive response to defend their habits or at best to see world through what they know by heart. What else can anyone expect? For example, try to explain the benefits and the meaning of functional programming to an object oriented programmer. Or try to explain the benefits and the meaning of object oriented programming to a functional programmer.

4.3 Rightshifting

Rightshifting is an organizational model. It illustrates the evolution of organizational mindset. Figure 12 illustrates rightshifting. When a new organization is born, it appears on the far left. Over time, with learning and improving, the organization shifts to the right. If the organization is continuously improving, it should be continuously shifting to the right. However, most organizations stop shifting after a while. [27] Continuous improvement is at the core of the thesis and rightshifting gives a tool to estimate progress. Rightshifting is also a power continuum. The more on the right the organization is in rightshifting, the higher it is in a power continuum. Read the chapter *Blub paradox* to understand the concept of a power continuum.

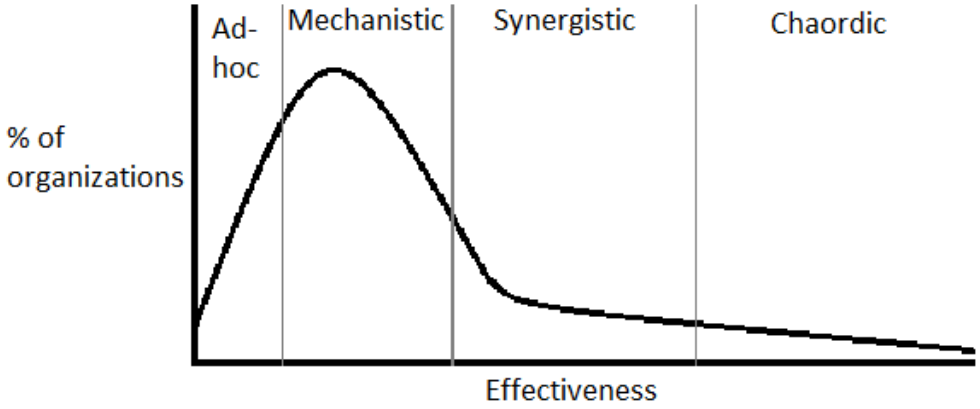

Figure 12. Four organizational states of rightsifting. The curve illustrates the percentage of all organizations at a particular state. [27]

The idea of rightshifting is to name four categories of organizational mindsets and how effective organization can be with a given mindset. [27]. Ad-hoc state is where many start-ups are working. There is no process in ad-hoc. One is working without thinking how it was done before, or how regular tasks should be done. When a start-up grows towards a corporation, more and more regulations and hierarchies start to appear. It enters into analytic, or Mechanistic, state. Command and control is descriptive to this state. Organization is seen as a machine. When the organizational mindset moves on, it enters the synergistic state. Mindset is transferring from machine to ecosystem. Synergistic state exemplifies the principles of lean movement. People are respected and organizations are seen as complex-adaptive-systems. Learning, flow of value and effectiveness are focused on. The last transformation is from synergistic to chaordic mindset. Chaordic mindset sees organization as a modern jet fighter that is aerodynamically too unstable to fly without on-board computers. Jet fighter is so unstable for the sake of top performance and agility. Bit less unstable and it is too slow to change is course in time. Bit more unstable and it would crash. Chaordic state means

finding a peak performance between orderly working and a chaotic collapse. Charodic mindset believes that being too organized, structured or ordered often means being too slow. Opportunities and threats must be dealt with fast. [27] Charodic state does not need as much coordination as lower levels, since the people are all aware of goals [28].

Anthropologist skills can be used to find out current states of organization. If people are talking about rules, regulations and controls then organization is on machine state. If growth, ecosystem and human relations and interactions are common topics then the state is synergistic. When the state is defined, it can be used when communicating with the people in organization. People must be approached at the same or lower level as where they are as *Blub paradox* chapter suggests. If organization is at the machine state then explaining something, such as agile, by human interactions and intrinsic motivations is not effective. [28]

4.4 Psychology of Change Resistance

SCARF Model discusses threats and rewards and how those interact with human behavior [18]. SCARF is used as a neuroscientific approach to explain and overcome change resistance. The rest of the chapter has a psychological approach to change resistance. While working on this thesis and talking with people, by far the most common reaction to new ideas was instant resistance. This is not only matter of public research, but a matter of being human being. Never underestimate the challenges you will face when changing habits. This chapter concentrates on change resistance and ways to ease it. Change resistance could be also described as willingness to preserve status quo.

4.4.1 SCARF Model

SCARF is a rather easy model that helps people to minimize threat-response and maximize reward-response in social interactions. Firstly, our social behavior is strongly affected by principles of minimizing threat and maximizing reward. Secondly, brain networks treat social needs and our primary survival needs much the same way. SCARF is a model to summarize these two. SCARF tries to give tools to minimize threats and maximize reward in any situations where people collaborate in groups. The SCARF model includes five domains of human social experience: status, certainty, autonomy, relatedness and fairness. [18]

Status is the relative importance to others. *Certainty* is being able to predict the future. *Autonomy* is to have control over events of one's own life, such as not being micromanaged at work. *Relatedness* is the feeling of cohesion and safety in group. *Fairness* is an experience of fair exchanges between people. [18]

Some events and circumstances are attractive to us and some are frightening to us. Attractive events and circumstances triggers approach –response and frightening events and circumstances triggers avoid –responses. Together these two responses are called approach-avoid-response. Studies show that approach-avoid-response is an involuntary

driver of human attention. This means to say that threats and rewards obtain our attention automatically. Add to this the fact that approach-avoid-responses have huge impact on cognitive performance and it follows that the avoid-response has a strong negative correlation. Even if the avoid-response is not an ideal situation, it is a default situation in many teams. On the other hand approach-response improves ability to overcome challenging tasks. Studies support the idea that the avoid-response generates far more arousal in the limbic system, more quickly and with longer lasting effects than an approach-response. [18]

Following are some examples. Threat to one's status activates similar brain network than a threat to one's life. When the work or a decision of respected researcher is questioned there is easily a perception of a status threat. On the other hand, increased autonomy activates the same reward circuits as receiving a monetary reward. To increase autonomy, the leader may consciously avoid micromanaging their employees. When boss or a workmate is making someone feel threatened then he is less likely to be able to solve complex problems and more likely to make mistakes [18].

Based on SCAFR, how would an old-school boss respond to agile transformation? Let us say he has 27 years of work experience. Throughout his career he has tried hard to climb the corporation ladder. He has a strong *status* that others must respect in order to work in the organization. He is the one making decisions. He has *certainty* in a way that he is the one who will be making the plans for the coming year. He has *autonomy*. After all, he is the boss. He does what he thinks he should do. No need to ask permission. He may not have much *relatedness*, since he is above others in an organization hierarchy. He has *fairness* at least to the extent that he can use his power to punish what he thinks is bad behavior or reward what he thinks is good behavior. Then the organization goes agile. What are the SCARF threats the boss is experiencing? Agile transformation threatens his *status*. He would not have the same hierarchical power to make decisions and to give orders. Perhaps he fears that his subordinates will not respect him anymore or think his is a loser, since power has been taken away from him. Agile is threating his *certainty*. He cannot make the plans himself anymore. Not only that, but new plans are less detailed and for shorter terms that they used to be. It gets even worse. Plans are less about control and more about motivation or less on rote and more on reasons. Being a servant leader threats his *autonomy*. He used to be a boss who did what he wanted and now he should serve others by motivating them and removing obstacles. Threat in *Relatedness* is not necessarily an issue. Is he experiencing *fairness* threat? Perhaps he thinks that after serving 27-years and earning to be where he is, it is unfear that he is losing his power. He is losing at least some of his power to unilaterally dispense justice. So he will probably need to live with increasing amount of experienced unfairness. Recall that avoid-response is much stronger than approach-response and social pain activates the same brain circuits than physical pain. The boss is in pain! Stress reaction, or avoid-response, decreases his cognitive skills which makes the situation even worse. The boss is probably fighting hard against the agile transformation and doing what he can to make it fail! Enforcing SCARF model's

approach-response may ease his pain. Still, transferring to agile is extremely hard to do successfully with old-school bosses. [44]

4.4.2 Fears

Fear is a natural way for humans to respond to change. The rule of thumb is that people feel afraid of losing more acutely than they feel the desire to win. This leads to situations where people prefer to fail conservatively over trying something new to succeed. [17] On the other hand, people are risk-averse when they are in risk of losing something and risk-accepting when they are losing something and may have a change to retain it [7; 17]. So when an organization tries to make a change, those who are afraid of losing are fighting harder than those who believe they are winning. Quite often settling the fears of those who are threatened leads to compromises that water down the original vision. [17]

Remorse has a stronger correlation with doing something that breaks routines and failing, than upholding the status quo and failing. The formation of the question can make a difference. Let us think of an example of organ donation. The rate of organ donation is higher when people must take an action to prohibit organ donation than when people must specifically permit organ donation. [17]

4.4.3 Tricks for Handling Fears

Reasonable risks need to be taken or nothing new will ever be created. So, what tricks are there to overcome the shortcomings of the human mind? The tricks in this chapter are not silver bullets nor best practices, but general rules.

Avoiding negative issues is far more important for a good human relationship than seeking out positive issues [17]. It is said that there should be at least three times more positive feedback than negative feedback. Seeking ways, such as rules of engagement, to decrease the amount of negative issues is a valuable tool for improving team work.

People should be guided by giving them positive feedback. Rewarding good behavior creates better results than just punishing bad behavior. Of course both have their strongholds, but for best results, positive feedback should be strongly favored [17]

Judging more based on approach than result can diminish the effect of hindsight. When decisions are judged it should be considered what was known when the decision was made. Even if results were bad the reasons for the decisions might have been justifiable and vice versa. [17].

Usually a wider frame works better. What is the chance that changing a single practice will succeed? If it is 0.55, then would you take that chance? How about if you have thousand practices to change and each of them has 0.55 probability of success? In a case of one practice you might win or lose. If you can make thousand profitable bets then in average you will most certainly win. [17]

The way you tell it makes the difference. What is the difference between these two sentences: “He has a 90% chance of surviving” and “He has a 10% chance of dying”.

They are telling the same thing, but surviving generates different kind of thoughts and feelings than dying. Changing discourse can change the viewpoint people are having [28]. Discourse was chosen between surviving and dying in the previous example. It can also be chosen between favoring individualism or teamwork. Even stronger feelings can be evoked by making the story more vivid. In general, the more vivid the image someone has in their head, the more important they will think the issue is. Think of these examples: "One child in a million will die because of this medicine" and "The probability of death caused by this medicine is 0.0001%". [17]

New knowledge of lower risks improves human beliefs of benefits, even if nothing was said about those benefits. New knowledge of benefits also makes risks feel smaller, even if nothing was said about risks. [17] It follows, that if people are not sure about the benefits of change, one trick to convince them is by talking about how small the risks are.

In a case where more risk taking is needed one may want to try one of these two tricks: First, phrase the proposition to sound like you are facing high probability to lose and that is why new working practices need to be tested. Second, if probability of failure is high, then make it sound like a lottery [17]. It is about trying out and maybe winning, more than it is playing risky and failing.

4.4.4 What People Believe and What Changes Their Mind

Risks are only one important part of change resistance. Some other aspects of how humans come to believe what they believe are descripted in this chapter.

Human have only a certain amount of self-discipline to use. When a person shows self-discipline in one matter, he is less likely to resist some other temptation. For example, say person needs self-discipline for two actions in his daily life: to work hard and to exercise. If he starts to work harder, he probably starts to exercise less. This phenomenon is significant in many businesses. When people are busy, they will probably not have enough self-discipline to simultaneously change their habits. [17] Moreover, when people are stressed, they regress to their old habits. In these circumstances it is extremely hard for people to change their habits [7].

Human beings do not suspect their opinions too often. At least it is very rarely spontaneous. However, in order to believe something a person needs first to think the claim is true and only then he can try to explore how the claim could be true. He would try to proof that the claim is true in his mind. [17] To help this happen, one could try to play co-operative games, such as "Let's try to think what would need to happen to make this true!"

Ones feelings have extremely highly correlated with his beliefs. When people believe a conclusion to be true, they probably believe the evidence as well. To make the effect stronger a person is always predisposed to evidence that supports their current beliefs. A good and believable idea is not the one with the most facts and a rich set of viewpoints, but the one with the most consistent and vivid story. Knowing too much can even make

it harder to have self-confidence, since the story is more complex than it is consistent. When there are gaps in facts, human subconscious fills these gaps without conscious thought. So it is natural for a person to feel like he knows everything that is essential. All this leads to a situation where a person feels too self-confident with his intuitions. The claim that he likes has no cost and the claim that he opposes has no benefits. He may adamantly believe in absurd claims, particularly if he is surrounded by like-minded people. [17]

The human brain always searches for signs of causality. Of event X being the reason for effect Y. Often the world is too chaotic for causality thinking to be accurate. It is rather rare that human beings recognize this. Luck does not fit in with causality. If a person P creates a web service and it fails, it is easy to see all kinds of reasons for that. Maybe he did not care for his clients well enough, or the front page was not cool enough, or the timing was bad or he should have used cloud services and so on. Most of startups fail. So how likely it is that this person P fails because he had normal luck, instead of good luck? Why did Google succeed? Were they mostly talented or lucky? Luck does not fit with causality. People are inclined to see causality even when circumstances are all about luck. This effect, as well as self-confidence, are enormous sources of hindsight. Judging by hindsight is less harmful when someone works aligned with well-established approaches. So hindsight diminishes risk taking and favors bureaucratic approaches. On the other hand people tend to be optimistic. They have a vivid image of the best case scenario, but not a comprehensive understanding of how things can go wrong. Optimism keeps people trying. Optimism compensates for fears. Research contains lots of adversity and rare success, so it is vital for researchers to be optimistic. [17]

Sometimes educating people on baseline probabilities may help people to see over their consistent story. For instance, if someone was in the process of establishing their own startup: They can be asked how likely they are to succeed compared to other startups, given all that is known about their business idea. The answer may be three times more likely. They can then be informed that in reality around 90% of startups fail. You may agree that 10% success rate is a baseline and the estimation is extra information on that. The Bayesian probability can be calculated and the probability of success would be $0.1 * 3 / (0.1*3 + 0.9*1) = 0.25$, which is 25%. People are bad with probabilities and baselines. [17] Educating tries to overcome that.

Subconscious replaces difficult questions with easier ones. It enables quick answers to complicated questions without much knowledge. [17]. For example, if a Finn is asked how Russia will behave next year, he tends to replace this question with how Russia is behaving now. Crimea was taken over by Russia around the time that this was written, how did this event change Finns answer to “How will Russia behave next year?” What does this matter? Let us think that an organization is trying teamwork, if its employees agree that it is a good idea. After all, teamwork cannot work, if employees are against it. Employees may feel like they have all the information they need and no education or conversation is needed. They already have an opinion. Then instead of answering “How

beneficial would teamwork be?" they easily answer to "How much do I enjoy working intensively with others?" Moreover, replacing questions about future with thoughts about how things are in the present has a big impact. It is hard to convince a person that changes need to be taken now, even if there is no vivid pain yet. Think about all sort of possible environment catastrophes for instance.

Human beings do not change their behavior based on general information. Not even if they understand and accept the implications of that general information. People tend to be bad at statistics. General information is statistical in nature. What helps a person to change their mind is a vivid example. Our own firsthand experience is even better.

Humans tend to believe and like things that are cognitively easy. Known things are easy. The idea of using Windows is easier to most than the idea of using Linux. Cognitively, it is hard to separate easiness from truthfulness. Factors that make an issue cognitively effortless are repetition, clear appearance of issue, focused thoughts and good mood. [17] For example, when there is resistance to using agile, a simple one afternoon demo may change people's estimations of its cognitive easiness.

Ownership increases experienced value [17]. This is no news to agilists, since agile cannot work without people possessing ownership over their work. An effective, but difficult trick to make people accept change is to make them owners of that change. Ownership increases the value, if the owned concept is thought to be using, not trading [17]. For instance, changing five one Euro coins to one five Euro bill is no problem. However, changing one's favorite coffee cup to a new one can pose a problem. Here money is an instrument of trade where the coffee cup is for use. In agile, people should have ownership over continuous improvement and business value. They should change working habits, processes and practices whenever needed. One thing that should not be owned is ideas. Owning ideas is very detrimental in teamwork. When ideas are owned, they are not shared. They are not built on top of each other. They are not played around with. They are not part of a co-operative game, but forts to defend. Ideas should be traded! In that way they are part of a co-operative game in which innovation and learning is boosted.

When change is needed, the vision of success should feel as good as possible. To make it feel good, the story must be made vivid, consistent and simple enough. Including probabilities makes stories less vivid, less consistent and less simple. Moreover, people tend to believe simple language more than complicated language. Some people like to use sophisticated language, but it may take its toll. Rhymes can be powerful on summaries to make people buy your idea. [17] Consider these sentences: "When life gives you a hundred reasons to be sad, show life you have a thousand reasons to be happy" and "When life gives you a hundred reasons to cry, show life you have a thousand reasons to smile".

The overall impression makes the difference to observer's feelings and thoughts. This is called the halo effect. Politicians with a certain type of face get votes easier than others. [17] It is limited what one can do to the size of one's chin, but it is possible to learn to control emotions to make one look positive instead of aggressive. A person can

also dress appropriately, learn to use gestures, control one's voice and tell vivid stories. A teamwork related question to think about is how much the research world is affected by the halo effect of geniuses? Well known geniuses, like Einstein, Edison and Copernicus are not known as team players, but as independent workers.

Finally, if you want to play dirty, trivial details can make people change their minds [17]. This practice can easily take its toll on you. It is widely used in software development and even more commonly in politics. Some use it unintentionally and some intentionally. Tricking researchers with details may be more difficult than tricking most people, since researchers are far more accustomed to questioning claims.

Collaborative engineering recognizes that an individuals' decision perspective is dynamic and affected by others. [7] When humans adopt a new viewpoint, they instantly lose their ability to recall most of the beliefs he had had before that. When defined well, people do change their beliefs even if they do not acknowledge or recognize it. [17]

4.4.5 Change the System and People Will Follow

People tend to adjust their habits to align with their company's way of working. A study suggests that system causes 95 percent of all changes in habits. The conclusion from that is that changing the system will cause most people to follow. [30] Cockburn suggests that people are remarkably capable of acting differently given new motives and new information [7]. He also writes that the initial reaction of most people is to force one group's values on other groups as well [7].To combine these three: Changing the system means to give new motives and new information. When doing so, a remarkable portion of people start to change, consequently forcing the rest to change as well.

Even if it is accepted that the system causes 95 percent of all changes in habit, then who controls how the system varies? The easiest answer would be the leaders. They do have power, but not as much as we think or financial magazines and business books, such as Build to Last, suggest [17]. Competitors, global economy, trends etc. are also changing the system. All in all, leaders are responsible for changing the system, even if results are not exclusively in their hands.

4.5 Social Aspect of Productivity

How does learning and innovation change between individual work and work done in small groups? This is an important question, since public research is largely individual work. Small groups can solve complicated problems better than individuals. For example Fabrizio Butera has done much research related this issue [22].

4.5.1 The Background of Public Researchers

From the beginning of school life, children are required to demonstrate their learning through tests done individually. To pass those tests, they practice by doing exercises individually. Focus is not on helping each other, but on make oneself look good. And one needs to be impressive, when growing older and applying for academics, such as secondary education or university. Even being successful is not enough. One needs to be better than a certain amount of your competitors, or fellow students.

At the university, an individual is still mainly responsible for their own results, rather than being a team member. You do your own exercises, your own exams and you study alone for exams. Even when studying for an exam can be done in teams, it is not supported or encouraged. The result is that people do not co-operate much to improve learning. There is some group work at university as well, which is good, but not sufficient. At this point in their academic career, people tend to not have good team skills. Group works are full of arguments, compromise and generally split for each participant to do their part apart from each other. When the pieces are put together, more arguing and compromising happens. Finally, intellectual demonstrations, such as a master thesis or a doctoral thesis, are usually done alone. Another aspect of doing it alone is that copying is seen as bad. You should be as original as possible - doing it all by yourself. This is the case even if you can save time and get better results by utilizing someone else's previous work. What is described here is the viewpoint of the author, who entered elementary school in Finland in 1990. Similar experiences are had in other cultures as well [7]. Some suggest that universities should organize more communication-intensive courses [7]. Group work is good practice, communication-intensive sounds even better. Perhaps educational institutions should also have entrepreneurial attitude education? For example, not teaching and examine facts, but giving an open problem to a group and requiring well-reasoned solutions.

Some of the university students stay at the university after they graduate. They do public research and teach new students. Being a researcher is a sort of natural continuum from studying. As a summation, there is a reason to believe that people in public research do not fully understand the potential of working in small groups. Read chapter *Blub Paradox*, for further reasoning. Advancing agile methods seems to be driven by industry practitioners, not by academic researchers [42]. This is one factor that supports the idea that academic culture is not agile at least on purpose.

4.5.2 Team's Effect on Learning and Innovation

Let us start with the big picture. Communication patterns are usually the most important factor in both productivity and creative output. Communication patterns are more important than education or class structure. Income per person grows exponentially as more people share ideas. So it is sharing ideas, not just contributing more that boost performance! [38] Sharing more ideas requires better communication!

Innovation and learning can have multiple meanings and multiple purposes. They are discussed here as tools for overcoming challenges and improving results. The challenge can be anything, such as a law of physic, domain knowledge, programming paradigm or a software development project. In a way, the target of innovation and learning is better problem solving.

Interaction drives innovation. Innovations emerge from the interaction of diverse individuals. [2] Teams are used and praised in software development. In a complex environment one person cannot have all the useful know-how himself. Different team members have slightly different skillsets. Moreover, team members putting slightly different viewpoints on the table, can help to better overcome the challenges in hand. For example, some suggest that teams with fewer than four programmers are less likely to have all the intellectual diversity they need [8]. There is also evidence that programming in pairs increases productivity [37]. Free-form socializing has been found to be more effective way of learning than documents even in less complex or abstract industries [16]. Natural human interactions seem to be natural way of learning and tackling complicated challenges for humans. This should not be a surprise for anyone. Additionally, accomplishing together rewards intrinsically motivated people [7]. When working well, small groups can increase motivation to face and overcome challenges. Being successful in overcoming challenges and in producing results builds a more coherent team [2]. A team that is more coherent motivates people even more, in turn helping them to build better results and so on.

The cone of experience, also known as the learning pyramid, is a well-known visual metaphor for placing learning activities in broad categories based on the extend they convey the concrete referents of real-life experiences [19]. A simpler and less obscure description is that the cone of experience illustrates how much a learner can recall, when different learning practices are used. The idea of the cone of experience is not to advocate one media and oppose another [19]. A wide-ranging use of different practices results in the best outcome [19]. The cone of experience is illustrated in figure 13. There is criticism concerning the cone of experience as well. Mainly about how accurate the percentages are, how many variables there are when measuring learning and how learning can be defined and measured by multiple different ways and about different purposes of learning. [19; 20] Still, the core idea of the cone of experience is helpful.

What is obvious from figure 13 is that active practices are more efficient in learning than passive ones. What is not so obvious is that generally more efficient learning practices involve more active interactions with others. When a team is working well, team members are using more or less all of the learning practices listed in figure 13, with the emphasis on active practices. For instance when a team member is facing a problem he would describe the problem to others. Surprisingly, often one is capable of figuring out the solution, only by explaining the situation to others.

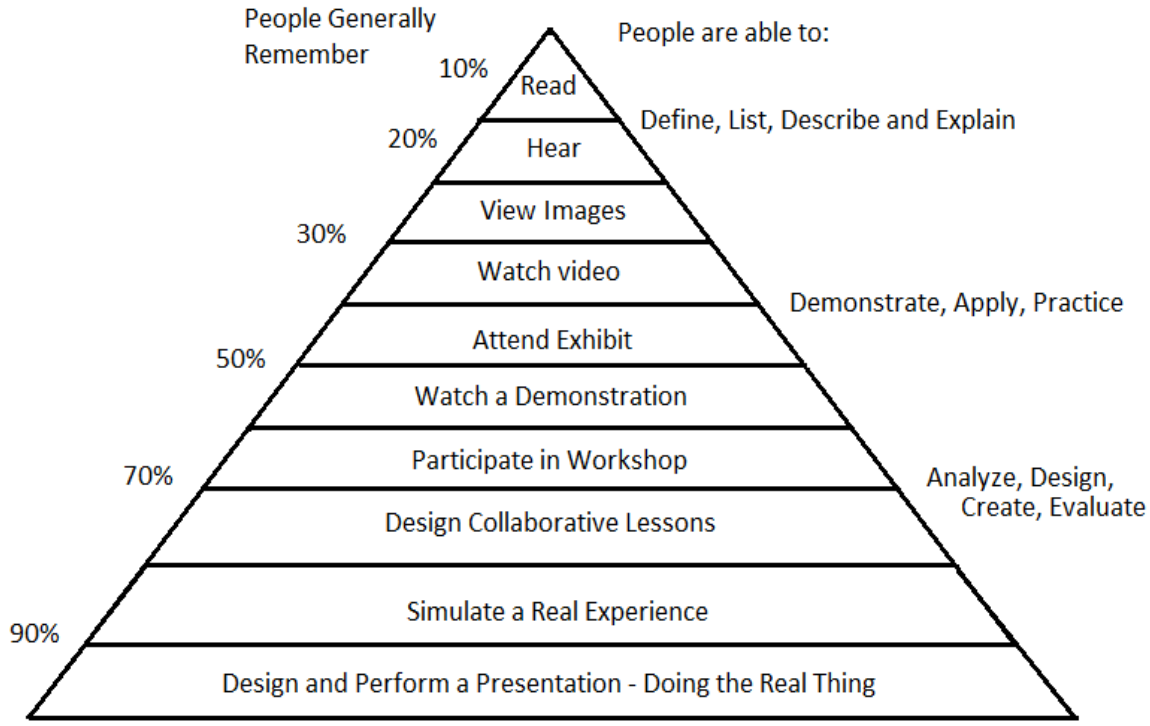

Figure 13. The cone of Experience demonstrates the strength of experience in different types of learning. The stronger the experience is, the easier it is to remember. [21]

4.6 Competition within the Team

As described earlier, there is a reason to believe that the educational system prepares people to work solo. As a side effect, it may encourage individuals compete with each other. What kind of effect does competition between team members have on the performance of the team?

There are many reasons to believe that competition decreases performance. For example using the SCARF model, it could be estimated that competition can increase status, relatedness and in many circumstances fairness threats. Even the definition states that "*relatedness is a sense of safety with others, of friend rather than foe*" [18]. If competition is causing a threat, then it is likely to also cause more mistakes and to reduce cognitive performance [18]. Effects of competition in mutual interactions have been studied a fair amount [23; 24; 25]. These studies confirm that mutual competition does decrease results. A minor threat such as talking with an expert may cause status threat and decrease a participant's ability to think creatively [25]. Studies show that possession of identical information is detrimental to participant-participant co-operation [23]. Identical information is fruitful ground for competition. Results are better when a positive resource interdependence is present. In that situation participants need each other and co-operation works better. There is so much relevant information in a complex work environment, such as software development, that people are practically always having a positive resource interdependency.

Not all think that competition within the team is purely detrimental. Cockburn suggests that competition can also be used to create better results. The catch is to create rules so that the competition framework fosters collaboration. For example a team member may gain points from reviewing code for someone else. [7] Detailed description of a framework can be found in Cocburn's book. The framework does not focus on mutual competition. It focuses on gaining points. So even though it is called a competition framework, it does not focus on mutual competition within the team. This is an important distinction. If mutual competition occurs, then gaming would harm collaboration. For example, one may not offer one's code for review, since the reviewer could gain points. Caution is advised before using a competitive framework. If a relative amount of the points gained are used, then there would probably be mutual competition.

4.7 Measuring performance

Performance measurements are widely used. Public research is no exception. Measurements used in public research are the number of master theses, doctoral theses and publications as well as the reputation of the papers in which the publications were published. There may be other performance measurements as well. The reason and inevitable consequence of performance measurement is that it impacts the work and the results.

The rule of thumb is that agile teams should be measured more by how they have improved than by their performance. High performance is not demanded, it is expected. High performance is not as much achieving a certain state as it is a journey toward something better. [31] This is an eternal loop between improving and high performance.

Studies show that academic researchers are disappointed in the current performance metrics in Finland. Their experience is that the performance metrics are misguided and lean toward impolitic results, such as partial optimization, result gaming and increased bureaucracy. Performance measurements are not encouraging, conversely building up pressure to publish. Stress to publish increases the amount studies that are valueless but can be published fast. [32] Another study claims that the race for obtaining funding does not improve the quality of research in Finland's universities [35].

Metrics often improve results initially. Those who are being measured learn to work the metrics at some point. Then, with pressure to improve they are forced to subvert the intentions to meet the measurement goal. Metrics are always disconnected from the desired outcome. Over time measured performance keeps going up and true performance declines. [2] Especially measurement obsession should change from time to outcome [2].

Academic metrics tend to favor individual work as discussed in the chapter *The Background of Public Researchers*. Agile teams should be responsible for the outcome as a team. Metrics should be aligned with that. As stated in the chapter *Agile Manifesto and Principles*, Amatriain claims that public research should demonstrate more

commitment and response to social needs than obtaining grants, patents, or publications. If this viewpoint is accepted then performance measurement should be aligned with it.

To summarize the chapter on measurement here are two quotations. First by Jim Highsmith: "*Delegatory agile system measurement should therefore be focused on two things: determining the value of output delivered to the customer and providing staff informational measurements with which they can do self-assessments to improve their own performance*" [2]. Second quotation by Rob Austin: "*Trust, honesty and good intentions are more efficient in many social contexts than verification guide and self-interest*" [2].

5 Thesis's Theory of Agile in Action

There are many practices and frameworks in agile. They are general rules that have been found practical by multiple teams in multiple projects. However, practices should not be settled on blindly. The bottom line is that only two things are required for agile to work: real teams and iterations. If these two are done well, then useful methodology will be found. Some say using general rules practices and frameworks will make immature agile teams progress faster and safer [8]. Real teams and essence of iterations are described in this chapter.

5.1 Iterative

Iteration is a central word in agile. For sure it means different things to different people and in different frameworks. Iterations are linked to product adaptability, improved time-to-market, people and process adaptability, reliable results and so on [2]. Iterations give a phase to development, retrospectives, reviews, planning and all other things teams are doing periodically [8]. Sprint iterations are one form of iterations and they can be of different lengths. When defined as agile methodologies do, the sprint iterations take from one week to several weeks. All other periodical practices, such as daily standup meetings, are also iterative. Iterations should have a constant length.

What makes iterations so important? First, iterations improve all sort of learning. Project stakeholders need to learn about updated customer values, technical skills, social skills, teamwork, market needs, practices, constraints, vision, how to improve performance, what other team members are doing, what are potentially shippable features and so on. Iterations are a way to get feedback. The faster the feedback, the more it helps learning. When hand is put on hot stove, you learn immediately, not to put hand there, since you get almost instant feedback. Let us imagine a scenario where the burning sensation would take six months to reach the brain. Now someone puts hand on a hot stove. After six months he feels a horrible pain. Did he learn not to place a hand on a hot stove? He did not. Maybe he was watching his favorite TV-show when the pain hit. His conclusion might be that the TV-show caused the pain and he stops watching it. The example may feel strange. First, it has nothing to do with software and second it is counter intuitive and not true for burning sensation to take six months to reach the brain. The point is that fast feedback is essential for efficient learning. Not placing a hand on a hot stove is learned fast and well. Learning to develop software is not learned fast and in many cases it will not be learned well. What would happen to the rate of learning, if feedback was as fast as with the hot stove? Basically the shorter the iteration is the more it improves learning. However, iterations that are too short are inefficient as well since real value needs to be achieved during the iterative period and that takes time. [2] Second, iterations also force tough decisions [2]. The question of what should be done

next is answered after each iteration. Table 2 shows two of the most common agile iterations and some basic decisions linked with those.

Name of the iteration	Typical decisions
Daily standup	• What tasks will individual do before tomorrow's daily standup
Sprint iteration	• Project can be canceled, if it does not look so feasible anymore. • Working processes are improved. • What to build during next few weeks is planned. • To release or not to release

Table 2. Two of the most commonly used agile iterations and decisions linked with particular iterations.

5.2 Real Team

Real team is a complex term in agile. Attributes like self-directed, self-organized, technically excellent, co-located, collaborative, cross-functional, right sized, business value driven, client involved, transparent and focused can be used to describe agile teams. It is said that self-organizing has no absolute value. If a team is forced to self-organize, then it will take longer for them to be self-organized. Then what could be done? Just engage with your team in purposeful dialogue and mutual learning about how the work should be done. [41] The same applies to all the other real team attributes. The point is not to implement real team attributes, but to improve continuously. The meanings of these attributes is discussed next.

5.2.1 Self-organization

Self-organization means that the team is empowered to organize its own everyday work. A crucial part of organizing one's own work, is that the team makes their own workload estimations. The team is told *what* to build, but the team decides *how* to build it. The team is therefore not micromanaged by project manager or anyone else.

There are multiple reasons for self-organization. First, it rewards and motivates people to have autonomy over their work and results [18; 39]. This helps them to be more self-disciplined and to take the initiative. Second, in a complex environment there are too many variables and too much information for any manager to micromanage successfully. Team members know best how their work should be done. It is natural to let them make those decisions for themselves. Third, self-organization, with vision and self-discipline, are the tools that help the team to adapt to changes so that creating business value is preserved.

5.2.2 Self-discipline

Self-discipline is one's ability to take action regardless of one's emotional state [9]. For example, one does not push messy code to the revision control system even when busy. Here the emotion is "I am in a hurry. No time to refactor that now. The customer will not notice, if the code is messy, but he will notice, if the feature is not delivered. Besides, when this messy code kicks back it will be someone else's problem". One knows all the drawbacks messy code has, so the programmer shows self-discipline and spends enough time on refactoring. This was an easy example to understand and to agree on for programmers, even though it is not so easy to obey. To work in an agile environment one must also demonstrate self-discipline by confronting reality through rigorous thinking, accepting accountability for one's own work, avoiding victim mentality, adapting actor mentality and showing interest in developing skills toward technical excellence [2]

Self-discipline is far more difficult in human interactions. Yet there are many team related issues in agile that require self-discipline: responding to criticism in a constructive way, respecting one's colleagues, being willing to work in a self-organized environment, taking the initiative to confront others when they are not performing or behaving according to team rules, directing a conversation toward getting all the relevant information out on the table without attacking anyone personally.

Even though human interactions are crucial for successful projects, they are not sufficient alone, technical knowledge is also required. Technical excellence creates new opportunities, produces higher quality products that can respond to changing business needs, shortens time to market and makes estimates more accurate [2]. Developing technical excellence requires self-discipline.

5.2.3 Co-located

Co-located means that the team is situated in the same room. They see each other and they have information radiators and low-tech, high-touch tools such as whiteboards. The reason for co-location is to make the teams communication as good as possible [7]. Communication temperature means how much informal emotion rich communication is used. Hot communication favors physical proximity, three-dimensionality, smell, kinesthetics, touch, sound, sight, cross-modality timing and low latency. Figure 14 illustrates Effectiveness of different modes of communications. Teams, often distributed, that use cold communication resort to emails, instant messages, wikis and all sorts of documents. Warm communication teams resort to face to face communications at the whiteboard.

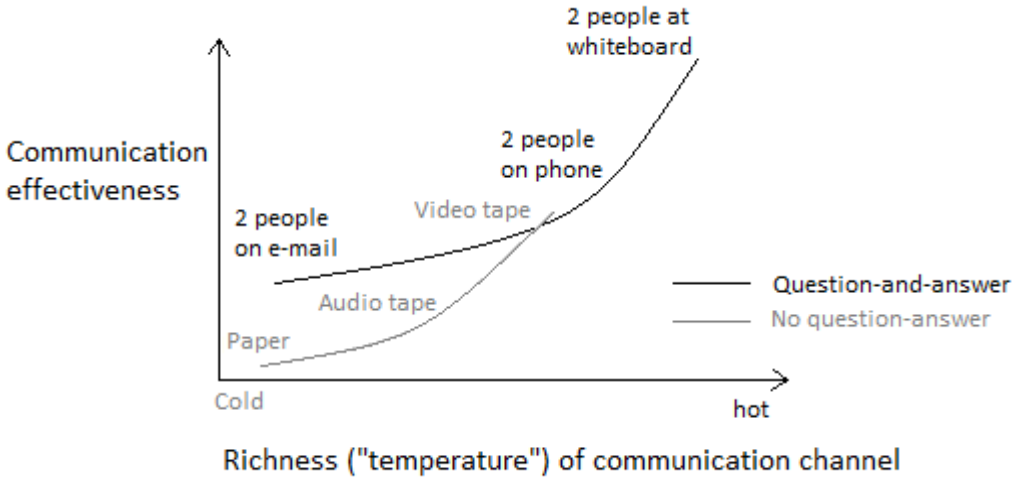

Figure 14. The effectiveness of different modes of communication. The temperature describes the level of informal, emotion rich communication. [7]

A co-located environment nourishes a low cost of information transferring and a low cost of lost opportunities [7]. Information transferring cost is a result of two aspects: First, how long it takes one team member to discover that another team member knows something useful. Second, how much energy it takes for these two team members to get together and transfer the knowledge from one to the other. Lost opportunities come from making poor decisions, because of not asking the question or having conversations with team members. [7] For example, a team member might think that asking a complicated question via email is too difficult and end up making a decision based only on assumption.

Osmotic communication is a means for low cost information transfer. Osmotic communication means that the team members are working so close to each other that they are picking up traces of the ongoing conversations even though they are not consciously paying attention [7]. Osmotic communication enables intensive team work.

A “us vs. them” attitude is another source of dysfunction with distributed teams. It means that each group forms its own community. There is a real risk of confrontation attitude evolvement between groups. People have a genetic instinct to having different

attitudes toward an inner group and an outer group [47]. For instance, you may hear someone to say: “The project would have met its goal, if only they had done their part”. A team does not have to be distributed to different countries, time zones or cultures for an “us vs. them” attitude to emerge. Just working on different floors of a building can do the trick. [7]

5.2.4 Collaboration

Collaboration involves two or more people jointly producing a result. Highsmith has written an excellent summarization of the key ingredients of collaboration: “*The quality of results from any collaboration effort are driven by trust and respect, free flow of information, debate and active participation – bound together by a participatory decision-making process.*”. Participatory decision-making is the heart and soul of collaboration. Let us imagine there is collaboration within a group. People are driven by trust and respect. They share all the essential information they have. While they are intensively taking part in debates, they are still telling their truth rather with compassion than with constructive criticism. Then at the end, someone makes a unilateral decision that is put into practice. Would people feel like they should participate next time? There is no healthy collaboration without participatory decision making!

Collaboration is important for many reasons. It enforces interactions of diverse individuals, which drives innovations. Innovations, ideas, are not built and owned by individuals, but by team. This in general makes innovations better and team members more committed to try innovations in practice. Collaborative innovation is more about reconceiving than compromising. Collaboration also improves learning from others and helps in creating a shared space and shared experiences between team members. [2] Collaboration should also be fun. Over-seriousness is a warning sign of mediocre bureaucratic thinking [31].

5.2.5 Cross-functional

Being cross-functional means that the team has all the roles, knowledge and people it needs to produce business value. Business value delivery is the reason for cross-functionality. In software development this usually means that the team is consisted of developers, testers, an agile lead and a client representative. A cross-functional team creates business value or the team is lost. If the client needs a tool for writing blogs, then there can be real business value in a feature for editing an old blog, but there is no real business value in an implemented service interface for HTML GET, POST and DELETE requests.

One great aspect of creating business value in cross-functional teams is that the team members all share the same goal. When functional teams create business value together, they are practically always having at least some of these: functional managers with functional agendas, functional reward system, functionally optimized goals and stronger

functional identity than team identity. When it is necessarily for people to work jointly together, they need to have a shared goal [2].

Collaboration increases knowledge transfer and learning across the team. When the team is also cross-functional, it makes the team members learn cross-functional skills. Specialists tend to become generalists in agile teams. When team members are generalists they can help each other and they are less dependent on a single individual in a case where he is a bottleneck or leaving the team. One might argue that you need specialists for the most difficult tasks. It may even be true in some circumstances. It is particularly true with new agile teams. However, teamwork tends to make learning faster. Even as generalists, they are still good with problems that need deep understanding. On the other hand, people can solve difficult problems together in a cross-functional team. There are evidences that small groups solve complex problems better than individuals [2; 8; 19].

5.2.6 Right sized

Agile teams are preferred small [2]. There is no rigid maximum number of members, but some suggest that no more than twelve people should be in the same team [7]. Others say that team size can go up to twenty, as long as it has no more than ten programmers [8]. If the team becomes too big, then it needs to implement special practices that are outside the scope of this thesis [7; 8; 2]. Same variance of opinions goes with lower number and optimal number as well. Some can argue that team size of four could be considered optimal [7]. Others say that they would not use XP, an agile framework, with less than five people [8]. Numerous anthropological studies show that group size from six to eight is ideal for peaceful collaboration in all kind of environments [26]. Some others claim that natural family size is a good size for a team. Evolution has made humans tribesmen, who are eager to form fairly closed small groups – tribes [40]. It is important to notice that agile teams require a relatively high portion of seasoned members to be efficient. It is said that the ratio of experienced to beginners should not be lower than 1:5 [7].

One person projects are problematic. There is no innovation or learning help from a team. Some suggest that several too small projects can be assigned to one team. The team then works these projects one at a time. [8]

5.2.7 Focused for Business Value

Agile teams are business value driven. For this to be true, business value should be the primary measure of success. Business value driven means several things. First, the feature with the highest business value is done first. Second, new features are delivered to the customer in a form of a releasable product in a time-boxed fashion. Third, the project processes within acceptable constraints. Fourth, improved business value is a valid reason for change in plans. Fifth, the quality must be high enough to make the product reliable and adaptable. Sixth, adapting the lean principle to minimize the

amount of work in progress and shortening throughput time is also a part of being business value driven. This leads to time to market optimization instead of resource optimization. [2]

All agile team members are focused on one and the same project. This helps them to pull project through as fast as possible, which means fewer intermediate products. Intermediate products are a waste in lean perspective, so focusing reduces waste. It also helps them to be a better team, since they all have the same and only the same target – to deliver the current project.

5.2.8 Customer involvement

The customer is active in agile software development. The customer is usually the sponsor of the project but also the vision owner. Therefore the customer must accept accountability for identifying, defining, prioritizing and accepting features [2]. Rest of the team can and should contribute suggestions and ideas, but the customer has the ultimate responsibility [8]. The customer must also be available for answering the questions the team has on a daily basis. Lack of customer involvement will lead to failure. The more the customer and the development team consider themselves as a single team, the more successful the project will be [8; 2]. Some even suggest as a rule of thumb to include one product owner plus two on-site customers for every three programmers [8].

The customer may or may not be the real user [8]. Typically the product owner, the domain experts, the integration designers and the business analysts play the role of on-site customer [8]. Even if the on-site customer does not necessarily have to be a real customer, the product owner has to be a real customer with a real product vision. People are more important than roles, but one needs to be extremely aware and enlightened before mixing customer roles [2].

5.2.9 Transparent

Agile teams are transparent. All stakeholders have all the information they need and only the information they need to know about project progress. Moreover, the information should be in a form that makes it easily perceivable. There are different stakeholders in a project, who all have different needs. Only transparency needed by the team and the customer are concerned here. Team members need to track what others are doing in a team for synchronization and for getting and giving help. A typical tool for this is a daily standup meeting described earlier. Team members usually want to know how the project is doing in comparison to the plans that the team has made. Information radiators are typical tools for this. For example, one glance of a burn down chart can tell if the sprint iteration is going as it was planned.

Customer has a good knowledge of what is done and what is in progress, since the customer is a part of the team. They are seeing what is getting done. Customer knows what will be done next, or in the current sprint iteration, since the customer has

prioritized features and can see the prioritizations and what has been selected into the current sprint iteration backlog. Finally, all customers and other stakeholders can attend the sprint iteration preview, where team represents what they have accomplished during the last sprint iteration. Questions and feedback is asked in these meetings.

6 Research practices and materials

The making of this thesis involved a diverse set of tasks: finding projects to study, deepening agile knowledge, studying projects and writing the thesis. The first thing to do was to find public research projects to study. This phase was carried out by sending emails and by arranging meetings with professors to talk about agile and the reasoning of this thesis. It is notable that only a relatively small percentage of contacted people and projects were willing to participate. This may have distorted the overall picture of the state of public research. The result of the first phase was four projects in Tampere University of Technology and two co-operative functional research groups with multiple projects at the University of Tampere.

The goal was to improve public research project management by giving them new viewpoints. Not to tell them how to work, but to challenge old habits. The first step toward the goal was getting to know the projects by way of interviews and observations. Observing project life at the team room and seeing how meetings were carried out. At this point the ambitious goal was to see some changes in the projects involved. For example co-locating a team would be very agile-like change. However, telling others what to do is not efficient. So the ambitious goal was to see people taking an initiative over project management and teamwork and making changes on their own. Teams participated as much as they felt comfortable. No project or team accepted all that was targeted by the thesis. This was a predictable result, but it was thought to give better results than trying too hard. The obvious fact is that projects decided how much they wanted to participate in the thesis and everything that was given was taken. Information on two of the projects was gathered by interviewing only one project member. Information on one project was gathered by interviewing two people with additional opinions from a third person. Information on one project was gathered by interviewing four people and observing project life and the team room. Information on the two co-operative groups was gathered by interviews, informal conversations and email-conversations with multiple people as well as by observing their meetings.

Interviews were the major tool for information gathering. Even ‘official’ interviews were free-form. Basically, the structure was as follows: The interviewee told all that they thought was important while the interviewer wrote down notes and asked clarifying questions when necessary. When the interviewee reached the end, the interviewer went through a checklist of questions and asked those questions that were still unanswered and still seemed important in the given project. Table 3 shows the question checklist.

General questions	*Short description of the project? *What would change? *What would you retain? *What are the priorities? What is important to change or retain and what is not?
Financing	*Where the money comes from? *How is financing related to taking risks? Encouraging? Rejecting?
Vision	*Who is the customer? *Does the project have a well-known vision? How was it conceived and retained? *Who prioritizes what should be done next? *How is project progress monitored and by who? *Are visual graphs, charts, workflow diagrams… used? *Do plans change? How often? *How are results evaluated? *How is the project evaluated?
Team	*Who work here? Roles, locations? *Are team members working on this project full time? *If not co-located, then how is communication handled? *Is current communication good, sufficient or poor? *How is atmosphere the here? Amicable? Hostile? Neutral? *How much autonomy do you have over your work? *How easy it is to get help?
Continuous learning	*How are you executing continuous learning? Technical excellence?
Project Work	*Is there a rhythm to your work? *How much your work and results are or could be incremental? *Are there intermediate results? *How do you share and store your work? Version control? *How much work related to documenting do you have? *How much project upfront planning do you have? *How much are you using empirical tests to find out the best way to solve your problems?
Finally	*Is there anything else that is important to know?

Table 3. Check list of questions for interviews. Questions were asked if considered relevant after a free-form project introduction.

7 Results

This chapter presents summaries of interviews and studies of the projects. Anonymity was promised to interviewees. There is always a risk of distortion, when writing down comments and making summaries. This chapter is as objective and fair as the author was able to make it. The projects are named to help later referencing.

7.1 Project Alfa

Project Alfa is a project at TUT. Its domain is software technology.

7.1.1 Project Initialization

When Alfa was established at least the key people knew each other. Perhaps they have already worked on projects together. Finding partners from industry is an essential part of current research projects. There is a recognized need to change the initialization phase. At the moment, when someone has an idea for research, he starts sending emails. More and more partners emerge. Soon the vision is patchwork quilt of different sites, institutes and goals. It has been though that workshops and more intensive conversations may help to clarify the vision, when done at begin of the project. Nevertheless establishing a clear vision has been difficult.

A patchwork quilt vision is accepted for financial reasons. Money comes from the European Union and Tekes. Sites from multiple countries must participate for getting financing from the EU. Project Alfa has sites from a handful of different countries. The EU also requires that private industry invest into a project. Tekes has yearly changing financing themes. New projects need to fit to current themes in order to get money from Tekes. Tekes also requires that the purpose of a project must be to create a new kind of business. Creating a new business has higher risk, so Tekes does accept high risk projects. It is not known whether financing depends on project staffing? If researcher takes a risk and fails, does it harm the financing of his future projects? Tekes usually requires that private companies participate in the project. The more financing is gathered from industry, the more Tekes is usually willing to invest.

Typically industries invest their human resources to project. Big industrial actors with vast user base were desired to participate to Alfa. These actors were not found.

There is one head coordinator and four country coordinators. The Finnish country coordinator works at VTT. There are three people working on this project at TUT. One has been taking part in the project for a long time, and one has been there for a while. The project was launched by four key people. One of them was from TUT, but he has since switched workplaces. Before the key person left he worked on transferring information to his replacement. Even so, replacing a key person was far from easy.

The goal is not to have a monolithic result. Building up one united result would be too difficult to manage and different interests of different actors would be difficult to merge. Distributed goal is not only a problem, but it possess a risk of a fuzzy or completely lost overall vision. Producing a product is not the goal. Software is built to test theories and to answer other research questions in hand. The results need to be publicly noticeable for credibility and financing. Academic results are measured by the number of publications and by lessons learned. Learned lessons can be taught to students, who will take the knowledge with them to the industry in the future. Sometimes financing is even granted for learning purposes. The higher level goal is to make Europe a better place. Industrial partners have their own measurements of success, such as a 20% performance improvement.

Trying anything wild and risky is rare in public research, therefore showy failures are rare as well. Current research habits are deep-rooted. It might be difficult to get financing if these habits are challenged. Practices of measuring success will change over time, but how and during what span of time is unknown.

7.1.2 Teamwork

The research world never has real teams, from an agile perspective. People in TUT have their own main responsibilities, but the TUT group is collectively responsible that work gets done. Work results are reviewed and problems are discussed at least on some level. However, focus is less than perfect. Team members at TUT are taking part in educational work and they are also working on another somewhat similar project.

Substance related communication is unofficial and is done as individuals want it to be done. There were no signs of information radiators or low-tech high-touch tools in the room where the TUT members were sitting. Communication between TUT and its industrial partners is frequent. Communication between TUT and other public institutions is minor. It was recognized that there is a need for a better discussion forum. Email is not a good means to have conversation, because sending the frequent emails having a conversation would require is considered spamming. Some sort of forum might solve this problem, if people would start using it.

VTT and TUT have steering group phone calls every few months. Enterprise representatives may take part in these conference calls. An email list is also used for communication. The purpose of these communications is to take care of administrative issues.

There are multiple means of taking care of international communication. Around every three months there is email exchange to tackle administrative issues, such as when the next meeting will be held. These emails are informative in nature, not so much an invitation to conversation. There are seminars around once a year or a bit more frequently. Seminars contain coordination, reviewing, auditing, evaluation, problem discussion and demonstrations to sponsors. After the official part there is always an unofficial part for unofficial discussion. Every now and then research parties arrange

wider scope, multi-project conferences. Even though there are conferences and people are rather active in participating, not all team members ever meet each other.

The project connects many people, in multiple countries and sites and the work is done under the same title. However, it is recognized that there is not that much real co-operation between sites and better co-operation could improve results. Competition between research sites can sometimes be a partial reason for lack of co-operation.

7.1.3 Work progress

There are no information radiators or other visual charts to show the state of work. The level of self-organization is very high. Researchers are individually responsible for deciding what to do next, even though opinions are changed over prioritization.

There is no work related rhythm, no iterations. There is an administrative rhythm orchestrated by reports and weekly meetings and a rhythm orchestrated by educational responsibilities.

7.2 Project Beta

Project Beta is a project at TUT. Its domain is software technology.

7.2.1 Project Initialization

The goal of the project is to improve continuous value delivery, continuous feedback gathering and continuous experimenting of the participating companies. In other words, the purpose of the project is to help companies to make a controlled shift toward agility. Creating software is important, but not the goal. Testing theory and creating new business models are important.

Tekes is financing Beta. SHOK is the financing model used. SHOK allows changes in plan [4].

Beta is a four year project that contains software development. The whole project consists of around twenty organizations and ten research institutions. The project is highly distributed. TUT have eleven person-years per year. VTT is another important Finnish partner. The project consists of three rather independent subprojects.

7.2.2 Major Meetings

There are two major meeting cycles. One repetitive meeting is for teams to go through work related issues. Another repetitive meeting is for the steering group.

Work related review meeting takes place quarterly with a duration of two days. This meeting is an extended steering group meeting, with participants from each participating organization. Subprojects demonstrate the progress of the past three months on the first day. Things that were done and things that were left out are put on the table. Progress does not have to follow the plan, but there needs to be some progress. There is a free-form socializing time for building up team cohesion in the evening. The second day is

about planning what should get done during the following three months and what is already known about that work. This is an important meeting for teams to find common direction and cohesion. People are behaving amicably and participating even when it is difficult to leave their daily work to accumulate for two days.

The steering group gathers monthly. It is the most important decision-making body. Its meetings resemble daily standup meetings. Questions of how the project is progressing and whether it is going in right direction are discussed here.

7.2.3 Subproject Teamwork

Subprojects are rather independent business cases. The real work is done in subprojects. Subprojects have one leading company and possible other member companies and research institutes. Leading company is responsible for organizing its subproject. Yet the subgroup is self-organized. In principle Tekes is the client, though of course the whole of Finnish society benefits.

Subproject groups have a teleconference every other week. Members at TUT see each other face to face every week. These meetings needed to be arranged, since people were not seeing by coincide. Weekly meetings consist of a status check and discussion of new possibilities. Creating group cohesion is important, especially because the group is not co-located or focused only on the given project. It is seen as an obvious drawback that the group is not co-located. Having all team members co-located is not possible, because the group members have so many other obligations, such as studies and all sorts of teaching responsibilities. Agile sprints are seen as impractical because of all the other obligations. Successful teamwork is seen as something to work for.

When facing a problem, first a group member tries to solve it. If the problem remains unsolved it is propagated forward one way or another. If problem stays unsolved, it will be propagated to technical lead. It is also notable that the problems that enterprises may see as blockers can be seen as a new improved direction for research. Having multiple organizations makes problem solving and participation decision making more difficult.

Beta is a new project. Conventions are still being settles on. There are many actors, so whose tools or visualizers should be used? What rights should external users have over tools? Who pays for licenses? Who has time to study new tools? These are some examples of open questions. The tools are not selected yet, but it has been decided that some sort of confluence and wiki functionalities will be used.

7.3 Project Gamma

Project Gamma is a project at TUT. Its domain is software technology.

7.3.1 Big Picture

Project Gamma is a EUREKA project [3]. Tekes is offering financing in Finland. Tekes is partially paying the costs of partner companies. Multiple sites from Finland, German

and France are taking part in the project. Even if the vision is clear between the sites, the project is challenged by differing interests.

France is a big player in Gamma. They have a lot of people and huge companies involved. The project coordinator is from France and concentrated on France's goals. Moreover, Gamma is a continuation to another French project. Germany is participating with three companies. Finland is participating by a couple of high tech companies as well as TUT and the University of Helsinki.

Finnish sites have not co-operated a lot with each other. On the other hand co-operation has been working well between TUT and foreign partners and especially with the Germans. Finnish and German sites have positive interdependence. Finns benefit from German results and vice versa. France and German have major architectural integration issues about proof of concept. So far this has had no effect on Finnish sites, but soon it will, if the issues are not resolved.

Learning from each other is an important motivator to have a project distributed over multiple countries and sites, not just software development. A highly distributed group challenges communication. Project personnel try to meet each other three to four times a year to do planning. A varied number of people participate in these meetings. Demos and reviews are widely used practices for synchronization and learning. Publications are the only documents that are not avoided. Retrospectives are not used. Moments of failure are used to improve processes. The high rate of personnel change gives the project an extra challenge.

7.3.2 TUT Related Work

There used to be three people working on this project at TUT. Participating people have been from relatively to extremely experienced researchers. Due to personnel changes there is only one person concentrating on this project at TUT at the moment. This person is new, so he needs to spend time on learning. Shortage of human resources is an identified problem.

A weekly meeting is the most important mean of communication and co-operation within TUT. Prioritization of future work is happening in weekly meetings. The same person is a kind of product owner and scrum master. Business objectives are not as well defined as in industry so it is considered that the same person having the roles of scrum master and product owner is not so harmful. If the TUT group acquires any more members then these roles will be dedicated to different people. Finland sites try to have regular meetings as well. Meetings are about what has been done lately and what should be done next. Neither of these meetings is very regular in reality, due to the other responsibilities the project members have. Meetings make the project work feel slightly similar to sprint iteration. There were one to three week sprints with German partners. Retrospectives are not used at this level either. Problems are solved as they are identifies. No information radiators, nor other visualization were used.

7.4 Project Delta

Project Delta is a project at TUT. Its domain is signal processing.

7.4.1 Big Picture

Project Delta is a collection of one person projects. All the projects serve a common aim. There is a vision owner whose responsibility it is to make sure the one person projects are all targeting the same goal. Basically Delta is an endless project, since the end of one subproject is a starting point for another subproject. The vision does not change over time, but obstacles force a change of course in actions every now and then. There are two aims in total. The scientific one is to study a certain phenomenon. The other aim is to help students develop toward excellence. Team members are not fully aware of the aims.

Financing comes from the department. Project success is measured by how often research has been referenced by others and by the esteem of the journals the research is published in.

7.4.2 The Team

The team consists of the project champion who is the vision owner, a technical lead and six team members who work on single person projects. Team members are mainly students and rather new to the project Delta. Delta is a cross-functional project that can be done incrementally. The project champion encourages team members to study cross-functional skills. The team members work on project Delta full time, so their focus is excellent. The team members are co-located and their working phase is intensive. They even had use of low-tech high-touch tool, namely a whiteboard. There was also work related communication in the team room. It was recognized that the team is rather new and things will evolve, but the team members are nevertheless rather happy with their team. The room was small and air was stale. The mood felt intensive, maybe a bit busy and joyless. All in all it felt like the project had a rather strong and instant resistance to new ideas about project management. However, observations are only snapshots in time, so heavy conclusions cannot be drawn.

There are lots of students working in project Delta. It has inevitable effects for project management. One big aim is to make them good researchers. Students are aware of only the vision of their own subproject. The first task for student is to get familiar with technology and how the work of others is linked to their work. Understanding the overall vision becomes important later. When students face a problem, they first try to solve it themselves. If they cannot, then problem is propagated to the project champion. It seemed like the project champion was eager to help the team members and would find the time when a team member needed help. Results are sent to the project champion for review.

The project champion works on prioritization, estimation and big picture planning mostly by himself. The students and the other team members can share their opinions with the project champion. Executed work is slightly path-like: first step X, then step Y, then step Z and so forth. Result or intermediated result of one subproject is often input for another subproject. Sometimes a subproject needs to wait on another subproject to get the input needed. When this happens, people with the extra capacity have other works to do.

Delta has some form of retrospective. The number of participants, frequency and content was dependent on who was asked and the answers were somewhat nebulous. Whether formal or informal, regular interval or random interval, focused on the team or the individual, it was consistently stated that there existed a habit of conversation on how to improve.

7.5 Research Group Epsilon

Research group Epsilon is a collection of two functional research groups at the University of Tampere. They are nestled under the same label, since they work closely together. Epsilon's domain is medical research.

7.5.1 General Discussion

The employees are motivated and committed. They are doing important work in order to help people. Student/seasoned -ratio is quite high in Epsilon compared to software companies, but still not exceptional among the studied projects. People are divided into functional groups. Project groups are gathered from functional groups as needed. Staff on the project is not co-located and specialists on one field are working only on issues related to their specialism. No use of low-tech high-touch visual tools can be seen at the office. One reason for not having co-located teams and visual low-tech tools might be that some seasoned employees feel like they know how to implement commonly used tests. No need to get help for solving problems or to transfer knowledge. What to test, and what the results mean are more challenging questions than how to implement tests. Human resourcing between projects might be a bit obscure time to time. The same people may be working on multiple projects with different staffing. One reason is that breaks, even long ones, are common in this type of research projects. For example, calculating analyses or peer evaluations or growing cell cultures may take time. So when one project is on hold, it is convenient to work on another project for a while. There are numerous smallish projects under a rather wide research domain. Occasional lack of clear prioritization was recognized. Even if employees are not focused on one project goal and they are not co-located into the same team room, they are still mainly in the same building and focused more or less on the same research domain. It is recognized that narrowing down the research domain of the projects could lead to better synergy.

Urgency and the everlasting race for funding are shaping their work. It is hard to try something new and risky and change habits, when people are always busy and when funding is unsure. Sometimes public funding is too rigid and not well-suited for explorative type research. Losing financing is not the only reason to hurry. Some other party publishing before you would collapse the value of your study. One recognized problem is how long peer evaluations are taking, which is a very Lean-like issue regarding throughput.

Retrospective type meetings were tried by one functional group, but it did not work well. Impediments were solved at the same meeting. Members of many projects were participating in the same retrospective, so whatever project was gone through, most of the participants were not involved. In the end this meeting was canceled since it was inefficient. A second functional group had their one hour status check and problem solving meeting once a week. Again members of many projects were participating. There is a clear need for a media for impediments discussion. People participated surprisingly well during these meetings. Still, no surprise to anyone, that many participants were quiet most of the time. They may even go to have some coffee and then come back. Even if the meeting has good value as it is, these symptoms may suggest that the value could be improved on.

Strategy day is held once a year. For example, brainstorming has been used to discover what to study next. Other questions, like where to focus, have been discussed as well. Overall feeling about the strategy day seems to be good. Some suggest that the results are either not good enough or not enough action has been taken to make the strategic day's outcome real.

Epsilon has no real teamwork culture, but an individual work culture. The situation is rather typical for the studied projects. Even with project groups, each member has his responsibilities and the work is done individually. Furthermore, experience from research world has shown that bigger groups tend to have duller and lamer results than fierce and stubborn individuals. On the other hand, it feels fair to say that Epsilon, as so many other organizations, is lacking in good teamwork skills. It can be seen for example in individuals owning ideas, or human interactions not being seen as a crucial part of innovations and learning or how the participatory decision making is working.

The employees' opinions are divided when it comes to changing the ways people are working together and the way the projects are managed. Seasoned employees tend to favor current methods and resist major changes whereas newer employees tend to think that processes should be improved. For example the following needs for improvement were raised: feeling alone in a project, need for better communication, projects having a clear starting point, in progress time and a clear end. Managers of functional groups seem to be open-minded and eager to hear new viewpoints. On the other hand it looks like the ideas are not that often tested in practice. With some people it felt like objection is an instant response to ideas that would change their current habits. However, when challenged, it was astonishing to see how someone thinks through an idea they are opposed to and answers more or less the question he was asked. Does not sound much

in the land of unicorns, but with humans it is a lot to ask. The author's opinion, supported by the chapter *What People Believe and What Changes Their Mind*, is that people tend to answer the question "Do I like that", no matter what was asked.

Coming chapters describe the studied projects in Epsilon. There are plenty more projects in Epsilon. Projects described are quite different from each other. Projects are named Epsilon One, Epsilon Two, Epsilon Design and Epsilon Single.

7.5.2 Project Epsilon One

Depending on how calculation is done, the project has at most seven participants. The participants also have other projects and they are located on two different floors and multiple different rooms. So, not co-located, but at the same site at least. Many highly competent people are taking part in the project.

The project has a lot of potential. It has highly experienced members and certainly some good results will come out. When it comes to project management, some feedback shows that the project has supported rapid occupational learning. Aside from all the good parts, there are significant challenges with group dynamic. It seems like there is a shortage of high quality communication, roles are a bit unclear, a shortage in amicability, rivalry between team members, lack of participatory decision making, idea ownership and unclear goals and constraints. The project is worked on by professionals, so it is going toward its goal, but there could be lessons learned for future projects.

A lot of communication is done via face-to-face talking, but emails are a significant means of communication as well. There used to be a regular project meeting, but some felt that it was not useful, so it was discontinued.

7.5.3 Project Epsilon Two

The group has three seasoned members from two disciplines. Two of them are located in the same room and one is about twenty meters away. The project vision is said to be clear to all. Analysis, tools and programming languages are all familiar, so getting stuck or needing help or opinion of others is rare when executing tests. Test results are a focal point of interest. Sometimes results, intermediated results and their effect on coming research are discussed with group members or in pairs. Google docs is used to preserve findings.

The end result will be from a four to six page long manuscript. Findings need to be clinical and reasoned. Contacts from scientific magazines evaluate the manuscript. The more esteemed the publishing magazine is, the more successful the study is considered to be. The more the articles of a magazine are referenced, the more esteemed the magazine is considered to be.

7.5.4 Epsilon Design

As reasoned before, small groups can be better than individuals in solving complicated problems. When it comes to research, deciding what to study and how to study are

complicated questions. The design team is not considered a team in Epsilon, but in the author’s opinion, it has quite a lot of team-like characteristics. The participants combine their knowledge in order to create something new. Only a few seasoned researchers are participating in this free-form unofficial undefined talking-talking group.

What does Epsilon Design do? Someone has an idea that generally emerges from earlier discussions, readings or other human to human interactions. This is the case even if the one with the idea does not recognize it, as can be loosely drawn from the chapter *What People Believe and What Changes Their Mind*. The idea is played around with in a design team. For example, the following questions could be answered: Is some other party studying this? What type of results would we expect to have? Why are the results important? Precisely what should be studied? Are the prices of some experiments coming down? Should we find partners? Is this study feasible or not?

7.5.5 Epsilon Single

One person projects were not studied. However, they are so common that many opinions and experiences were heard. Because they are so common, it seems appropriate to have a few words about them.

Particularly students are working on one person projects. Doctoral theses and master's theses are examples of one person projects. Financing may even force people to work their one person projects solo, instead of helping each other and working together. These projects were said to feel lonely. Since the subjects of the projects are not so near to each other, it is hard to share opinions with other students. When stuck, it takes time to get help. Often getting help means to arrive at a solution. It is not about having participating conversation about the solution, which in turn could lead to better learning.

7.6 Summary of Projects

Table 4 helps the reader to form an overall view of the studied projects. Table 4 summarizes what was written about Alfa, Beta, Gamma, Delta and Epsilon.

The projects were rather different from one another. Large tolerances must be accepted to create a summary table. For example, what is the project team size in project Beta? Is it the people working at TUT or all the participants? All the participants were chosen, but other solutions may have been justified as well. How multidisciplinary a project is or are there lots of students in a project? These are all rather ambiguous classifications. Classification ambiguousness should be kept in mind when reviewing table 4.

	Alfa	**Beta**	**Gamma**	**Delta**	**Epsilon One**	**Epsilon Two**	**Epsilon Design**	**Epsilon Single**
Field	Software	Software	Software	Signal processing	Medical	Medical	Medical	Medical
Use of retrospectives	No	No	No	Yes	No	No	No	-
Project team size	> 10	> 10	> 10	Seven	At most seven	Three	Around four	One
Collocated	No	No	No	Yes	No	No	No	-
At the same site	No	No	No	Yes	Yes	Yes	Yes	-
Internationally distributed	Yes	Yes	Yes	No	No	No	No	-
Focused	No	No	No	Yes	No	No	No	-
Very high student percentage	No	No	No	Yes	No	No	No	Yes
UTA and TUT personnel participates teaching	Yes	Yes	Yes	No	No	No	-	-
Has private partners	Yes	Yes	Yes	No	No	No	-	-
Strongly multidisciplinary	No	No	No	Yes	Yes	Yes	Yes	-

Table 4. Summary of projects.

7.7 Discussion

This chapter sums up findings. The first subchapter *How Agile They Were* is the most important. Rest of the chapters give additional viewpoints to some common issues.

7.7.1 How Agile They Were?

As mentioned in the chapter *Thesis's Theory of Agile*, the team needs to live out two values to be agile:

1. Complicated problems are better solved by small and equal and intensively collaborating groups than by individuals.
2. Continuous improvement.

The result is that all studied projects had some co-operation. For example, when one gets stuck, there is a mechanism to get help. However, none of the studied teams were working intensively together. They were not actively throwing ideas and building a shared theory of problem. Actually, none of the studied teams were a real team. They were more projects staffed by people.

Only one studied project tried to improve continuously. Even that project did not have a well-defined way for continuous improvement. Different project members described slightly different processes. However, it seemed obvious that they do try to improve their performance continuously.

If the thesis' theory to agile is accepted, then public research is not particularly agile. If agile business objectives are accepted, as described in the chapter *Agile Business Objectives,* then public research would gain benefits by being more agile. Public research should focus more on creating real teams and a solid mechanism for continuous improvement.

7.7.2 Funding and Reasonable Risks

Public research should refine its relationship with risk and experimenting. Relatively high risk taking and experimenting should be at the heart of public research. Working habits should be experimented on as well. The same goes for education. The author's understanding is that possible failures and relatively high risks are rather avoided in the studied projects and in public research in general. If projects or experiments never fail then not enough risks were taken either. There should be a culture of taking reasonable risks. Some companies celebrate projects that were terminated. For example Super Cell is said to do so. Terminated projects prove that risks were taken.

Participation of privately held companies has a rather big role in public research nowadays. Using the same approach and mindset with all companies can be inefficient. See chapters *Rightshifting* and *Psychology of Change Resistance* for further reasoning. Industrial partners were discussed as industrial partners. No differences were made between different types of industrial partners. So it is unclear whether all industrial partners are considered more or less the same or is there another reason, such as

confidentiality, for the generalization. A big and more bureaucratic corporation will probably be a big and bureaucratic research partner. A small and adaptive company will be a small and adaptive partner as well. Different partners have different needs and therefore the same service does not fit for all. Agile does not work as well with big, hierarchical and bureaucratic partners as it does with small and entrepreneurial partners.

Public funding was an issue more or less with all the studied projects. They all had public financing in one form or another. Almost all projects recognized various shortcomings with how public financing is distributed. It was also somewhat common for project personnel to wonder how the funding system could be changed or how to successfully apply funding in a new way. If risk taking and experimenting are in the focus of public research then financing should be made supportive. Having public financing may be cumbersome, if one wants to try out new ways of working. Probably *Blub paradox* works here as well. It does not matter how good the idea is if the sponsor does not understand it. Perhaps the sponsor does not even want to understand it. If bureaucrats are giving money, then you probably have to be bureaucratic to please them. A single research team cannot change the way money is given. Working with rules is always gaming with rules. It is true when considering the likelihood of getting a speeding ticket and it is true when gaming measurement metrics to get higher bonuses. Perhaps bureaucratic money can be spent on something that is not bureaucratic, but it may need sophisticated gaming of the rules. However, at least Tekes is said to accept risks.

7.7.3 Students and One Person Projects

Students and one person projects are combined under the same chapter since students are largely working solo. The chapter *The Background of Public Researchers* delves deeper into the reasons. Education and public research has an obsession with making people demonstrate personal accountability even to the point where it has serious consequences on teamwork skills. This is especially true with students. Students may not have the experience needed for executing research tests efficiently. Students may be able to perform less structured brainstorming and visioning to solve non-linear problems. They should have a more rebellious and ideological attitude to changing the world. They certainly have more potential for becoming something greater. With or without this reasoning, it should be a no-brainer why successful student guiding is socially a key to success. First, not allowing students to participate in visioning may not be optimal for their growth or for the solution. Students should learn visioning skills and that ideas are not owned. Perhaps students can be made to participate more by better facilitation. Second, teaching students to be solo players is harmful for the students and the society. Small groups are better for learning and innovation as described in the chapter *Social Aspect of Productivity*. Agile supports this idea, by requiring real teams.

What would be a more valuable skill than to have entrepreneurship? Perhaps some student works, like thesis or courses, could be replaced by start-up type works. The

educational focus would not be on theoretical competence or comprehensive documentation, but on learning what real teamwork is, on thinking outside the box and on studying something interesting by spike solutions. Especially theoretical and technical competence is important to learn, but it may be even better learned this way. There have been start-up-like degrees. For further information or to hear experiences, contact Saimaa University of Applied Science or ProAkatemia at Tampere. Some rumors even say that some VTT projects are like start-ups. This section contains lots of options and no answers. This is well aligned with agile. When dealing with complicated environments, one solution never works for all. Testing solutions and adapting is the way to do it.

Single person projects have their place in education, however when learning and innovation is needed, the only real amendment to single person projects is to make them less solo. Paired projects or two one person self-organized projects building one solution could be examples of improvement. Certainly there is no easy solution and all the solutions have their real drawbacks. That is life and it goes with all solutions everywhere.

7.7.4 Teams

There are no real teams in the studied projects. Here are listed common notifications about teamwork over the studied projects. First, throughput should be focused on more than workforce utilization. It is better to work one project from start to end and then move to another project than it is to have multiple projects in progress. Having multiple projects in progress is almost universal to public research. When multiple projects are the way of life, it would be better that all participants are focusing on one project at time. It would be even better if they were co-located as well. For example, people from TUT and VTT are working on their project for two weeks every three months and when they do, they are co-located and focused.

International projects are special cases of distributed projects staffed by people with different interests and multiple projects. There are lots of dysfunctional characters in these projects. However, the funding is not the only reason to accept international projects. There is a great potential of learning from each other and of gaining a new perspective and synergy. Then how could international projects be improved? International co-operation should be improved so that people with similar interest could find each other. How to do this, is way out of the author's competency. When a project is so big that there are local teams, then problems diminish. On the other hand the potential diminishes as well. It would be good for the research, if researchers would locate to their team. For example researcher from TUT would move to Germany for a two year project. Life is not only working and often it is hard to leave. Students, on the other hand, may be doing student exchange. Let us say there is a project that lasts a few years. It contains three local teams in three countries. Each team has three researchers and three students. When hired, students may be required to do, let us say, a six month

exchange to another project. This would combine local teams, internationalism and educational purposes.

Cross-functionality is another aspect of a real agile team. Cross-functional is a term used in software development. The university world may use the term interdisciplinary. The meaning and benefits of cross-functional teams can be read from the chapter *Cross-functional*. In general, cross-functionality seems to be rather well used in public research compared to software development. Interdisciplinary could be a great advantage of public research. Oftentimes universities are interdisciplinary by nature. Perhaps this could be one viewpoint companies might be willing to invest in. Depends on the circumstances, but for example anthropology, social psychology and marketing may all be connected to software development.

Collaboration is not heavily used in many of the studied projects. For more information on collaboration, please read the chapter *Collaboration*. Collaboration cannot happen, if there are big egos. The essence of an ego is not understated here, but egos that are too large will not work jointly together. Practicing may help big egos to work better together. For example pair-programming is said to have this sort of effect. However, this may be a painful process and if benefits are not clear then why bother? If big egos must be smoothed over, then threats should be studied beforehand to make the pain as minor as possible. More information on pains and threats can be found in the chapter *Psychology of Change Resistance*. Enjoyment and joyfulness are signs of collaboration. Based on the author's limited experience, professors tend to welcome ideas and criticism more constructively than people on average. A trick that may work to improve collaborating is to concentrate on performance. Not at any cost, but instead of worrying who can work with who and who is willing to do what, more effort could be transferred to concentrating on higher performance. [7]. A trick that may harm collaboration is to make some team members more important than others. For example, if you have your name first in the published manuscript, you are more important than, if your name were fourth in the list. Agile team members should be equal to each other. For instance, if performance is measured, then team performance should be measured, not individual performance. Researchers seem to have a very strong sense of self-discipline. No matter what obstacles are thrown in the way, they keep on trying. In many cases they also demonstrate very strong self-organizement.

7.7.5 Vision and Project Management and Customer

A project cannot succeed without vision. Vision and agile is discussed in the chapter *Customer and Vision*. At least these instances have influence on vision in the studied projects: EU, Tekes, TUT, University of Tampere, numerous enterprises and researches. It is said time over time that the goal is not a product, but acquiring new learning and testing theories. So is there sufficient vision? Who owns it? Who nurtures it, so that everyone in the project is familiar with the vision? These are questions that the author does not have an answer to in many cases, since the author does not know these projects

in depth. It may be that the vision is too fuzzy and the project is drifting, or it may be that they all know where to go and a light touch is all that is needed.

Research departments and centers should also have well-coordinated direction for research [35]. Having numerous small projects and no obvious client, easily blur the direction. A project portfolio may be a helpful tool for maintaining well-coordinated direction.

Consider the chapter *Rightshifting*. There are different types of organizations. An adequate vision sounds and feels different in mechanistic than in chaordic organizations. Some of the studied projects were mainly in a mechanic state, but some might have been in a synergistic or even a chaordic state. There are a lot of mechanistic characters in the research world. The research world is controlled by rules and regulations: peer evaluations, published in which paper, how many publications, whose name was on which paper, climb the ladders by doing thesis, get grades and so on. Some groups do have a strong flavor of mechanistic state. On the other hand, the research world is labeled by freedom. A single researcher may have a big impact on what to study. They may have decades of research experience. The language they use may contain rather open and abstract wondering on greater goals and principles. They may note constrains and then move on to discuss reasons. When only seeing a snapshot, ad hoc and charodic may seem the same. Even though it seems obvious that some people in the research world are beyond mechanistic state. Organization needs to improve continuously to rightshift.

Agile promotes short cycles. A shorter cycle improves learning and payback time. Read chapters *Rhythm* and *SCORE* for more information. There are projects as long as four years in publish research. Agile suggests splitting long projects to value adding subprojects. For example, one four year project to four one year projects so that the result of each one year project has value on its own. One good way to shorten a long project is to make team more focused, as described in the chapters *Focused for Business Value* and *Lean*. Another good way of shortening projects is to end them at some point. Have an end ceremony so that it is clear the project has ended. It is not too uncommon that projects never end, but they tend to fade out little by little. To try something wilder, spike solutions could be a modern way of shortening some projects and of doing unorthodox research. There has been discussion to the point that to be successful, most start-ups should publish their product at the earliest moment possible and that the number of early users is indicative of success. More or less when spike solution is ready, a lot is done to make many users to try it out. Start-up-like spike solutions would improve early feedback and if done in a user friendly way, they may make more people interested in science and in contributing to its progress.

Retrospective is barely used in public research. It has been tried by some groups. Oftentimes it has not worked well. Two reasons can be recognized. Retrospective requires lasting and focused teams in order to work. First, if there is no lasting team, then there is no continuum. There is no lesson to learn, if the same people are not participating in retrospective after retrospective and the same people are not trying out

what was decided. Second, if there is no focus, then participants are optimizing their own interests and are not so much interested in others. On the other hand, if participants are not willing to change their habits, then the retrospectives are not working. Retrospectives are the minimum amount of iteration every project should have.

It seems like some people are chronically busy. They are worried about one hour or even fifteen minutes. Being too busy makes it practically impossible to change habits as reasoned in the chapter *What People Believe and What Changes Their Mind*. There should always be time to discuss new ideas. There should also be time to have breaks for free form discussion. Informal chit-chat can easily improve overall performance. Perhaps critical thinking, as valuable as it is, is also researchers' occupational disease. Having enough slack time and having enough chit-chat may treat that disease.

Information radiators and other low-tech high-touch visual tools are barely used in the studied projects. Tools like the ones described in the chapters *Work Flow Visualization* and *Facilities in Agile*. These tools should be at least tried out to see whether they provide any help or not. They have proven to be very useful in many software development projects. There are electronic visualizing tools for distributed teams.

Researchers seem not to do maintenance work. However, they do have multiple responsible, like multiple projects and educational duties. If there is one major project, then all interruptions may be considered as maintenance work. This may or may not be a helpful analogy. The chapter *Agile and Maintenance Work* describes software development ways of dealing with maintenance work.

A superior reviewing the results is common with the studied projects. Groups are small and researchers do not know what the others are doing, so review done by one's superior can be handy. It is also good practice that results are reviewed somehow. However, to have only superior review is not an adequate form of feedback and peer support in many cases.

7.7.6 More on Project Delta

Project Delta seems to have some incredibly agile aspects. Such a situation is not an outcome of random acts. They must have worked for that. How could it be taken even further? Delta has a lot of students. Some say that there has to be at least one seasoned project member for every five juniors in agile. Delta is pretty close to this. It is worth considering whether the current single person projects are the optimal way of achieving their greater aims.

Working toward the same goal and participatory decision making improves teamwork skills. It improves understanding of the big picture of research and of one's own field of study. It may increase the feeling of relatedness, which is rewarding based on SCARF. Self-organizing and participating on estimating and on decision making would improve autonomy which is also rewarding according to SCARF. On the other hand, working more closely together may cause status threat or fairness threats as stated

by SCARF. Rules of engagement, as described in the chapter *Team Rules of Engagement*, are a good tool for dealing with fairness threats. Working toward the same goal may make sprint iterations possible, which in turn may make incremental value delivery possible. Furthermore, themes could be chosen for different sprint iterations. When done so, the result is not little here and little there, but something concrete is finished at the end of each sprint iteration. There are many question marks and only one way of finding out how would it work.

Having that many students working on the project makes things, like participatory decision making, more complicated. It may even increase the workload of the project champion in the beginning. If the bigger aim is to teach students to be excellent researchers, doing a bit of extra work in the present may pay off big time in the future. Some say that every rule the project manager has is originated from a bad experience. The rules are what project managers use to avoid bad things happening again. Control and freedom, rules and trust, are all balancing acts.

This project team is in need of a better room for working. The current room seemed to be too small and the air was stale.

7.7.7 More on Research Groups Epsilon

Most of the discussion regarding Epsilon has already been dealt with in other chapters. This chapter tries to highlight a few issues and gives a few Epsilon tailored suggestions.

There are functional teams and somewhat intensive co-operation between them. One solution would be to make one cross-functional team that owns all the small cross-functional projects. Teamwork on such a team may be easier to improve than multiple groups of patchwork quilt projects.

Maryland university has successfully used agile to mentor doctoral students. Their circumstances are similar to one of Epsilon's functional teams. How to have enough time to help students when resources are limited and how to help students to track what each of them is doing? Epsilon solved this by a status check and a problem solving meeting. Maryland went a bit further. Please read the chapter *SCORE* for more information.

Epsilon has an unofficial design team called Epsilon Design. Where does Epsilon Design meetings take place and does it matter? If Epsilon Design were a co-located team with some students in it, then the students would be a part of research design even if they are not active. They would be in a team room and they would hear how planning is done. When they have something on their mind, they would participate. Even if they are not focused on listening they would still hear something. Everyone in the room would have a rather good understanding of what is going on. Being co-located would be a low ceremony way of committing students to research design.

Implementing an agile framework is not possible in Epsilon. Implementing agile overnight would probably lead to long lasting chaos. Here are some baby steps toward agility. These steps are the author's opinions tailored for Epsilon.

1. Small groups are better in solving complex problems than individuals. Small groups also have the potential to improve innovation and learning as discussed in the chapter *Social Aspect of Productivity*. The first step is to find out what is so complex in our work that it would take a team to solve it? A seasoned employee would answer this question differently than a junior employee. The one who is doing the work should answer. For the junior it could be how some experiments are run.
2. Who wants to do teamwork? If one is definitely opposed to teamwork, then he should not have to do it.
3. The vision, the answer to question *what*, needs to be clear to the whole team.
4. The team should be empowered to self-organize, to answer the question *how* by themselves.
5. The team should make the rules of engagement. Read the chapter *Team Rules of Engagement*.
6. Real teams and iterations are at the core of Agile. Steps from one to five are about real teams. A minimal way of starting up with iterations is to have a retrospective every two weeks as described in the chapter *Retrospective*.

7.8 Sources of Error

Let us start with the author's know-how. The author has three and half years of experience with more or less agile projects, but no experience of a truly agile project. How able is he in recognizing the needs of other projects to be truly agile? The author has done his best to mend the lack of experience by studying agile a lot and having many conversations about "true agile". However, the author has read only pro-agile books and mainly chit-chated about agile with pro-agile people. There is a reason to believe that the author's opinions are biased in favor of agile.

The practices to gather data for the thesis are error prone. First, only snapshots were taken. No long term, living with the team through their daily life, was done. Second, only one person was interviewed in some projects, which makes the result unilateral. Even if more people were interviewed some people had more time and more opinions, which also skews the results. Third, questions made were not always open enough. Especially, when the author did not understand the answer or the interviewee did not seem to understand the question, the author may have ended up summarizing his own thoughts and asking close-ended question. When the author noticed this and received "yes" or "no" answers, the answers were left unnoticed. Still, quality of the questions had a big impact for sure. Asking truly open-ended question is extremely hard. Fourth, the author was a stranger to the project teams. He just showed up and started asking questions. Did the project team members have a real reason to trust the author? There must have been a lack of trust, which must have caused distortion to the answers. Fifth, only so very few projects were studied, that no generalizations can be made.

This thesis is focused on offering opinions about teams and project management. There is never an exact result when dealing with people. People and their living environments are too complex for that. This does not diminish major sources of error in used practices. It does highlight the target accuracy. The result is not and is not supposed to be accurate. The result is supposed to offer viewpoints. Besides you cannot tell people what to do. They would not act as you told them to. It is better to help them to demonstrate how the work should be done.

8 Conclusions

There is a rather large scale of variation from one project to another, which makes it hard to have one conclusion true for all. The conclusion is drawn from generalization within studied projects.

Researchers carry responsibility over their actions. They are highly capable of doing individual work. They tend to be intrinsically motivated and to believe that they are working on something that matters. This is true even when the funding system is challenging and success measurements would not reflect the core reason of public research. Public research does not tend to value face-to-face communication or intensive teamwork nearly as much as agile suggests. Usually they do not have clear a customer role. Perhaps because of the listed issues, researchers are not focused on just one project and one goal. Not even close. The more distributed by location and by focus the group is, the more open-mindedness it seems to have for new ideas. Perhaps the more people are taking part in education, the more open-minded they are to discussing new ideas? The more same site, co-located and focused the team is, the less they seem to have a culture of being open to chat about new ideas. Agile team seems to require focus of a co-located team and open-mindedness of an unfocused group. The situation is divided when it comes to being an agile project lead. Project leading was not highly aligned with agile in cases where project leading was made concrete to the author. In other cases, based on the information gathered, it is hard to say much about a project leader's role. When it comes to the thesis's theory of agile, public research is not very agile. There is no intensive teamwork or much continuous improvement.

Is agile a good fit for public research? Deep down agile is mostly psychology and group dynamics in a complex environment. Agile is optimized for software development, but agile-like doctrines can be found in other industries as well. For example Lean shares pretty similar values to agile. Anthropology and psychological and social psychology studies explains the human side of agile rather well. Agile is about gaining competitive advantage by learning and solving problems together and by doing it fast. Agile is about fears and rewards. It is about accepting complexity, that all things cannot be foreseen. Military works in a complex or even a chaotic environment so it is no surprise that many well-known generals have been quoted by agile books when it comes to planning. On the other hand, high risks and big projects require more structure and thus are less optimal for agile. High risks and big projects as well as acceptable risks and small projects happen outside software development as well. Agile is nothing special and it is mostly not about software, but human beings. What makes agile remarkable is how well it has been tested in real life over the years. Strongholds and shortcomings have been exposed. Developing software is very complex and oftentimes it takes a dedicated team to do it – ground zero for agile.

Agile does not work with victim attitude. Some people behave like they are born unlucky, like they are victims always finding excuses and blaming others, claiming they are in an impossible position to make any changes themselves. With them, agile may indicate problems fast. Agile will not make them work better, at least not without changing their attitude along the way. Agile would rather make them produce worse results. This has also been demonstrated in real life. Agile works when there is actor attitude. When people are intrinsically motivated and when they are brutally honest with reality and when they demonstrate ruthless self-discipline and want to constantly improve. Agile works with people who feel like they can and they should and they will make a difference. There are things that are beyond their reach, but there are always things to be done. When it comes to having challenges at work, instead of playing the victim, agilists tend to think there are frankly three options: "*Accept it, change it or leave it* [46]".

9 References

[1] Naur P. Computing a Human Activity, Programming as a Theory Building. 1992, ACM Press. 630 p.
[2] Highsmith J. Agile Project Management. Second edition. US 2010, Pearson Education Inc. 392 p.
[3] [accessed on 13.4.2014]. http://www.eurekanetwork.org/
[4] Ministry of Employment and the Economy, Finland. 19.4.2013. [accessed on 13.4.2014] http://www.tem.fi/files/36546/SHOK-kehittamislinjaukset_26042013.pdf
[5] Amatriain X. 2009. [accessed on 27.2.2014]. http://xavier.amatriain.net/AgileResearchManifesto/.
[6] Amatriain X. [accessed on 27.2.2014]. http://technocalifornia.blogspot.fi/2008/06/agile-research.html.
[7] Cockburn A. Agile Software Development the Co-operative Game. Second edition. US 2007, Pearson Education Inc. 467 p.
[8] Shore, J., Warden S. The Art of Agile Development, First edition, Sebastopol CA, O'Reilly Media Inc, 415 p.
[9] Pavlina S. [accessed on 4.3.2014]. http://www.stevepavlina.com/blog/2005/06/self-discipline/
[10] Cohn M. Agile Estimation and Planning. US 2006, Pearson Education. 330 p.
[11] Kniberg H. [accessed on 5.3.2014] http://blog.crisp.se/wp-content/uploads/2013/08/20130820-What-is-Agile.pdf
[12] Johnson J. Build only the Features You Need. XP 2002 Conference. Standish group study reported at XP2002 by Jim Johnson, Chairman
[13] Shalloway A., Beaver G., Trott J.R. Lean-Agile Software Development Achieving Enterprise Agility. US 2009, Pearson Education Inc.262 p.
[14] Hickc M., Foster J.S. Adapting Scrum to Managing a Research Group. 2010. University of Maryland, Department of Computer Science. 9 p. [accessed on 13.4.2014] http://www.cs.umd.edu/~mwh/papers/score.pdf
[15] Nummi P. Fasilitaattorin käsikirja. Helsinki 2007, Edita Publishing Oy. 126 p.
[16] Brown J.S., Duguid P.The Social Life of Information. First edition. 2000, Harvard Business Review Press. 336 p.
[17] Kahneman D.Thinking, Fast and Slow. 2013, Farrar and Straus and Giroux. 512 p.
[18] Rock D. SCARF: a Brain-Based model for Collaborating with and Influencing Others. The NeuroLeadership Journal 2008 Issue 1.
[19] Molenda M., Kovalichick A., Dawson K. Education and Technology: An Encyclopedia, .Cone of Experience, 2003.
[20] Strauss V. Why the 'learning pyramid' is wrong. The Washington Post 2013. [accessed on 13.4.2014] http://www.washingtonpost.com/blogs/answer-sheet/wp/2013/03/06/why-the-learning-pyramid-is-wrong/

[21] The Learning Pyramid. Belmond-Klemme Technology PD Wiki. [accessed on 13.4.2014] https://bkpd.wikispaces.com/04++The+Learning+Pyramid
[22] Butera F. [accessed on 12.3.2014] http://butera.socialpsychology.org/publications
[23] Buchs C., Butera F., Mugny G. Resource Interdependence, Student Interactions and Performance in Cooperative Learning. Educational Psychology 24(2004)3, p. 291-314.
[24] Buchs C., Pulfrey C., Gabarrot F., Butera F. Competitive conflict regulation and informational dependence in peer learning. European Journal of Social Psychology 40(2010)3, p. 418-435.
[25] Butera F., Caverni J.P., Rossi S. Interaction with a high- versus low-competence influence source in inductive reasoning. The Journal of Social Psychology 145(2005)2, p. 173-190.
[26] Auriemma A. Why the Best Offices Are Like Jails. The Wall Street Journal 2014. [accessed on 13.4.21014] http://blogs.wsj.com/atwork/2014/03/28/why-the-best-offices-are-like-jails/
[27] Marshall B. The Marshall Model of Organisational Evolution. [accessed on 13.4.2014]. http://fallingblossoms.com/opinion/content?id=1006
[28] Gyllebring T. Introducing Rightshifting – a conversation on work, effectiveness and joy. Scan Agile Conference, Helsinki 11.11.2013. [accessed on 13.4.2014] http://vimeo.com/84065968 [12.3.2014]
[29] [accessed on 13.4.2014] http://c2.com/cgi/wiki?BlubParadox
[30] Brodzinski P. Against Rightshifting. 2012. [accessed on 13.4.2014]. http://brodzinski.com/2012/12/against-rightshifting.html
[31] Adkins L. Coaching Agile Teams. US 2010, Pearson Education Inc. 315 p.
[32] Kallio K.M. ”Ketä kiinnostaa tuottaa tutkintoja ja julkaisuja liukuhihnaperiaatteella...?" - Suoritusmittauksen vaikutukset tulosohjattujen yliopistojen tutkimus- ja opetushenkilökunnan työhön. Doctoral thesis. 2014. University of Turku, Department of Accounting and Finance. 341 p.
[33] Sliger M. Broderick S. The Software Project Manager's Bridge to Agility. US 2008, Pearson Education Inc. 353 p
[33] [assessed on 16.5.2014] http://agilemanifesto.org/
[34] Hock D. Birth of the Chaordic Age. First edition. San Francisco 2000, Berrett-Koehler Publisher. 345 p.
[35] Laurinolli H. Rahoituskilpa ei paranna yliopistojen tuloksellisuutta. Aikalainen 5(2014).
[36] Smith G., Sidky A. Becoming Agile in an Imperfect World. Greenwich CT 2009, Manning Publications Co. 380
[37] We’re In This For The Money. [accessed on 12.5.2014] http://geepawhill.org/?p=31
[38] Pentland A. The death of individuality. The New Scientist. 5 April 2014.
[39] Hamilo M. Reilu kapitalismi rikastuttaa. Tiede. 2(2014).
[40] Puttonen M. Moraali koulitaan yhä uudelleen. Tiede. 2(2014).

[41] Marshall B. Gateway Drug to Synergism. [accessed on 12.5.2014] http://flowchainsensei.wordpress.com/2012/09/06/gateway-drug-to-synergism/
[42] Laanti M. Agile Methods in Large-Scale Software Development Organizations. Doctoral thesis. Oulu 2012. University of Oulu, Faculty of Science, Department of Information Processing Science. 192 p.
[43] Highsmith J. [assessed on 14.5.2014] http://agilemanifesto.org/history.html
[44] Hellström R. Social Neuroscience of Agile Transformations - Team's Gain & Manager's Pain. Turku Agile Day, Turku 13.5.2014.
[45] [assessed on 16.5.2014] http://agilemanifesto.org/principles.html
[46] Tolle E. The Power of Now: A Guide to Spiritual Enlightenment. 2004, New World Library. 236 p.
[47] Puttonen M. Poliittinen asenne riippuu geeneistä. Tiede. 5(2014)

10 Appendices