EMPIRICAL STUDY OF PROJECT MANAGEMENT PRACTICES

Peter Cork

Master's thesis
November 2015
International Project Management

TAMPEREEN AMMATTIKORKEAKOULU
Tampere University of Applied Sciences

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Master's Degree, International Project Management

Peter Cork
Empirical study of project management practices

Master's thesis 94 pages
November 2015

The complexity of modern projects, especially in the software industry, require formalised tracking at the same time as allowing modern methodologies such as Agile methods to be used to ensure the right product is delivered.

This research investigated how projects are currently executed in five companies of varying size in Finland. The research was done to support the hypothesis that projects are often managed using more than one method to support varying needs of the organizations in which they are executed.

Nine project leaders were interviewed in five sample companies and data was gathered in the style of an 'Appreciative Inquiry' to support finding good working practices, not problems. The interviewees were asked to describe how projects are executed throughout the lifecycle, along with roles, responsibilities and constraints, and how the projects are supported by the organization. The interviewees were also asked to imagine an ideal working environment to gain ideas about how projects can be run in a more suitable way.

The research found that in smaller companies the projects tend to use very lightweight methods that are similar to Agile methodologies but often scaled down to suit. In the larger companies more formal planning such as Waterfall style road mapping is used, while the team attempt to execute in an Agile way.

The thesis gives the reader ideas about how projects are managed. It also gives insight from the interviewees on how they feel that projects could be executed more successfully.

Key words: Agile, appreciative inquiry, project management, Lean.

CONTENTS

ABBREVIATIONS AND TERMS

AI	Appreciative Inquiry / Appreciative Enquiry
a.k.a	Also Known As
API	Application Program Interface
BBC	British Broadcasting Corporation
CEO	Chief Executive Officer
DSDM	Dynamic Systems Development Method
FDD	Feature Driven Development
HBR	Harvard Business Review
HW	Hardware
Ibid.	Latin, short for ibidem, meaning ‘in the same place’
MVP	Minimum Viable Product
NDA	Non-Disclosure Agreement
OOPSLA	Object-Orientated Programming, Systems, Languages & Applications.
PMBOK	Project Management Body of Knowledge, a PMI publication.
PMI	Project Management Institute
PMO	Project Management Office
R&D	Research and Development
SW	Software
TAMK	Tampere University of Applied Sciences
TPS	Toyota Production System
WBS	Work Breakdown Structure
XP	Extreme Programming

1 INTRODUCTION

As software systems have become increasingly complex in the past few decades the need for formalized projects with a large number of team members has become a critical part of successfully delivering projects. Since the turn of the millennium, there has been new thinking and techniques about the best ways of managing software projects along with other methodologies taken from other sectors such as manufacturing. The most notable of these is Agile software development, which tries to tackle the unreliability of software development by traditional methods, such as the Waterfall method.

Software projects are also expensive in terms of the costs involved in engineering staff and ensuring that the quality meets the expectation of the customer. For this reason, many companies strive towards Lean thinking in order to reduce costs. Agile has been the answer to this dilemma, but still the boundaries need to be pushed further. Teams are adapting and new ways of thinking are arising, such as the #noestimates movement that promotes reducing project overheads to the next level.

Anybody leading a software development project knows that the reality of day-to-day work differs largely from textbook guidance of the methodologies. Project managers and team leaders typically do not – and should not – follow the process to the word and apply a very often a mix of techniques to ensure success. It is also the case that in many large companies, individual teams may use, for example, Agile methods but the company as a whole is not Agile. Companies still need to be able to plan and follow projects, and the default mode of operation is the use of more traditional methods that are easily understood. This need to plan and track, combined with the need to be Lean can force upon teams the need to mix practices to satisfy the local and wider needs within a company.

Software project management practices at least move very quickly and it is difficult to predict the future or indeed what the best practices are. This research project attempts to take critical look at what methodologies projects are currently using and also gain insight to how project managers envision an ideal project.

2 RESEARCH PLAN

2.1 Focus of the research

This topic was chosen in order to gain insight into how companies are confronting the need to reduce cost at the same time as ensuring customer satisfaction. As it was mentioned in the introduction, some companies may choose to use different methods at different levels of the organization and this research investigates the effects of those decisions.

The objective of this thesis is to research how companies in Finland actually manage software projects. It will attempt to gain insight whether methodologies are being used as guided or mixed in order to ensure success in the working environment of the team. The thesis aims to provide a good overview of the state of project management practices in the companies studied.

2.2 Concepts and theories

There are many theories about management from the traditional Waterfall models to modern Lean methods. There are movements that also promote the change of existing practices, for example, the *#noestimates* movement on Twitter.

Since the research will focus upon software project management in practice versus textbook theory, sources will be drawn upon that describe both scenarios. Texts books, of course, will provide the clean process and research papers and articles will form the basis of the methodologies in practice.

2.3 Research questions

This thesis aims to answer the key questions:

1. How projects are managed in a selection of Finnish companies?
2. Why such project management methodologies have been chosen?
3. What constraints are the projects subjected to by their organizations or customers?
4. What motivates project managers and what they see as the ideal environment to manage projects?

Question one will research what methodologies and practices are used in the project lifecycle by the companies in the research pool. It will look into the roles and responsibilities – along with the initiating, planning and execution phases. Lean methods will also be discussed to see if they are being applied effectively.

Question two will investigate how teams have chosen the methodology they use; i.e. is it freely chosen or forced upon them? It will also look at how the organization supports the projects and if it has any effect on the chosen methodology.

Question three will research what constraints are applied to projects to control the dimension of cost, time and scope. Also, it investigates if satisfying the customer constrains the project in any way and whether the teams co-located or virtual.

Question four will highlight what motivates the project managers in the research pool, gain some insight into their experiences of well managed projects, and then discuss the ideal project environment from the perspective of the interviewees.

2.4 Data and Methods

The data collection of this thesis is carried out by means of an Appreciative Inquiry (AI) – which will be described below – at several companies in Finland. The research pool is formed from large corporate software houses, with the inclusion of small and mid-sized professional services to ensure the diversity of responses. The above-mentioned selection of companies is intended to support the hypothesis that project management practices may be imposed by organizational or customer constraints. Also, included is a small independent gaming company that develops games and provides media production services to other companies. This company was included to bring to the study a point of what practices are used when the team is completely self-organizing. That is not having little or no organizational or customer constraints. The companies included in the inquiry will not be identified to ensure that this thesis remains in the public domain.

The research data was collected by conducting nine interviews, within five companies, which were conducted as a relaxed conversation rather than a strict question and answer scenario. The first interview was a conducted as a pilot to test how questions would work

during the conversation. As a result of the pilot, the base questions were not altered but, in later discussions, the participants were asked not to go into so much detail in some aspects. All of the interviews were transcribed into text to ease the analyses of the data. The details of the data collection will be described in detail in chapter four, where the companies and participants are described along with more details of the interviews.

2.4.1 Appreciative Inquiry (AI)

There are many ways that an inquiry of this kind can be conducted such a problem identification or gap analyses, but these can easily lead to a negative spiral of comments that do not give a meaningful insight into practices. AI works from a different perspective that within people, teams and organizations exists great experiences and practices, which can be built upon (Carter 2005). Therefore, AI is based on a positive approach that focuses on strengths rather than problems (Ibid.). AI is essentially a tool for change, but since changing any of the organizations is out of the scope of this thesis the process will be cut short as necessary. The transcriptions of interviews have been made available to the respective interviewees if they wish to follow the process through. Berrisford's (2005) article about using AI at the BBC discussed how using the methods led to 15 000 unique ideas and 35 concrete initiatives of change from some 10 000 employees.

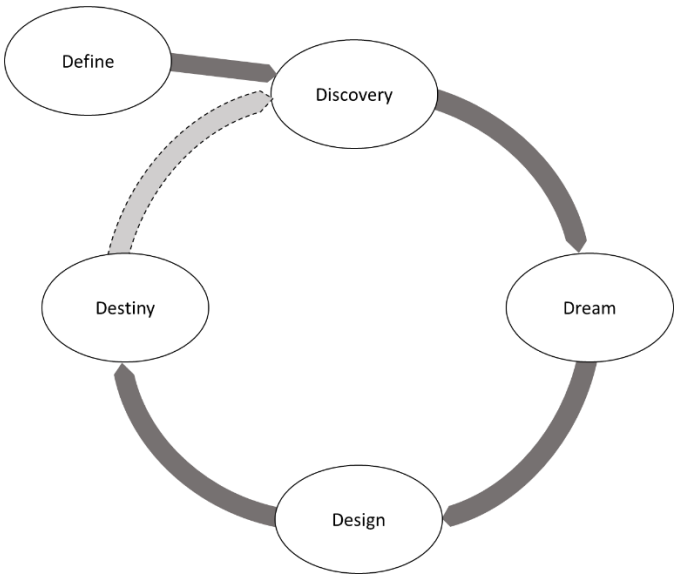


FIGURE 1. Appreciative Inquiry process

The Appreciative Inquiry starts with the Define phase, as shown in Figure 1. This is considered a key component of the process that should be done in an open-minded manner (Lewis, Passmore and Cantore 2008). This is the focus of the inquiry and in the case of this thesis the topic is predefined to focus on project management practices at the respective companies (Berrisford 2005).

The Discovery phase focuses on discovering the key strengths and uncovering the 'best of what is' (Berrisford, 2005; Lewis et al. 2008, p. 49) In this phase data is captured through the use of interviews and then mapping elements that emerge into common themes (Lewis et al. 2008, 49). The interview questions should be framed in a positive way. The aim is to draw out stories and experienced about the organization and the person at their best (Lewis et al. 2008, 52). Asking the participants to describe an event that went very well could draw out such stories.

In the Dream phase the participants imagine 'what could be' in the future (Berrisford 2005). The phase is not about unbounded thinking and should remain grounded in the organizations best practices (Lewis et al. 2008, 55). The phase seeks to create a positive future based on the discovery of past successes (Ibid.).

The Design phase involves agreeing a common desired future taken from the themes in the dream phase (Lewis et al. 2008, 58). Participants imagine and create ways in which the dreams can be brought into being (Berrisford 2005).

The Destiny phase is essentially creating an action plan. It forms the next concrete steps that should take place, in order to bring the agreed dreams to realization (Lewis et al. 2008). People should be made accountable for enacting the changes and regular communication meetings should be planned.

After the Destiny phase, the process can be optionally started again if desired to create a continuous improvement process. Care should be taken not to constantly load employees with change, which can lead to overload and loss of productivity as described in Bruch & Meges' article 'The Acceleration Trap' (2010).

2.5 Content of the research

Chapter 3 will describe different project management methodologies to form an overview of the different techniques and characteristics. It will also discuss the circumstances under which the method is best utilized. The focus is mainly on Agile methodologies but Waterfall and Lean are introduced to facilitate discussion.

Chapter 4 will introduce the data collection process and from whom the data was drawn. Company's names will be coded to avoid discussions of privacy but some data about the business of the companies will be revealed, for example, device, gaming, consulting, etc. It will also discuss the interview process in more detail.

Chapter 5 will discuss the research questions using the data collected in the interviews. Each meta-theme and theme found from the data will be discussed using the answers from the interviews as background.

Chapter 6 will again revisit the research questions and make conclusions about the data found and where applicable, it is compared to the methodologies introduced in chapter 3. Items for further research are also included within the conclusion of some of the themes.

Chapter 7 closes the thesis by reflecting on the research and the process used. It discussed what went well and what problems arose as well as the reliability and validity of the research.

3 PROJECT CONTROL

All projects are unique, thus producing a unique deliverable (Brewer and Dittman 2013). Projects are complex because their activities are not typically predictable, nor can they be repeated (Karlos, Martinsuo and Kujala 2011, 18). Therefore, these projects need some kind of means to control and measure the outcome of the project. Since all projects are temporary, they must have a beginning and an end, and if no control method is utilized, it would be very difficult to tell what the projects goals are and when those goals have been met.

3.1 Project environments

Brewer and Dittman (2013) suggest that using a systems approach allows projects to be viewed in the context of the entire environment. This can be taken to mean that although projects are temporary entities to carry out a specific change, the practices and procedures used are usually the result of the environment in which the project is being executed.

The hypothesis is that the environment is the most the most common reason to the second research question – why a certain methodology was chosen – not the needs of the actual project in question. Companies, especially corporations, impose project processes defined by the Program Management Office (PMO). In consulting companies the customer may have defined which process or methodology is used in an attempt at having some level of control or a follow-up tool to ensure that their money is well spent.

As Brewer and Dittman (2013) state in their systems approach, the boundary surrounds the project and separates it from its environment. The environment is everything outside the project such as the organization, company, country, etc.

3.2 Project constraints

According to Brewer and Dittman (2013) projects are subject to a triple constraint of scope, time and cost.

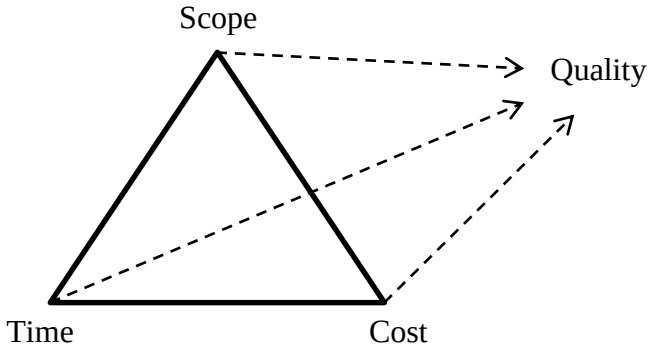


FIGURE 2. Project constraints

Scope is typically the content of the project that is may or may not be predefined. In classical project environments, the project requirements are specified at the beginning of the project, leading to features and tasks in a work breakdown structure. In more modern project types such as Agile the scope is defined on a higher level, such as a vision, which is allowed to change during the project according to customer feedback. It could therefore be said that the scope is the minimum system that satisfies the customer's needs.

The Time constraint can come in several forms. The most common is a deadline, which is the date by which the project must be finished. It can also be the maximum amount of time that can be used, for example, the maximum number of hours or weeks. Typically, man-days or man-weeks are commonly used as a unit of measurement.

The Cost constraint is the amount of money that is used to fulfil the project requirements, but it can also refer to resources which translate to money, such as people, office space, etc.

The three mentioned constraints need to be kept in balance to avoid never-ending projects. According to Yourdon (1997), projects – especially in the IT industry – turn in to 'Death March' Projects that never end. Figure 2 shows the three constraints bounded by a triangle because a change in one of the constraints will affect the other two in some way. For example, if you increase the scope it will directly affect the time and cost constraints. Typically the fourth dimension of quality is usually mentioned in projects. It is usually good for the customer to define the quality required so that the project does not endeavour to build a Rolls Royce level of quality when a Ford would suffice or vice versa. Quality is linked to the bounds of the triangle in the sense that a change in quality can easily affect

the other three constraints. For example, increasing the quality level usually creates new requirements that increase scope, cost and time.

3.3 Project lifecycle

On a general level the project lifecycle can be broken down into three stages (Karlos et al. 2011, 35). This thesis is mainly concerned with the project execution stage. How this stage is executed depends on the methodology used by the team. The following sections will discuss a few of the most common methodologies.

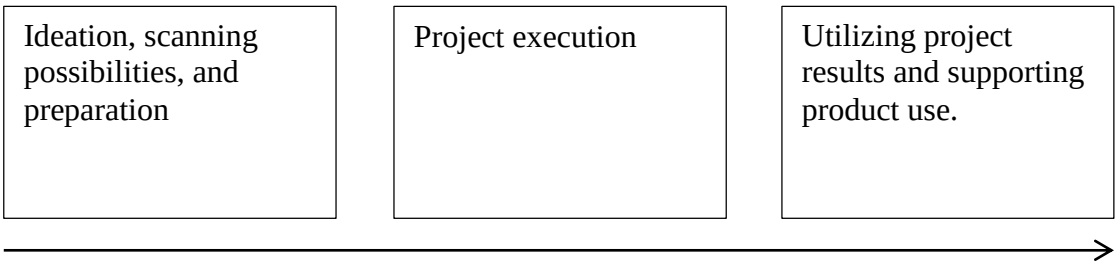


FIGURE 3. General project lifecycle (Karlos et al. 2011, modified)

3.4 The Waterfall model

The Waterfall model is the most well-known method of managing a project that fits well to most project types. It was taken from the engineering community and adapted to the software industry. It is thought that its first introduction to use in software was given in the paper titled ‘Managing the development of large software systems’ (Royce 1970).

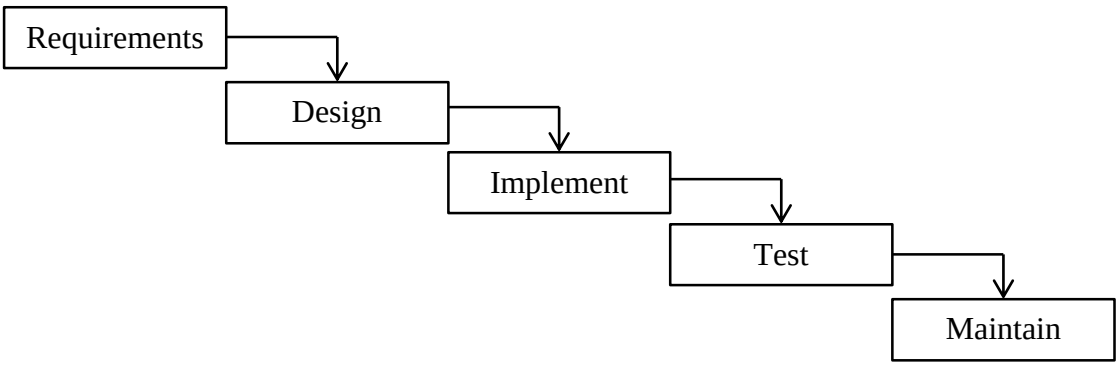
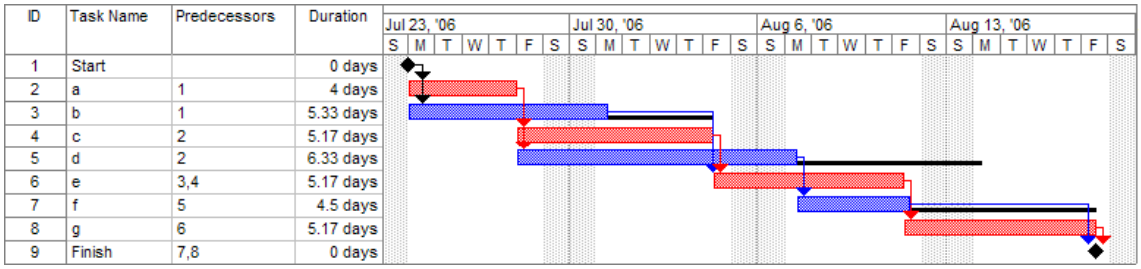


FIGURE 4. Example of a Waterfall process

The stages in the project are sequential and linear and it requires that the system requirements are understood well before the design and implantation stages. It also enables project managers to track progress and identify slippages early (Davis, Bersoff and Comer 1988).

Tasks within in each stage of the project often follow the same model and usually are manifested in Gantt charts that more accurately define when and who will perform a task, along with its dependencies.



ID	Task Name	Predecessors	Duration
1	Start		0 days
2	a	1	4 days
3	b	1	5.33 days
4	c	2	5.17 days
5	d	2	6.33 days
6	e	3,4	5.17 days
7	f	5	4.5 days
8	g	6	5.17 days
9	Finish	7,8	0 days

FIGURE 5. Example Gantt chart. Source: (wikipedia 2015)

The Waterfall model is often criticized by the software development community for its lack of flexibility. The key drawback is the belief that it is not possible to go back to a previous stage and re-engineer when problems are identified. The model gets its name because once water goes over the falls it cannot go back (Brewer and Dittman 2013). It is, however, heavily used in other industries such the construction industry because stages of the project often depend on the completion of previous stages in the project and it is matches the environment well.

The Waterfall model is rarely used directly in software engineering anymore but the author's hypothesis is that the model does actually manifest itself in many forms throughout the software industry either deliberately or accidentally. For example, the Incremental development model is a series of mini Waterfalls that happen within the time boxes of the increments but is more relaxed in the sense that redesign is allowed at the beginning of the next increment if needed. In Scrum, introduced in section 3.6.2, teams can easily fall into a pseudo Waterfall model when high-level Sprint planning is carried out. The team might decide to do a requirements clarification Sprint followed by an architectural design Sprint and then move onto development. This can start to look a lot like Waterfall implementation. This could be a sign that Scrum is not the appropriate model for the project in hand.

Although the Waterfall model is not seen by the software community as effective anymore, it does still have its place under certain circumstances. It can be used if the project must adhere to a strict, well-documented specification. Assuming that the above conditions are met, it is also appropriate when the customer is not available to give regular feedback, or when members of the team are separated geographically and cannot communicate effectively (Brewer and Dittman 2013). The software system is large and complex and requires multiple teams to work independently (Ibid.).

Using the Waterfall method

The most appropriate time to use the Waterfall model is when the requirements are well predefined but it should be noted that the methodology does not tolerate changes in requirements well. Waterfall is usually well understood by all levels of personnel. It is also practical to use when the team members are geographically separated and there is little regular contact with the customers to gather feedback about the current solution.

Waterfall's greatest weakness is that it can be very difficult to go back and refactors or rework finished work that no longer suits needs (Brewer and Dittman 2013). The main criticism of Waterfall in the software industry is that history has told that customers do not know what they really want up front and often change requirements when they see the product in action. The method also assumes that work division between the team personnel is clear with separate people for each role. This is nowadays seen as not efficient, especially in small companies where people contribute multiple skills. (http://www.techrepublic.com 2006)

3.5 Lean

Lean is not a software delivery methodology but originates from manufacturing and is based on the Toyota Production System (TPS). Taiichi Ohno Developed the TPS between the mid-1940s and mid-1970s because he recognized that it was inefficiency and waste that was the key reason that Toyota's car production was lower than that of competitors in Detroit, in the USA (The Economist 2009). Ohno wrote several books of which the most well-known is, 'Toyota Production System: Beyond Large Scale Production'. Although the concepts of Lean in production are comparatively old, it has not been until

recent times that the techniques have transferred to other industries. It could be said that the principles of Lean and the TPS are found throughout Agile methodologies and probably Kanban and possibly XP are the closest match to the TPS in terms of team behaviours and practices.

Nowadays, many start-up companies use the principles of Lean because the principles of well-defined business plans, solid strategy and thorough market research no longer necessarily work in the fast-paced and changing world (Reis 2011). The concepts of both TPS and 'The Lean Start-up' are useful in the software industry to help define product content and ensure fast paced development.

3.5.1 Toyota Production System

In the Art of Lean Inc's, TPS handbook (2006, 5) Just-in-Time and Built in Quality (Jidoka) form the two pillars that support the goals of producing highest quality, lowest cost and shortest lead time.

Just-in-Time is defined as producing and delivering the right parts, in the right amount at the right time using minimum resources (Art of lean Inc. 2006, 6). In software development this forms the base of XP's later discussed principle of simplicity that nothing more than what is needed is developed.

Jidoka is the concept that humans and machines can detect faults, abnormal conditions and prevent those from being passed onto the next stage in production (Art of lean Inc. 2006, 17).

One of the interesting and most overlooked results from the TPS is that of Total Efficiency. To improve efficiency, a holistic view must be taken across the whole system. The TPS handbook (Art of lean Inc. 2006, 15) states that managers tend to think of improvements in efficiency and quality only in their own responsibilities. They must, however, consider how those changes might affect the whole operation.

TPS defines seven types of waste which can be found in any industry from large-scale manufacturing to running a small retail space. Eliminating waste in its many forms known a Muda in Japanese is the driving force of TPS. 'Waste encompasses all factors that do

not add value to the product or service, whether in parts, labour or production process' (Art of lean Inc. 2006, 9). Below is a hypothesis of how the seven forms of waste can manifest themselves in a software project.

Defects and correction: The consequences of software defects are well known by anybody involved in software production. When software defects occur, it creates waste by creating rework, extra labour and costs in terms of development and testing. Defective code may need to be thrown away and, at worst, it can mean delivering defective product to the customer causing rework and costs on their behalf too.

Over production: This can happen when requirements are not well thought through and are subsequently produced by the team and thrown away when they are found not to be useful. This is one area that Just-In-Time is valuable. Requirements should only be delivered to the customer when they actually need them and by doing them as late as possible is a way of ensuring that the requirements best meets the customer's needs. To put that another way, the later the requirement is delivered the more likely it is that the customer will know what is actually needed.

Waiting: In software projects, time spent waiting by engineers is very expensive. Software dependencies often leave people waiting for a release to happen or for a manual task to take place. This can easily happen if people are pre-assigned tasks and have no flexibility to start something else if it is not possible. Automated builds and continuous integration can eliminate this waste by creating the continuous flow that Just-in-time-promotes.

Conveyance: In TPS this means poor layouts of workspaces and materials and people. In software engineering this manifests as teams that are not co-located for geographical or political reasons. An example of this might be cross-functional personnel such as graphic designers and test personnel that get held in their own departments. Even if they are in the same building they still produces a lot of waste because of poor communication and unnecessary meetings that could have been avoided.

Motion: Like conveyance, if the team are not co-located it will lead to unnecessary walking or worse, travel, which usually leads to waste of both the company's and people's personal time.

Processing: Processing or, to be more precise, over-processing easily happens – especially when the quality level is ill defined. Engineers may spend time trying to handle every exception, where a simpler catch all framework would suffice. Automating tests is important with respect to Jidoka but spending a week automating a test that would only use one day of manual testing throughout the project is clear waste. In project management, especially in large corporations, time is often spent producing reports and filling out checklists that add no benefit to information sharing or the development of the product.

Inventory: In TPS this means having material on hand that will not immediately be used in the production process. In software engineering this could be detailed specifications produced for work that will not be developed for weeks or months. It can also easily arise when software developers try to 'future proof', by creating software modules or APIs that never actually get used.

3.5.2 The Lean start-up

Using conventional wisdom, the first thing that any business must do is create a business plan that describes a problem, the opportunity and includes a several year plan of profits and cash flow (Blank 2013). Uncertainty makes it harder and harder to predict the future: planning and forecasting only works well within stable environments with an operating history (Reis 2011). The facts compounded by firms – especially technology firms – disrupting traditional business models by using unforeseen business models as described in the article 'Big Bang Disruption' (Downed and Nunes 2013) makes starting a business in the modern world a fail prone endeavour. This fact is true in developing software projects and in many cases a software project such as a game or service is the actual business model.

Research has found that 75% of all start-ups fail. Instead of coming up with a product and developing for a long time, fail fast and continue learning (Blank 2013). It has now been learned that business plans rarely survive the first contact with a customer and start-up that succeed move quickly from failure to failure (Ibid.). Using these finding in software development it should go without saying that the faster you deliver something to the customer and get feedback, the lower the chances of project failure.

Lean start-ups typically work with a Business Model or Lean Canvas. It is a single poster in which the nine building block of the business and the hypotheses that need to be tested. The core of the Business Model Canvas is the value proposition. That is the value a customer will gain from the product. The time-tested method of finding out a customer's needs is to ask them either directly or by providing a prototype quickly and finding out by empirical means – a process known as early validation. The other aspects of the canvas are not in the scope of this thesis but it would be wise to understand in terms of recognizing if the investment is worthwhile, and what partners and people you need in order to succeed.

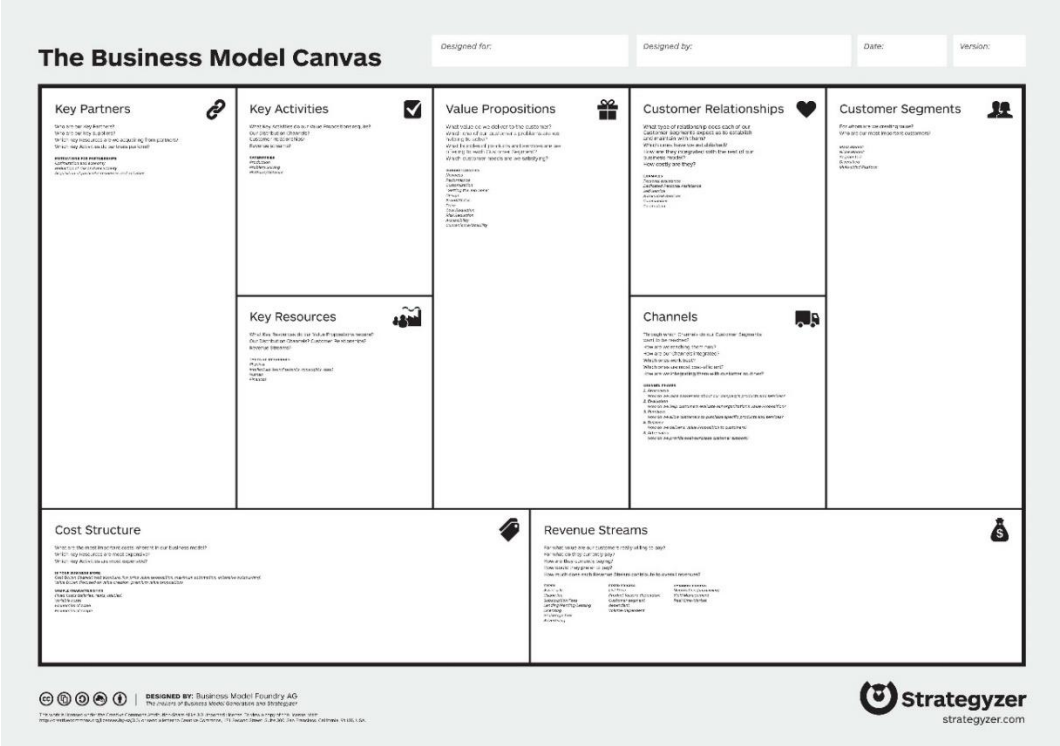


FIGURE 6. Business Model Canvas. Source: (businessmodelgeneration.com 2015)

Using Lean in software development

Lean is not a methodology but a business minded approach that can be used when tackling software projects and can be applied no matter what methodology a team uses. Any software that is developed should solve a problem for a customer or improve a process or activity in some way. Using the combination of the Lean start-up to understand the business case for the project, and a Lean mind-set to eliminate unnecessary tasks from

the project form an excellent foundation on which to build useful software with minimal cost.

The Build-Measure-Learn feedback loop is the core of Lean start-ups (Reis 2011). It enables the fast movement from ideas to validation and iteration of the product to best meet the customer's needs.

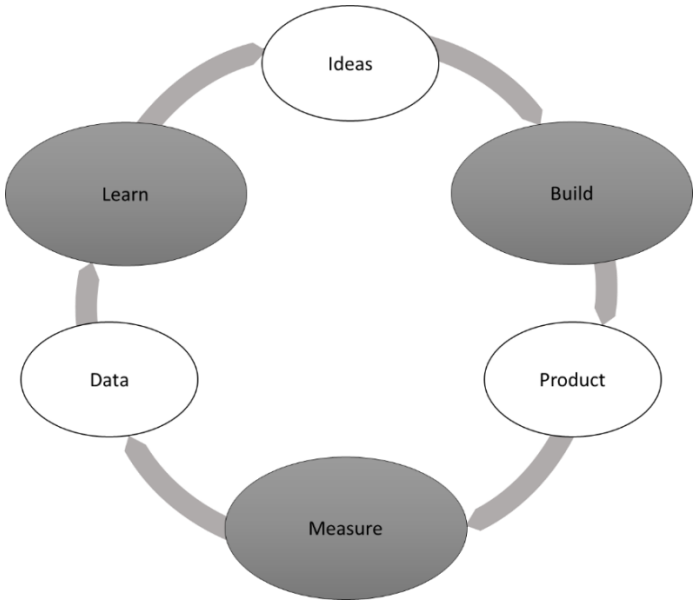


FIGURE 7. The Lean Build-Measure-Learn loop (Reis 2011)

Interpreted from The Lean Start-up (Reis 2011). Starting with ideas the Build step should be entered quickly to build a Minimum Viable Product (MVP). This MVP is the first version that will enable a full cycle of the Build-Measure-Learn loop by being at a level that can be put in front of customers using the least amount of time. In the Measurement phase, the progress should be determined using quantitative methods to discover if the customer actually wants the product or how they would like it improved. The data collected will give input to the Learn phase, which will allow the team to decide what comes next. The Learn phase also gives a good opportunity for a milestone to assess progress. Then comes the 'Pivot' upon completion where the hypothesis is assessed and if proven false, a change is needed. If the hypothesis proves true the next iteration can begin with ideas to enter the loop again.

In Kinnunen's Thesis (2014) the use of Lean practices were used to deliver a web application for advertising real estate. Early validation was gained from the use of bloggers as early adaptors of the product. The team found that ideas had to be refined

many times as they got feedback. The biggest challenges were following practices in the long term and fear of the Pivot because sometime a hypothesis had to be abandoned. It was also recommended that the team choose a language that they are comfortable with to avoid technical learning on top of process learning.

3.6 Agile

Agile is a set of values and principles and there is no such thing as an Agile methodology (Wells 2009). There are methodologies and practices that are used by Agile teams. The principles are the result or collaboration and endorsement of people in the software industry and form the foundation of all methodologies used in relation to Agile. There are twelve principles that have been set out by the Agile community that are listed below (agilemanifesto.org 2015).

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

The following sections describe the various methodologies and practices that are often found under the banner of Agile. There are many overlapping activities and teams often combine practices to ensure the best outcome in projects. This thesis will focus mainly on Agile methodologies because they are the most commonly practiced methods in the software industry.

3.6.1 Extreme Programming (XP)

XP is a set of practices that focus on close teamwork and customer satisfaction (Moreira 2013). It was first described by Kent Beck in his 1996 book 'Extreme Programming Explained'. The method is successful because it concentrates on delivering the highest value features to the customer as soon as possible and embracing change in the requirements as the project proceeds. XP is an iterative process where feedback is used as input to the next iteration. XP has four recognized values (Wells 2013) that are stated below. The explanations are interpretations of the values.

Simplicity: Many software projects try to design ahead and the developers are tempted to code into the systems functionality that is for future use. XP steps away from this practice by only developing software that completes a single user story. 'It may never be needed' is the mantra used for future requirements. In summary, developers should only do what is needed, and absolutely no more!

Communication: The team are co-located and talk face-to-face every day in stand up events. The point of standing up is to ensure that the team meetings are kept short and to the point. Team competence is the key instead of individual competence and everybody is encouraged to help solve each other challenges. One easily recognizable aspect of XP is pair programming, where two developers sit at the same computer and write code together. This also has the benefit that all code is peer reviewed by at least one other person.

Feedback: The team actively present working software at the end of each increment to the customer. During this process the developers are expected to be open to feedback and the changes required to improve both the product and the team behaviour.

Courage: The team should have courage because they solve problems together. It also means that they should have the courage to say when things are not right and tell the truth about the system. Also, the courage to accept that a piece of code or a whole increment has not worked as planned, and be willing to throw it away. This follows the principle of the 'Pivot' in Lean thinking.

In pure XP teams there is typically just two roles, the customer and the developer. The project members are self-organizing and it is the customer who drives the project, by setting goals and providing the requirements (Moreira 2013). Like most Agile methods, user stories, which are described in the section about Scrum, instead of formal requirements form a high-level specification of what the system must do from a user point of view.

Although not strictly adhered to, XP is a test driven method, meaning that unit tests should be written before the production code. The testing rules outlined by Wells in his XP programming rules web page (2013), for example, all code must have Unit tests and pass those before releasing, is a must. Writing the test beforehand may help with the design but probably does not affect the final quality significantly.

Using XP methodology

The key strengths of XP are that progress can be made in the project even when requirements are not well defined and that the customer can see the results immediately and give feedback. Project planning is easier to manage. It must be possible to create automated unit and functional tests. The team must be able to communicate well which makes it unsuitable for large or teams that are not co-located. Typically, most of the developers need to be experienced and XP is not suitable for mission or life-critical systems (Wells 1999; Brewer and Dittman 2013) XP is also not seen as effective in middleware and device driver projects or the maintenance of legacy systems since changes are not visible to customers and the number of automated tests needed will far outweigh the benefit (DiFalco 2014).

3.6.2 Scrum

In the HBR article, ‘The new product development game’ by Takeuchi and Nonaka (1986) it is suggested that a Rugby approach should be used in product development. The suggestion a group of engineers could move away from highly structured stages and design the product before all the feasibility studies are completed. Team play is the key to success, where the members work together from start to finish and engage in iterative experimentation. The article led to the presentation of the Scrum theory by Ken Schwaber and Jeff Sutherland at the OOPSLA conference in 1995. Schwaber and Sutherland are considered to be the godfathers of Scrum and have written several books and articles on the subject.

Schwaber’s original paper (1995) states that Scrum is an empirical approach that assumes that the analyses, design and development processes in the Sprint are unpredictable and need to be flexible to change. The planning and closure have well-defined inputs and outputs that sandwich the Sprint (development) phases, which are nonlinear and flexible. Since then, Schwaber and Sutherland have actively maintained ‘The Scrum Guide’ (2013), which is drawn upon below to describe the modern process.

The Scrum team

The Scrum team is made up of a Product Owner, Scrum Master and a development team that is self-organizing. A Scrum team is usually made up of experienced personnel due to the chaotic nature of the method, but it is usual to have a few junior members who are guided the more senior members. It does not usually work well without experienced developers or large teams (Brewer and Dittman 2013).

The Product Owner has several responsibilities in the team, such as maintaining a Backlog of user stories (requirements) and ensuring that the teams understand them. When a user story is complete the Product Owner is also the sole person who can accept that a user story was implemented as expected. In most modern teams a mix of XP and Scrum is very common and the practice of having the customer act as the product owner renders the best outcome.

The Scrum Master is literally a referee that ensures that the team adheres to Scrum practices and rules. The role is usually misunderstood to be that of a project manager but the role is more like a servant of the team whose job it is to provide a good working environment. In Scrum theory the role of the Scrum Master can even be rotated in round robin fashion at Sprint boundaries, but it is the author's hypothesis that the Scrum Master is usually a more senior member of the organization leading to team members being managed by the Scrum Master. This is usually caused by the fact that projects still need to service organizational demands such as reporting, risk management and financial planning, which are the tasks of a project manager. Both roles are typically carried by a single person.

The development team is made up of a small number of cross-functional personnel that have all the required skills to develop the product increment. As mentioned in XP, they should be self-organizing and nobody tell the team how they carry out the work. The team as a whole is accountable for the results of the increment and it must be a single team. Multiple sub-teams or teams that are spread across different geographical locations are not effective.

Scrum artefacts

There are several artefacts used in Scrum to promote clarity to both the team and the stakeholders as well as used to track progress and achievements. They form the sparse documentation and deliverables that is usually found in Agile and are mainly used to guide and track the team's output.

User stories describe one piece of functionality that a developer will produce as part of the overall product. A user story must be possible to complete during one Sprint. It should be written from the end user point of view by the product owner in terms of the functionality that it provides. An example might be 'As a user I want a button that will start the process of sharing my photo in the internet'. Note that text did not imply that the sharing must be completed feature but some basic part of that feature because that might not be possible to implement the whole feature in one Sprint. Breaking down the stories to as small pieces as possible is important. Related user stories can be grouped together in a theme to make up the required functionality.

The most important is the product Backlog, which at the start is essentially a list of user stories that are known to be needed to complete the product. As the project proceeds the Backlog evolves according to current known facts. Some stories become obsolete and others are added as needed. It is very common for architectural spikes or defects that were not identified in Sprints to be added by the team. As mentioned, Scrum is an empirical process and the observations and experience of the team feed back into the Backlog to ensure that the best possible product is created. The product owner maintains and prioritizes items in the Backlog. Transparency is provided to the team and stakeholders by having the highest priority features at the top of the list and the team usually ignore items lower in the list due to volatile nature of the list.

The Sprint Backlog is created during Sprint planning and is the items that the team have taken from the Backlog and committed to develop during the Sprint. This list is created in cooperation between the product owner and the development team. It is essential that the items in this list are achievable and measurable.

The ‘Definition of Done’ (DOD) is written document that describes what development, testing documentation and other artefact must exist before the user story can be said to be done. It is a contract between all members of the Scrum team describing the level of quality that is to be attained. The principle of Scrum is that potentially shippable software is produced, therefore the DOD must strive leave nothing undone that anybody needs to return to. As teams gain more experience the DOD should expand to attain higher quality criteria (Schwaber and Sutherland 2013, 16).

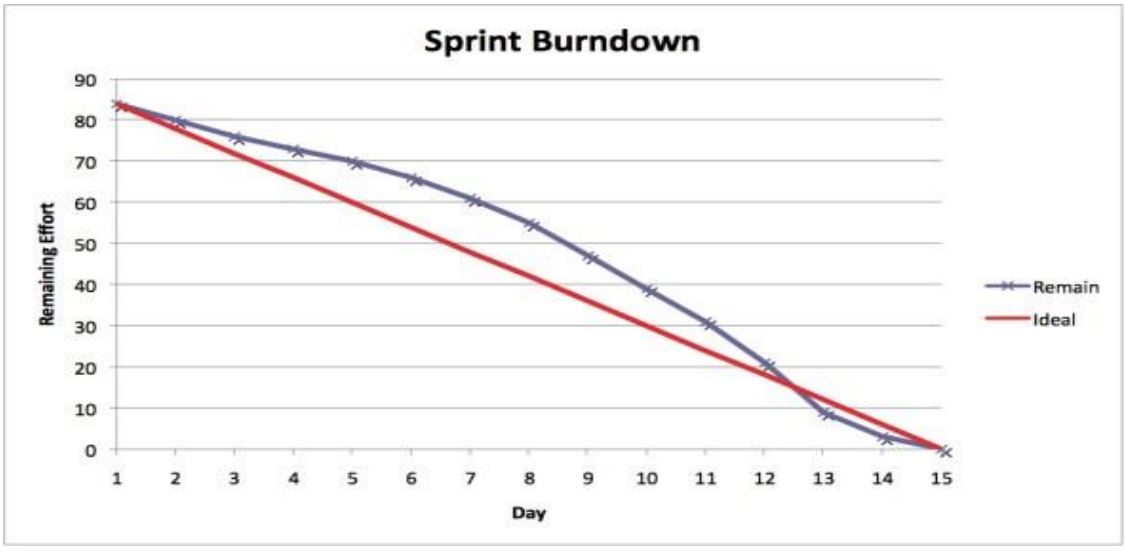


FIGURE 8. Example of a Sprint burn down chart. Source (Scrum-institute 2015a)

In Scrum there are several types or burn-down and burn-up charts that visualize the team's progress. The most common is the Sprint burn-down that is shown in Figure 8, which indicated the effort remaining for the Sprint Backlog. The red line shows the estimated burn-down at the beginning of the Sprint and assumes that effort in the Sprint is linear. The blue indicates the actual progress based on input given by the developers on a daily basis. This data is usually collected by asking the team members for each user story that is in progress an estimate of the remaining effort. This chart helps the team to understand if they are behind or ahead of schedule during the Sprint. Backlog burn-down charts exist but, as discussed earlier, Backlogs are highly volatile and tracking the Backlog the burn-down is misleading to and may lead to false conclusions. In the cases where the Backlog is stable it might be valid to follow but if the Backlog is stable it could be argued that the project is actually Waterfall development, not Agile.

Scrum events

There are several events or team meetings that place in Scrum. These include story sizing also known as Planning Poker[1], The Sprints, Sprint Planning, Daily Scrum (also known as Daily Stand-up), Sprint Review and the Retrospective meeting.

Story sizing is not part of the Sprint regime and the team may handle this as it suits best. Story sizing is where items in the Backlog are given a relative size. The estimates are not time based and are freely chosen by the team. It can be any arbitrary value such as the number of cups of coffee the developers will need to drink to get the story completed. Usually, the sizes can be numeric [1 to 10], T-shirt sizes [XS, S, M, L, XL...] or based on the Fibonacci sequence [1, 2, 3, 5, 8, 13 …] (Scrum-institute 2015b). In this event, a user story is presented by the Product Owner and each developer thinks how much effort it would be – often compared to a baseline story – and all reveal the estimates at the same time. If some have wildly different estimates, they explain to the rest of the team why in an effort to build clarity to the rest of the team. The estimation process is iterated until all are in agreement and a consensus is found. The game is continued until relevant user stories are estimated, usually the stories expected in the coming Sprints. The relative sizes are used to aid the Product Owner in prioritization by understanding efforts. The figures are also used to measure the velocity of the team output in average story points per Sprint.

[1] Planning Poker® is a registered trademark of Mountain Goat Software.

This provides empirical data to estimate the number of Sprints it might take to deliver the minimum viable product.

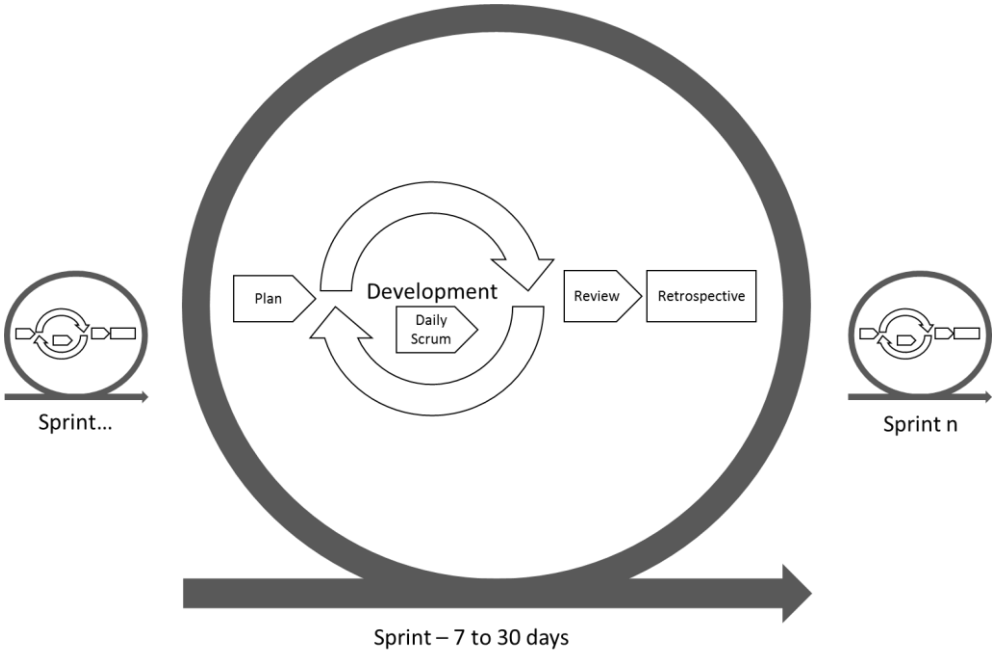


FIGURE 9. Events in a Sprint.

The Sprint itself is a time period in which development will take place undisturbed. It is an iteration within the project execution phase. It is recommended to last anywhere from 7 to 30 days although the team is free to choose. There can be any number of Sprints in a project but it is recommended to keep them the same length. Figure 9 illustrates the events in a Sprint, which are described below.

The Sprint-planning meeting is where the team plan the next iteration in detail. During this event the whole team plan the content of the Sprint, that is the stories that will be moved from the Backlog to the Sprint Backlog and then the tasks that are needed to complete those stories are also created, analysed and given an estimation of effort in hours. This is one of the cornerstones of Scrum that actual effort is only ever estimated for the work that will be done within the iteration. The Scrum board is created in the meeting to facilitate the daily stand-up meetings. The board is visual and normally made with sticky notes. Figure 10 shows an example board where the notes are moved to a status column as the Sprint progresses. If all goes well, all the notes should be in the ‘Done’ column at the end of the Sprint. Electronic boards can be used which enable more accurate burn down charts and other statistics if needed.

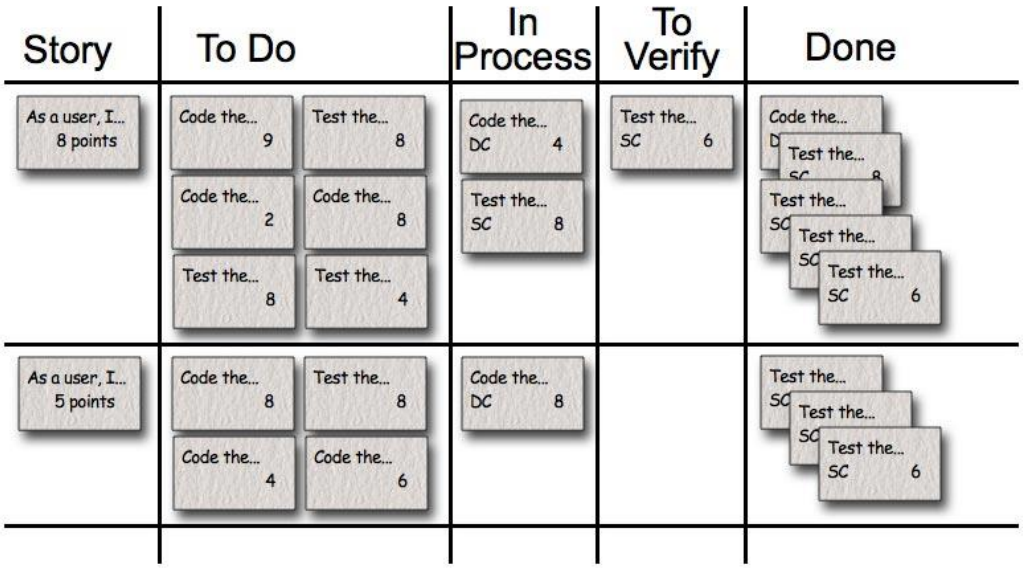


FIGURE 10. Mock Scrum board. Source: (Mountain Goat Software 2015)

The roles in this event are slightly adversary in the sense that the Product Owner should push to get as much content into the Sprint to maximise value and the Scrum master should push back to ensure that the team are not overloaded or having unreasonable demands placed upon them. The output is the plan for the next iteration. It is customary that during the Sprint the team are left to organize themselves and should not be disturbed by having their work changed in anyway. It is, however, a reality in most companies that those employees are expected to carry out company tasks such as attend departmental meetings and this should be taken into account by the Scrum Master when content is added to the Sprint Backlog.

The development phase is marked by a daily Scrum or Stand-up meeting. It is traditional that the meeting is held at the same time everyday within the team's working environment. This facilitates that the team are all informed about the activities that are happening and most importantly allows the team to solve problems together. The Scrum board is updated here so that the team can clearly see the progress of tasks in the Sprint. During the meeting all team members answer the following questions.

- What has been accomplished since the last daily Scrum meeting?
- What is he/she plans to accomplish until the next Scrum meeting?
- Are there any impediments that are preventing tasks from being completed?

(Scrum Institute 2015c)

Like the stand-up meeting in XP, the idea is for it to be short and informal and should not last more than 15 minutes. Bigger issues should be noted by the Scrum Master and managed outside the daily Stand-up as appropriate. In practice, it is very common for the bigger issues are handled right after the daily meeting with the parties concerned, freeing the others.

The Sprint review is held at the end of the Sprint and in this meeting the team demonstrate what user stories have been completed according to the definition of done and note what has not been completed with respect to the Sprint plan. (Schwaber and Sutherland 2013, 11; Scrum Institute 2015d). It is in this meeting that the Product Owner accepts of rejects the implementation of user stories and, if possible, it is advisable to have the customer present in this meeting if that is a different person from the Product Owner. The team should use the meeting to focus on their accomplishments and reflect upon the technical problems that occurred and how they were solved (Schwaber and Sutherland 2013, 11). As part of the empirical process, the team should review how the product sits in the marketplace or working environment and what would be valuable to do next, reviewing the current Backlog accordingly (Ibid.). There is usually a temptation to merge the Sprint review and Retrospective into one meeting (Radigan 2015). The Sprint review should be reserved for celebrating the success of the team and raising morale.

Whereas the Sprint review focuses on the work carried out during the Sprint, the Retrospective focuses on the team itself, and is considered a continuous improvement process. The main purpose is to focus on how the Sprint with respect to people, relationships, processes and tools. It should analyse what went well and what could be done differently. The output of the meeting should be a plan for implementing the improvements that the team should carry out (Schwaber and Sutherland 2013, 11).

Using Scrum

Scrum is quite similar to XP with respect to strengths, such as the project progressing with unstable requirements and the customer see results quickly and can give feedback. Good team communication is key, which can make it unsuitable for large or separated teams (Brewer and Dittman 2013). Scrum does support the concept of Scrum of Scrums as it scales to allow multiple teams to work on single project. Scrum requires hands on management and good monitoring in both quantitative and qualitative dimensions (Ibid.).

Scrum should not be used where the team cannot be left to complete a Sprints worth of work without changes. Changing content mid-Sprint should only be done under extreme circumstances. Scrum is also unsuitable when fixed deadlines are in place since it can lead to Waterfall style planning.

3.6.3 Kanban

The roots of Kanban are also in the Toyota Production System as a way of creating demand through the supply chain – the so-called pull system in which finished goods are replenished as they are sold to the customer (Art of lean Inc. 2006, 23). Kanban can be described as a card that passes between processes, communicating what materials to replenish (Ibid.). It is this card-passing concept that forms the way that Kanban is utilized in software.

Kanban is a workload methodology that aims to limit work in progress to what the team is capable of delivering (Cooke 2012). The key difference between Kanban and Scrum is that Scrum allocates work in a time-boxed Sprint whereas Kanban allows work to enter the process continually.

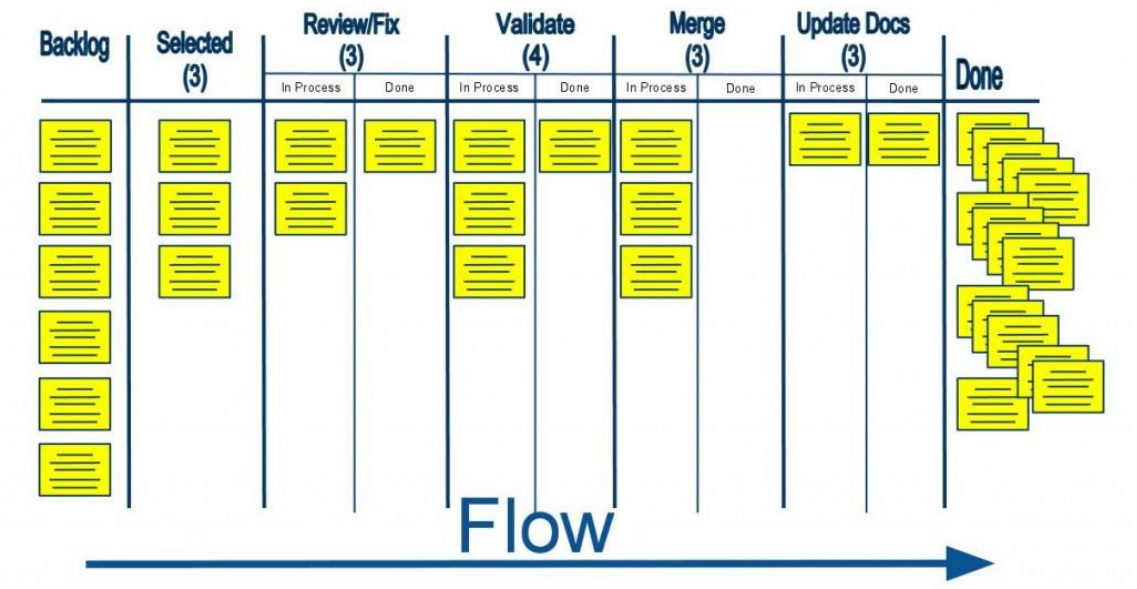


FIGURE 11. Example Kanban Board. Source: (Mulesoft.org - Rinaudo, Ramiro 2010)

Usually tasks in the project are managed using a board much like in Scrum but whereas Scrum boards describe only the state Kanban boards also describe the process as needed and it is dependent on team practices.

Like Scrum, Kanban also uses a Backlog where items are moved into the work area upon demand. This differs from Scrum where Backlog items are only moved during Sprint planning events. The demand or pull differs from manufacturing by a team member being free rather than something is sold away from finished goods. In software finished goods do not need to be replenished. This is why limiting or managing the work in progress is of utmost importance because a team member may be tempted to pull a new item if they are free rather than checking if something help can be offered to work in progress. The team itself must decide how to limit the work taken in at any point. Limits can be set by limiting the number of cards that are in use like tokens. If no tokens are free then it is required to wait until one becomes free (Ladas 2015). In Figure 11 the work in progress is limited at each stage of the process by the number at the top of the column.

The simplicity of Kanban enables it to work well in most organizations that have other governance practices in use and does not interfere where other standard project management methodologies. (Ashmore and Runyan 2014). This often leads to hybrid methods such as Scrumban.

The fact that Kanban does not have predefined events like in Scrum a method of ensuring Kaizen (Continuous Improvement) is through the use of Kanban Katas. The definition of ‘Kata’ is ‘an exercise consisting of a sequence of the specific movements of a martial art, used in training and designed to show skill in technique’ (dictionary.com 2015). One example of the Kata is interpreted below based on the description from ‘Kanban in Action’ (Hammarberg and Sundén 2014).

Daily Kata: Questions similar to the Scrum system to clarify what is trying to be achieved, the current status, any problems that need to be addressed, next steps and what have we learned?

Improvement Kata: Looking at the current processes and working practices and agreeing what is going well and what needs addressing. A plan should be drawn up for actions to correct where needed. This can be done as frequently as the team daily, if needed.

Coaching Kata: Coaching people in the team to improve through standard coaching methods.

Using Kanban

Kanban is best used when priorities changes often and it is difficult to make a plan for a Sprint, such as in Scrum. When the items in the Backlog are difficult to break down into short-term pieces or estimate effort at all (Hawks 2012). Examples of suitable places would be ticket-based work such as IT department service desks. In software projects errors are also much easier to manage in Kanban than Scrum. The key to Kanban is its ability to handle a constant workflow and release at any point as needed.

3.6.4 Scrumban

Scrumban is obviously a hybrid of Scrum and Kanban. It takes on the best practices of Scrum and brings Kanban in as a method to control work in progress per phase instead of per Sprint (Loitto 2012, 42). Ladas (2015) suggests that it can come about as a result of teams who are exploring Kanban but find comfort in the established methods of Scrum. Some limiting factors of Scrum may be unworkable, for example, some user stories may simply be too large to complete in one Sprint (Loitto 2012, 42). Another working hypothesis is that the wrong method is in use for the circumstances in the project. Perhaps the Backlog has become too volatile for some reason or there are items coming to the Backlog that are more urgent than items in the current Sprint. These could be, for example, urgent defects coming from published versions of the software that need addressing.

Ladas's web article (2015), approaches Scrumban as a learning platform to get to the true effectiveness of Kanban. Starting with Scrum and using the Continuous Improvement process to slowly replace the artefacts of Scrum with those of pure Kanban. Ladas also states that Kanban can be used with teams of up to 50 people. This hypothesis is unproven but, if true, can alleviate the restriction of keeping teams small that Scrum imposes.

Using Scrumban

As suggested Scrumban can be useful as a transition phase from Scrum to Kanban to facilitate a learning environment for a novice team. Scrumban should be considered at the end of a Scrum based project, during the hardening phase and when work becomes event driven by defects (Pahuja 2012).

3.6.5 No-estimates

No-estimates is a movement that can be found in various blogs, and revolves around a twitter feed #noestimates. For this reason, it is so far hard to find books or empirical studies on the subject. The movement was said to have started with a blog post and link to it, in a Twitter feed on 10 December 2012 using the aforementioned hashtag by Woody Zuill. The movement has many proponents, among them Neill Killick and Duart Vasco who regularly attend seminars in various countries to promote the movement.

In the blog, Zuill made an observation concerning a project he was involved in, in which the customer wanted estimates as well as the work done as quickly as possible. The first bold statement being that ‘regardless of how we define the word “estimate”, it is not a deliverable in the world of software development’ (Zuill 2012). The project had a predefined set of requirements that were to be implemented. As software releases were made, the customer started asking for items that were not in the requirements document. Observations from the project were that in the end about a quarter of the requirements from the original document were delivered and the customer decided that the project was ‘Done enough’ (Ibid.). Zuill convinced the customer that since he wanted work as soon as possible he would revisit the estimates later. After the first release, the customer no longer asked for estimates. The point being that working software is more valuable than estimates (Ibid.).

A presentation given by Killick (2013) summarizes the principles of the No-estimates movement. The points of the presentation are interpreted below.

Normally the debate of estimates comes down to the questions of what, when and how much? They are constraints put on projects by sponsors to ensure that they get value for money. These are actually questions of predictability, but does estimating tasks really

provide predictability? Arguably not, since most software project are said to come in late or over budget compared to the estimates at the start, and therefore the estimates become the arbitrary deadlines. Software projects can deliver predictability by delivering software and value frequently, not by estimating. As Zuill observed when this happens the customer loses interest in estimates because the value for money is clearly in place.

Estimating can create arbitrary constraints, so focusing in value not cost. Stating that a feature will take a month to develop actually creates a constraint that is not meaningful in the delivery of value. It is recommended to use real constraints such as a time frame or budget. In other words, get the most value in a fixed time or budget by delivering the highest value features first. If needed each increment can be a decision point to continue to sponsor the project.

When developing, use Lean and Agile practices and increment properly, assessing the value of features empirically. The feature must be published so that it is known if the customer actually needs it and avoid incrementing features that have no feedback. Delivering often will increase the courage to 'Pivot' properly. This is achieved by having so little effort between releases can make the decision of throwing away all the easier.

As the project progresses the delivery rate will slow down as the software get more complex. This is when the real constraints of time or budget can create an effective and predictable finish point.

Break up the work into as small pieces as possible using a slicing heuristic. An example might be a story can only have three tasks or one acceptance test. This enables estimation based on throughput rather than guessing, which is a proper empirical process.

The No-estimates movement is not against doing estimates. It simply promotes the use of real data and metrics to provide the predictability, and estimate when features will be done.

Also it is clear that, estimating the Backlog is a fruitless exercise. This is because the size, in effort, of the Backlog does not matter if you are not committed to deliver items in it. If the stories have been broken down small enough, or using a heuristic, their average size

will even out when there are enough stories so estimating their size becomes either pointless, or very easy.

Predictability can be achieved using techniques based on the existing performance of the team. When using for example a Kanban style of managing tasks, it effectively create a queueing system. When a queueing system is in place, Little's Law can be applied, for example.

Little's Law

Little's Law states 'that the average number of items in a queueing system, denoted L, equals the average arrival rate of items in the system, λ, multiplied by the average waiting time of an item in the system, W' (Little 2011) .

$$L = \lambda W$$

As suggested by Thomas (2015), it is often written in software circles as:

$$WIP = Throughput * LeadTime$$

$WIP = Work\ In\ Progress = average\ number\ of\ items\ in\ process = L$
$Throughput = average\ departure\ rate = \lambda$
$LeadTime = average\ time\ an\ item\ spend\ in\ the\ system = W$

Using a hypothetical Kanban team's metrics we could predict when an item in the Backlog would be ready for release. If empirical data from the team concludes that the team takes two days to complete each user story (Throughput) and the work in progress is four user stories at a time, we can calculate that the Lead-time is two stories per day. If the interest was how long it would take to deliver the eighth item in the Backlog, the sum of the queue and WIP can be used 4 + 8 / 2 = 6 days to deliver.

Using the No-estimates method

As discussed, it requires breaking down stories into as small parts as possible for the use of No-estimates to be successful. If the organization has a positive cash flow and does not charge directly to customers, for example, by charging a monthly fee, time spent doing estimates can be considered time not spent on coding and adding value to the product (Heusser 2013).

3.6.6 Dynamic Systems Development Method (DSDM)

DSDM was created in 1994 to address the problems of the traditional approach to projects such as, too big, too slow and not transparent enough. It was designed to build quality into Rapid Application Development. It is a framework with a rich set of roles and responsibilities that are suited well to corporate project environments. It is an iterative approach that allows details to emerge over time and is not limited to software development (DSDM Consortium 2014).

The key idea of DSDM is to fix the time and cost restraints in a project and adjust the functionality (Scope) accordingly (Abrahamsson et al. 2002, 61). One of the fundamental assumptions of DSDM is that nothing can be built perfectly the first time, and that 80% of the value can be delivered with 20% of the effort that it would take to produce the full solution, known as Pareto's Principle (DSDM Consortium 2014). Using this argument, it is functionality that should vary, not the other constraints.

MoSCoW prioritization is applied to requirements, user stories, tasks and tests. The acronym comes from the categories into which features are placed: Must have, Should have, Could have, Won't have (DSDM Consortium 2014).

The Process

DSDM projects have six phases of execution. The first three are to establish that the project is aligned to business goals and is feasible to do. The following are the actual development and delivery phases, which are done incrementally and in a time-boxed manner. The post-project phase reviews, are how the business goals are met (DSDM

Consortium 2014). The following sections interpret the key points of the four main phases taken from the DSDM Agile Project Framework Handbook (Ibid.).

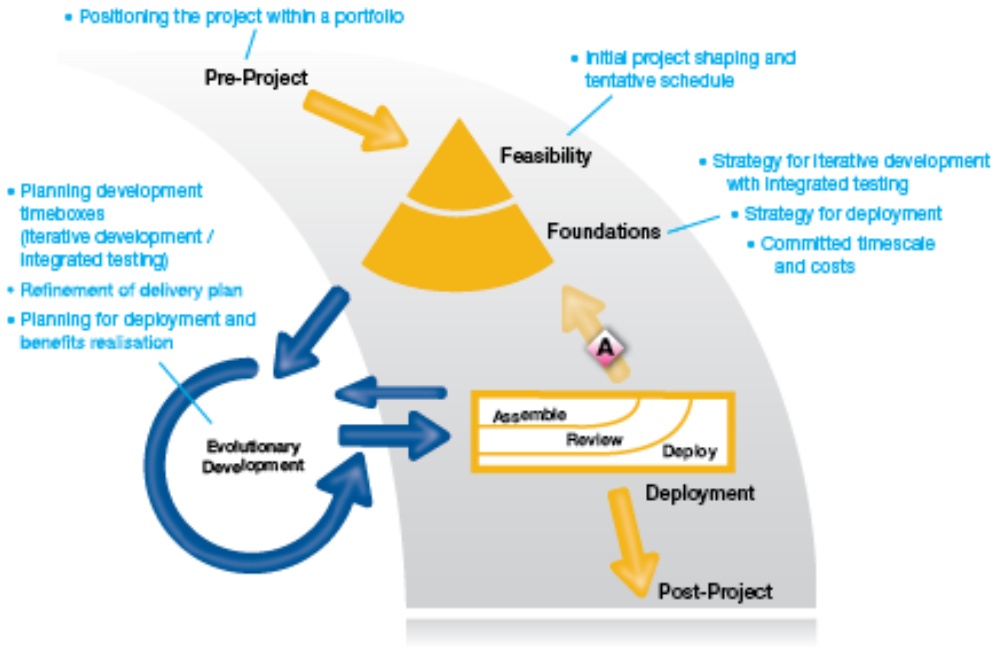


FIGURE 12. The DSDM Process. Source: (DSDM Consortium 2014).

The **Feasibility** phase is used to establish that the project is possible from a technical standpoint and if it will be cost effective. This phase should only use enough effort to establish whether further investigation is of benefit, or should the project be stopped immediately.

The **Foundations** phase is intended to build upon the Feasibility phase. It should be used to understand the solution that will be created by the project and how the project will be managed. Essentially, it is to create insight into the scope of the work, how it will be done, who will do it and possibly the constraints of time and budget.

In the **Evolutionary development** phase the project features should be developed in an iterative and time boxed manner. The highest value features should be developed first as prioritized by the MoSCoW method. The team can explore the finer details of the features and test both output and the business value continuously.

The **Deployment** phase is broken down into three sub-phases. The *assemble* phase brings together a coherent delivery. This can be anything from integrating software to gathering

documentation and training needed for the project delivery. The *review* phase establishes that the solution meets business needs and is complete enough to deliver. The *deploy* stage is the act of putting the delivery into operational use or enacting business changes.

Roles and responsibilities

The roles in the project are broken down into two main categories. In the **Project Level** category are roles such as the Project Manager, Business Sponsor, Business Visionary and Technical Coordinator. There is also a Business Analyst who will also form part of the development team to assist with development requirements, for example. These people will usually form some kind of steering group. It is important to note that these people must use an empowering leadership style, which will allow the Agile development team to self-organize and learn (DSDM Consortium 2014).

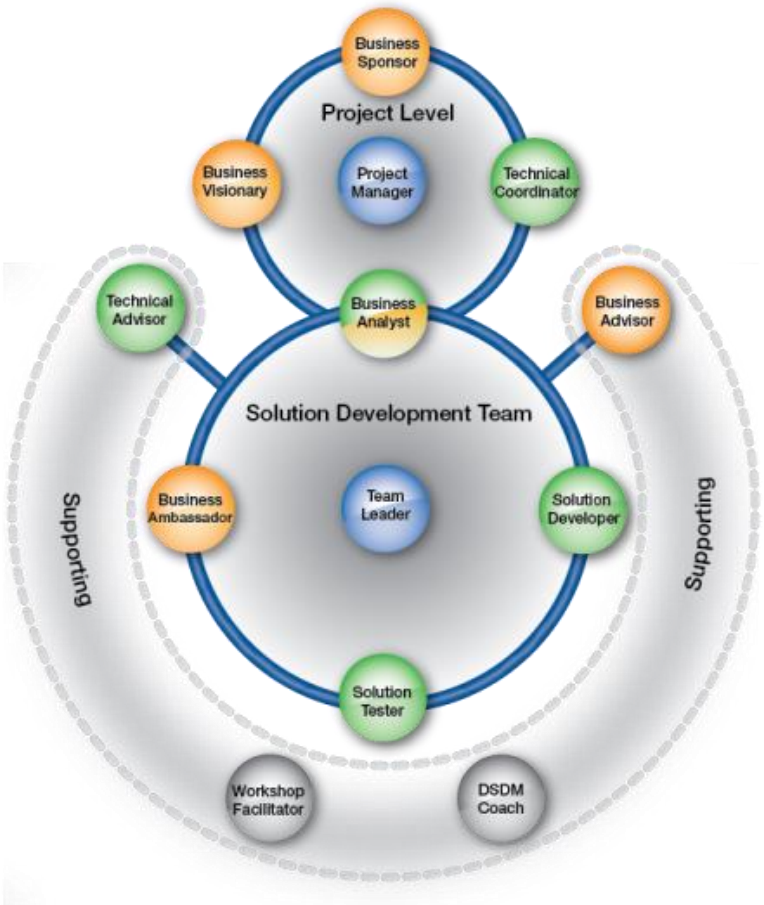


FIGURE 13. The DSDM team model. Source: (DSDM Consortium 2014)

In the **Solution Development Team** is the already mentioned Business Analyst, a Team Leader, Solution Developers, Solution Testers and a Business Ambassador who will be asked to create and prioritize features. More than one role can be assigned to a person. Other support roles exist too, such as a DSDM coach and other roles that will facilitate the project either on a process or technical level, as needed throughout the project (DSDM Consortium 2014).

Using DSDM

As mentioned, DSDM is well suited to corporate environments where the requirements are not well defined at the start of the project but the project is sponsored and defined in a roadmap. DSDM is a good Agile alternative to use when the project has a strict deadline and features can be prioritized using the MoSCoW method.

3.6.7 Other Agile methods

There are many methods that are considered by the software community to be Agile but are not documented in this thesis because they are unlikely to be in use in the sample pool of the study. This thesis has heavily documented Agile methods due to their persuasiveness in the software community.

Other methods such as the Crystal, which include a number of different methodologies to enable the selection of the most suitable, for a project based on varying levels of heaviness of the process (Abrahamsson, Salo, Ronkainen & Warsta 2002, 36). Feature Driven Development (FDD) that focuses on the design and building phases (Abrahamsson et al. 2002, 47). The Rational Unified Process (RUP), that focuses on use cases to model requirements in object orientated systems (Abrahamsson et al. 2002, 55).

The list goes on as Agile methods are adapted, built upon and mixed to try to suit ever demanding project needs. Many of the methods overlap or are used interchangeably to suit environmental requirements; one of the hypotheses spelled out in this thesis.

4 RESEARCH DATA COLLECTION

This chapter discusses the collection of the research data in detail. The background of the companies and interviewees is told to give insight into the experiences and competences of the research pool. Since the interviews were done in an informal way, a description of how the research questions were indirectly approached to avoid an interrogative style of interview.

4.1 Diversity of the research data

In total, nine interviews were carried out among five companies. The names of the companies that volunteered personnel to be interviewed are withheld and coded. This is because a few of the companies carry out confidential R&D activities and wish to obfuscate any information about their methods in the public domain. The original plan was to interview more companies but real life factors such as legal matters, difficulty in finding volunteers and of course time constraints reduced the focus somewhat.

The first pilot interview was the only interview carried out in a company which will be coded Company A. The company is a successful professional services company. The company is focused around providing user experience design and software delivery project in the mobile and internet economies. The project strategy of the company is working very closely with customers delivering projects using various Agile methods. The culture of the company is fairly relaxed and trusts project managers to decide the best practices used. The interviewee is in the role of a project delivery lead, which also includes project management, being the customer interface and software development activities. He has worked in the software industry for well over a decade and can be considered a seasoned software professional and manager.

Two interviews were held at Company B. This company is very similar to Company A in many respects such as customer base and style of management and products, which are mostly software and user interface design. The company does have specific skills in a popular development language but does not limit its activities as such. The structure of this company is very flat with respect to organisation with almost all employees on one or two levels down from the CEO. In this respect the management of the company fully trusts its teams to work and organise themselves and use best practices. Like Company

A, Company B works very closely with its customers. The first interviewee works as a project lead, which included project management, technical leadership, development and working as the customer interface. He has worked in the software in the software industry for less than a decade but is considered as an experienced developer and project leader. The second interviewee is mainly focused on user interface design and graphical projects, and leads people with a similar skill set. He was not as experienced in leadership, having only worked in the role for some months.

Company C is a large corporation that has many business streams globally. Three people were interviewed in this company who all came from software delivery projects in the same organization. Since the company develops its own mass-market products, the project managers do not come into contact with the customer directly. Project requirements are formed by various means from end user feedback to competitor analyses. Due to the size of the company the interviewees cannot be considered to represent the company as a whole. They represent the practices of a small software delivery organization. Most of the project managers in this organization hold PMI certification, even though PMI practices are not used. The first interviewee has a long background of software project management with more than fifteen years of experience. She focuses only on project management and leaves technical leadership to others. The second worked as both a project manager and a technical leader. He had between six and ten years of experience in leading projects in a technical sense. The last interviewee also has over ten years of experience in leading projects and also chooses to focus only on project management and team leadership activities.

Company D is a small game producing company that has a background in developing a 3D first-person shooter game. Only one person, the CEO was interviewed, who along with managing a company has a background in managing projects in a large multinational corporation. The interviewee manages other companies in the media entertainment business but when interviewed tried to focus on the game development activities and how a small company approaches the development.

Company E is a moderately sized business that manufactures heavy lifting equipment such as cranes and large forklift trucks among other things. It is a company that executes projects globally and teams are rarely geographically co-located. The first interviewee works in an automation organization that specializes in the automation of machines that

are already in service, in other words, retrofitting automation equipment. She has around ten years of experience in project management gained in both multinational and small companies. The second interviewee worked in an organization that specialises in delivering large-scale projects that have sub-project managers and other functions, i.e. it is a program management role. He also has some eight to ten years of experience, mostly gained in the company in question.

In order to continue the anonymity of the interviewees any references to their interviews will be coded

TABLE 1 Overview of researched companies and people.

Company ID	Company type	Nº	Interviewee
Company A	Professional Services	1	A1: SW Project Lead
Company B	Professional Services	2	B1: SW Project Manager B2: Graphics lead
Company C	Multinational Corporation	3	C1: SW Project Manager C2 SW Project Manager / Technical C3 SW Project Manager
Company D	Small Games developer	1	D1: CEO / Game producer
Company E	Large equipment manufacturer	2	E1: Project manager E2 Project Director

4.2 The interview process and questions

The interviews were executed as an Appreciative Inquiry (AI) as discussed in section 2.4.1. The interviews were structured in a relaxed way to allow the interviewee to elaborate and talk freely. The line of questioning was structured as discussed below but not all questions were asked since the interviewees often covered the topic indirectly when answering other questions or discussion items.

The discovery phase of the Appreciative Inquiry tackled the first three research questions. These questions were not asked directly but were approached by asking leading questions that attempted to bring up the answers to the main research question of interest. The breakdown of the questions below shows the key structure that was used in the discovery phase.

What motivates project managers?

Although this question is part of the of the fourth research question, all the interviews started with this question to warm up the interviewee and make them comfortable with talking about themselves. The question was usually asked as a simple ‘what do you enjoy about project management?’ style question.

How projects are managed in a selection of Finnish companies?

The interviewees were asked to describe what practices and process they use to execute projects – from initiation to completion – including the roles, events, meetings, practices and quality assurance activities. They were prepped before the interview to avoid using statements like ‘we use Scrum’ so as to avoid constraining the answer and not allowing the discovery of actual activities. In most cases they were separately asked to clarify the roles in the team, a question of who does what and why.

Again, not asked during the Dream phase, but asked here to prepare them better for the Dream phase later, they were then asked to share an experience at any point in their career that a project went very well and describe why they think the project exceeded expectations or just went well.

Based on the fact that Lean management should be possible no matter what methodology is in use, the participants were asked to describe how they as a project leader, avoid waste in their projects.

Why such project management methodologies have been chosen?

To gain insight into this question, the participants were asked what kind of project management methodologies they are using; what others in the organisation that they work are using; and how the organization selected those methodologies.

What constraints are the projects subjected to by their organizations or customers?

In this case the participants were asked about how their projects are constrained, how they satisfy customers' needs and how the organization that they work in contributes to the successful outcome of projects by means of training or other activities such as PMO oversight.

What project managers see as the ideal environment to manage projects?

The last question forms the Dream phase of the Appreciative Inquiry in which the participants were asked to use their imagination to describe how they would manage a project if their only constraint was satisfying the customer's needs. What practices, methodologies they would use and what they would expect from the organization that they work in, in order to support the successful outcome of projects.

This formed the end of the interview but the participants were asked if there was anything that they wanted to add to the discussion or to offer feedback. The Design and Destiny phases of the appreciative inquiry were skipped, but it was described to the participants how these stages could be put to use if similar data was collected from a larger audience, for example, in a team workshop.

4.3 Methods of data collection and analyses

All of the interviews were recorded using a digital audio recorder. The recording formed the raw data of the interviews, which was then transcribed to text. This produced approximately one hundred pages of textual data to be analysed. The textual data was then separated roughly to break up the data to correspond with the questions that were asked forming the meta-themes. This was then followed by copying relevant sentences and statements into respective meta-theme Excel sheets for further breaking up into themes or sub-themes.

The various Excel sheets were then analysed by means of coding the statement into interpretations, which the statements corresponds to. In other words the themes and sub-themes were found by looking at the statements holistically, and assigning a code to each

statement. The sheet also maintained the data of who made the statement. An example of the coding process is given in Figure 14.

Coding	Statements	A1	B1	B2	C1	C2	C3	D1	E1	E2
CUSTOMER	discussions with the customer	x	x							
SOLVE_CUST	solving their problem	x								
SOLVE_CUST	understand that what they want done	x								
SOLVE_CUST	improve their lives	x								
PEOPLE	meet with a lot of different people		x		x					
NEW_DAY	can contribute to different tasks		x							
MULTI_PROJ	multiple different projects		x							
CUSTOMER	Get the customer involved		x							
PEOPLE	working with highly experience people		x							
RESULTS	guarantees project success		x							
BIG_PIC	Sharing knowledge		x							

FIGURE 14. Example of coding of statements.

The column on the left-hand side was the codes that were given to each statement. As it can be seen from the example, some interviewees may have made several statements that corresponded to and were coded as the same theme. In Figure 14, for example, A1 made several statements that were given the code SOLVE_CUST.

The codes were then filtered down into a summary table that gave a title to the themes based on the statements that were made. This allowed the summing up of who made a statement to a corresponding theme. Although an interviewee may have made several statements that correspond to a theme, only one point per theme mentioned was given. This means that the maximum number of contributions to a theme cannot exceed the number of participants. Figure 15 is an example of the summing up table based on the coded themes.

Filtered and interpreted		A1	B1	B2	C1	C2	C3	D1	E1	E2	
RESULTS	Seeing and affecting results	x	x		x		x	x		x	6
CUSTOMER	Working with the customer	x	x			x			x		4
PEOPLE	Meeting different people		x	x	x			x			4
MAN_LEAD	Managing and Leading	x				x		x		x	4
BIG_PIC	Seeing the big picture		x	x			x				3
NEW_DAY	Every day has something new	x	x							x	3
SOLVE_CUST	Solving customer problems	x							x		2
REAL_ISS	Handling real issues								x		1
MULTI_PROJ	Multiple projects		x								1

FIGURE 15. Filtered themes.

The summary tables were then used to create the graphs that are used throughout the discussion chapter. In some cases, the themes were grouped to support the discussion

further. An example of this is the meta-theme of virtual teams where the themes were grouped into two categories. The first being the question whether the team is virtual or not and the second question is that if the team is virtual does it produce problems? This example can be seen in section 5.3.3.

The author has a long experience of over 12 years in managing mostly software projects. In this time, various projects methods such as Waterfall, incremental and, of course, Agile have been used to deliver projects. This experience has brought various opportunities for learning. These include hands-on practice as well as courses and seminars in management techniques. Also, the project management courses in the Master's degree program, which this thesis is part of. This experience, plus the concrete data gathered through the interview process described here was combined to derive the conclusions discussed in chapter 6.

5 SURVEY DISCUSSION

5.1 How projects are managed in a selection of Finnish companies

The initial idea behind this question was to identify the processes used in companies, but unfortunately that became very difficult to analyse due to the sheer amount of data and differences in every company. If the process in each company were to be documented, it would produce a large volume of unrelated strands of information which would probably add little value beyond sharing good practices.

Due to the volume of the data, it was broken up into three parts. The first part was looking at the roles that exist within project organizations. When asking about the roles within a project the respondents also spoke of responsibilities. The responsibilities were difficult to tie to a specific role, so for the sake of simplicity they are not linked.

The second part is the activities that take place at project initiation and planning. Originally, initiation and planning data was separated but, as the analyses began, it was realized that the project managers did not separate them so clearly – and in some case perhaps did not see any difference in the stages. Therefore, the data was merged and analysed together.

The third part is the project execution phase. Again, the activities which took place were the main focus of the analyses. There were a lot of other statements in this data set but it was difficult to identify themes or any common statements.

During the interviews, as a matter of course, project-closing activities were asked but only a couple of project managers had anything significant or clear to discuss. Why this is the case, it is hard to tell. Most of the project managers are managing software projects and they tend to finish quickly once the final release is done. It could be argued that these types of projects just end without formal ramp down activities and documents. The people are simply moved to another project immediately because of priorities and demands.

5.1.1 Roles

It should be noted at this point that because certain roles were not mentioned – or mentioned by more of the participants – it does not mean they do not exist. It just did not come to the mind of the person being interviewed. A good example of this is that engineers are only mentioned twice in conversation. It is self-evident that all of the teams have engineers. Therefore, it can be assumed that the numbers below are not reliable for anything other than what was important to the person being interviewed.

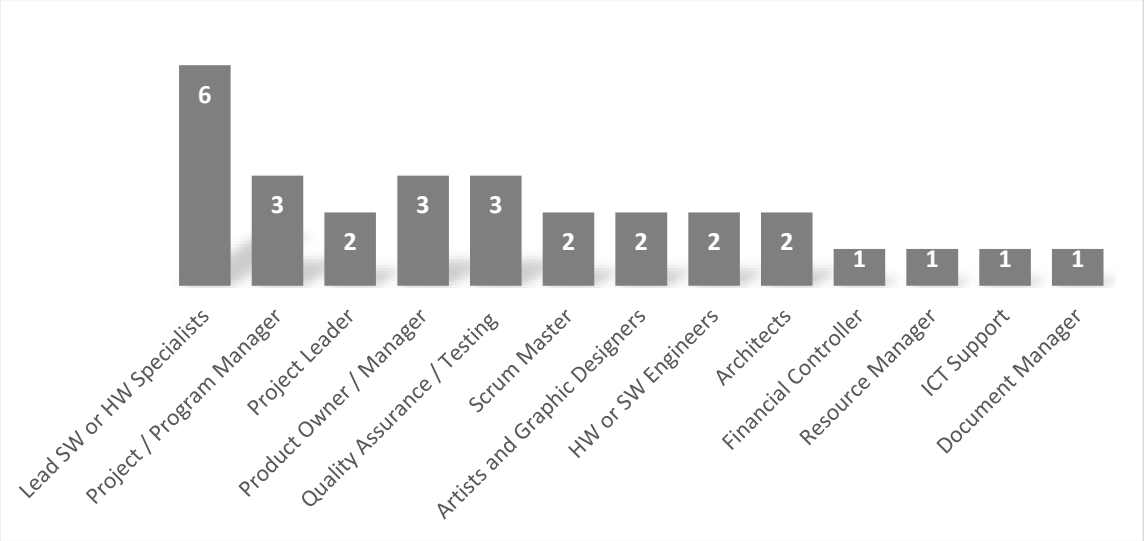


FIGURE 16. Roles within a project organization.

Specialists or lead SW and HW developers were the most identified personnel in the projects. A1 roughly described the lead developer as '*someone who knows about the something that that we are doing*' and B1 noting that it is '*people who know technologies*'. Most of the other participants mentioned that they have a 'Lead Developer' quite directly and in these cases it means lead software developers. The only exception to this trend is in Company E where the engineering staff are involved with equipment automation. In those cases the lead hardware and software people are involved in planning and designing systems.

As we know from Table 1, only five of the nine participants described themselves as project managers. As we can see from Figure 16, only three mentioned the actual role of a project manager in their conversations. This could be because the role was implicit and did not warrant a separate mention. For the sake of this discussion it seems that merging the project manager and project lead role is valid; this would make the mentions of the

role up to five. A1 and B1 used the term project leader and when they described the role and it sounded similar to a project manager's responsibilities – for example, being responsible for the outcome of the project. The others, when mentioning the term project manager gave examples, such as from E1 who stated in so many words that typically the project manager is supervising the on-going project. Another mentioned his role as being a project director; which appeared to be the company term for 'Program Manager'. He specifically mentioned that he has Sub-project Managers in his teams, which fits the description very well. In the PMBOK a Program Manager is described as managing the program staff and project managers, providing vision and overall leadership. (PMI 2013, 8).

Teams that have product owners seemed to be limited to Company C, where the product owner was described as the business owner and communicates with the outside world. As noted earlier, the projects in this organization do not have direct contact with customers, so this person appears to work as a subject-matter expert who knows the trends of the products being produced. The only other mention of product ownership was in the gaming company where the CEO also is the product manager.

It appears that the role of separate test or quality engineers is the privilege of larger companies in the projects in the sample pool. In the case of Company C, there seemed to be the roles of manual test engineers and test automation engineers. It can be assumed that in other companies either the project engineers make automated tests or customer is responsible for that.

Scrum mastering was only mentioned twice as a specific role and only in Company C. In this case it was the project managers who were acting in this role. This is a good example of non-textbook application of the role. This is an example of the Hypothesis in section 3.6.2; that the Scrum Master is usually a senior member of the team, not equal as textbook Scrum promotes.

The other roles are clearly less prominent, according to Figure 16, so each will be described shortly. Artists and Graphic Designers exist in some teams, usually on a part-time basis. The only exception to this is, of course, the team led by the Graphics Lead in Company B and a full-time artist in the gaming company. In this case, the artist is, of course, designing graphics and textures for the game full time.

Software Architects typically work as Senior Specialists outside the teams and tend to take a holistic role in designing systems and managing their dependencies. They are usually more experienced than the Lead Developers.

The last four roles identified are Financial Controller, Resource Manager, ICT support and Documentation Manager, but they were only mentioned once and therefore cannot be considered any kind of theme. They are kept here for completeness and to emphasize that in larger projects tasks such as financial management are delegated by the project manager to other professionals.

5.1.2 Responsibilities

The responsibilities came up as an extra piece of data that could be analysed from the roles conversations. This is also not reliable data because not all of the participants spoke of role responsibilities. In some cases it is also difficult to tie the responsibility to the role so that fact is ignored for simplicity. There was also a low number of themes when the data was analysed, leaving in question the reliability of this data. It is included in the thesis because it gives some insight into which activities the participants see as important, at least.

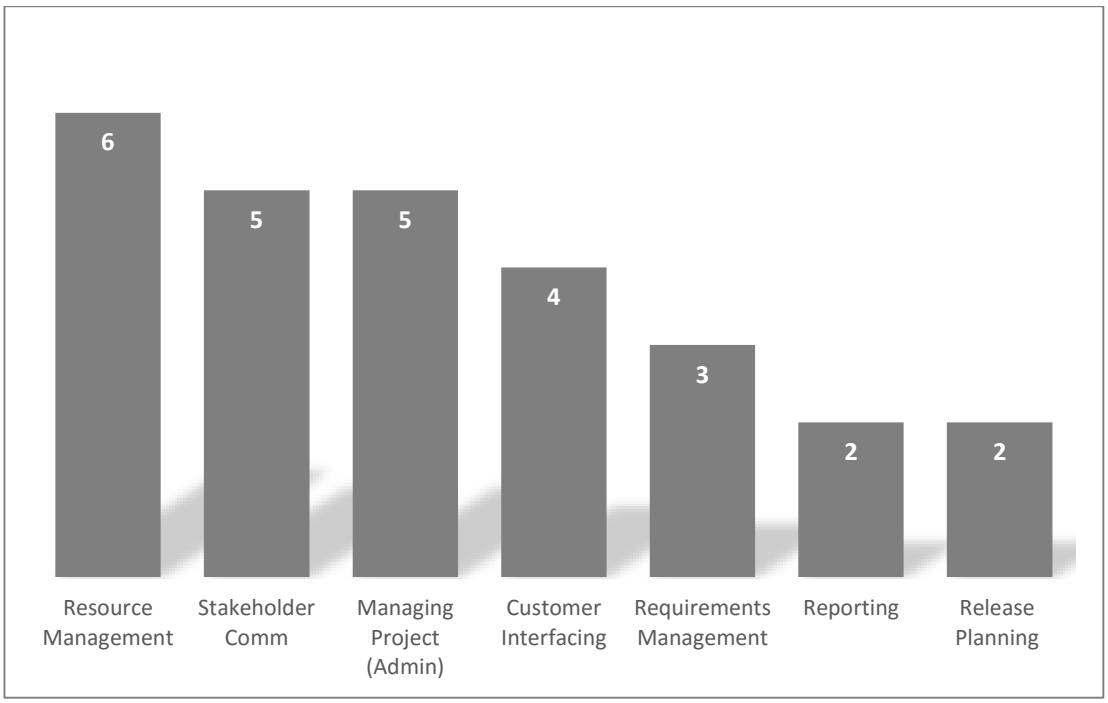


FIGURE 17. Responsibilities in projects.

The theme of resource management has quite a wide scope. In Companies A and B the discussion was along the lines of having good goals set for the team members and finding the right kind of people to work on the projects – i.e. finding people who have the right motivation. Also, moving people between teams was mentioned, in order to create an environment where people could grow and learn better – which makes for an interesting strategy. In Company C the participants spoke more of networking to get expert opinions and working with related teams. In Company E resource management was clearly in the domain of having the right people in the right place at the right time.

Stakeholder communication also includes facilitation activities, too, such as facilitating meetings. B1 spoke of spending effort with talking a lot with stakeholders in an effort to keep everything visible to everyone. The participants from Company C were quite vocal in this area. All spoke of cross-functional communication to other organizations in the company to check legal matters, communication with different teams within their organization and, of course, stakeholder management. An interesting comment arose of acting as a proxy between the team and others to prevent unnecessary interruptions. E1 spoke of having to be in the centre of all of the information flowing around from different functions and following what is the status of other delivery teams.

A1 explained that many administrative tasks fall into the Project Leader's hands along with decision-making, but the style is that decisions are often made by discussing with the whole team. At Company C tasks such as line management and gaining support from the organization was part of managing projects, along with of course monitoring and controlling the projects. In Company E the task of project admin involves the collecting of information and checking everything is acceptable along with ensuring the correct parts are ordered for projects.

The task of customer interfacing mainly involves being in close co-operation with the customer. In Company A they like the customer to be able to make decision because it is their money; they like to get steering from the customer. In the other companies the discussion mainly focused around conferring with the customer on a regular basis.

Only Companies A and C mentioned requirement management. A1 mentioned that the project lead is responsible for the Backlog, giving a possible hint to the question of who

is actually producing user stories. C1 simply mentioned requirements management when they were listing their tasks – no further insight there. C2 opens this subject a little by telling that it involves refinement of the concept and managing the dependencies.

Reporting and release planning were hardly mentioned, but in this case it would be safe to assume that they are implicit. Project sponsors demand reports about their investments, without doubt. It is clear that release planning concerns the timing of deliveries to the customers.

5.1.3 Project initiation and planning

Project initiation usually refers to the point in time when a need has been identified by a customer or organization for a change. The change can be anything from a process to a physical product that meets the needs. In some cases, the need might be very clear and the solution is sold based on variations of existing products. These are classified more as delivery projects. In these cases the project initiation is more like a sales phase and quickly moved into planning.

In some cases, especially in the software industry, the need and the solution are somewhat fuzzy and indeed it is very common for the customer to not know exactly what they want. This can then be considered more of a Research and Development (R&D) project. In these cases the initiation phase focuses on defining the concept and how that will be developed to meet the need.

The planning phase should take place after the initiation, in the perfect world, but projects – especially software projects – do not adhere to that rule. Very often the planning and initiation is merged to assist in understanding the size of possible solutions and in the case of true Agile thinking, the concepts is never fully agreed so planning can be cumbersome. As mentioned earlier, when the interviews took place, in most cases there was no clear distinction between initiation and planning.

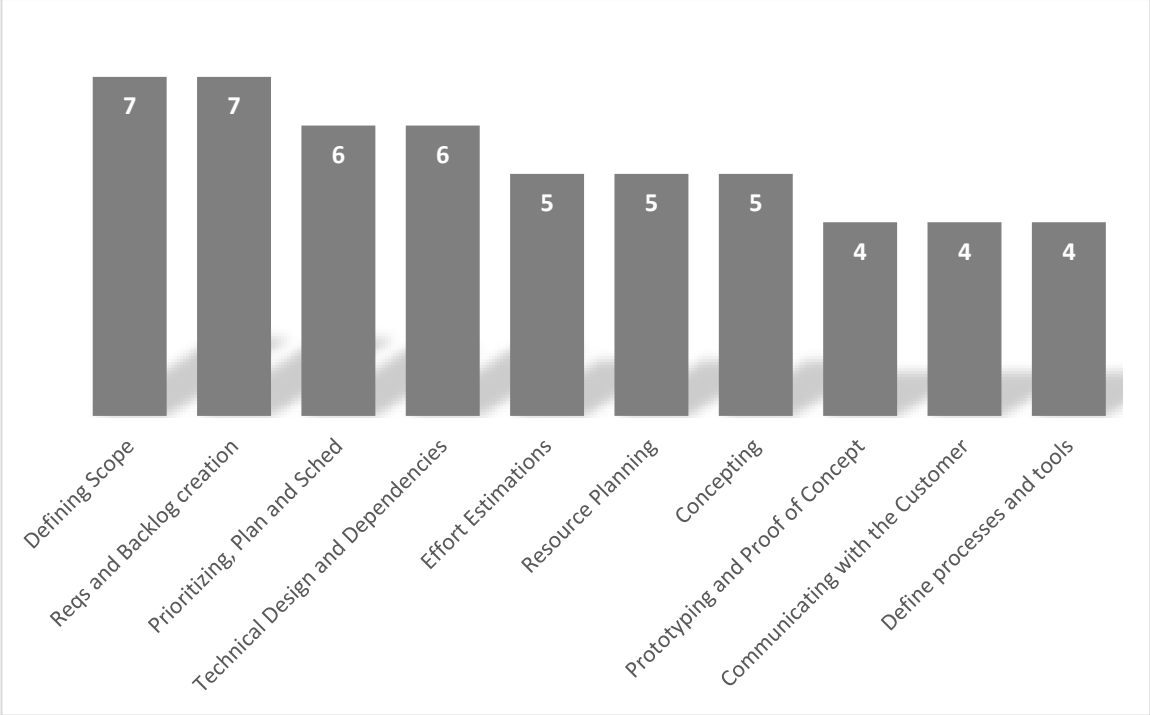


FIGURE 18. Activities during project initiation & planning.

All participants except those in Company B spoke of defining the scope of the project. A1 described this as defining the content on a higher level, figuring out what needs to be done, understanding the expectations and having a kick off to lay the foundations of the project. In Company C there was mention of looking into trends and consumer needs. This is probably a consequence of the fact that very little direct contact with the customer exists. C1 mentions it is planning '*what we they are going to do and understanding the needs*'. In the words of C2, it is '*identifying the key deliverables*'. For C3, it is also about managing the feature set and introducing the concept to the development team, and checking that they understand the goal. In the gaming Company D scope had to be defined because there were demands from organizations that were providing funding and they had to fulfil their requirements. In Company E, the projects are clearly biased towards delivery and this can be seen with statements like: '*handover from sales*', '*a need to record everything that was sold*' and '*discussing with the factory*'.

When planning was brought up in discussion, it appeared to focus mainly around schedules and deadlines, or to be more specific, how long will a project take? A1 also mentioned that they plan what tools and methods will be used to complete the project. B1 specifically mentioned prioritization of the tasks and, interestingly, the observation that customer is not always capable of prioritizing themselves. C1 also mentioned prioritization and ensuring that they have the means to carry out the work. All of the

people in Company C mentioned fixed deadlines and although the teams are using Agile, they have to do a fair amount of planning, such as releases in a Waterfall manner. D1 said that they had a plan from the beginning and they did three weeks of hard-core planning. This was to again satisfy the demands from investors.

In technical planning, the common theme was designing the system and understanding the technical depth of the project. C1 mentioned that the dependencies are important since the software is rarely standalone and D1 continued that point with a need to understand the relationships between all of the modules of the game. E2 departed from this somewhat by stating that they reanalyse the project requirements after sales handover to ensure that the system design really makes sense. This is probably a result of the delivery nature of the project.

A Backlog is essentially a list of items to be handled and differs from a work breakdown structure in the sense that the list is usually just a list with no structure or hierarchy. In most of the discussions the creation of the Backlog is breaking down the scope and requirements into tasks that have to be carried out. In some cases this was done into Excel sheets and other tools such as Team Foundation Server. C2 also mentioned the idea of using the Backlog to determine the minimum viable product or using the MoSCoW way of thinking – in other words, 'must-haves' are defined. A1 also mentioned the creation of an Epic, which is a collection of Backlog items that put together make a meaningful feature; much like a WBS item.

Among the pure software teams, it appears that effort estimation is done very roughly if at all. In Company A they only evaluate the relative size as discussed in section 3.6.2 about Scrum. In Company B it is not seen as that important, although, the interviewee did mention that they do not go to the extremes of #noestimates. They just try to get ballpark figures together. In Company C it was more like estimating the size of the project in man-months to help with resource assignment. In Company E we are again looking at delivery projects and one of the participants said that '*we always estimate*' and nowadays they use historical data to assist in the process.

Of those who mentioned resource planning, the theme was always about understanding the size of the team needed to complete the project. Although only five of the nine participants mentioned the practice, it can be assumed that all projects do this. In

Company C the organization gives the resources based on the man-month estimations. Some of the participants did mention that lack of resources is a common problem.

The conception phase is usually based around the idea and the needs of the customers and what problems should be solved. A1 mentioned the use of Graphic Designers to work with the customers to refine the concept and C1 and C2 also mentioned the use of Graphic and User Interface designers too. In Company E the creative people were gathered at this phase to do game level design. The phase is littered with kick-off meetings and discussing the features needed. Some interesting methods showed up in Company B, such as the use of mind maps and wall-sized white boards where the idea can be developed.

Lean thinking showed up in some places with the use of prototyping. In Company A, the idea is to get something working as quickly as possible for the customer, to give feedback and iterate; a strategy highlighted in #noestimates. In Company C the prototypes are used as a proof of concept to show that the technology is feasible.

From the results, it appears that Companies A and B have the biggest focus on the customer. A1 said that they like their customers to take a product owner type of role, if possible and they will adapt to their customers systems if needed, for example, by using their version control systems. B1 likes to work closely with the customer and usually they run most questions by their customers, too. They believe that the customer knows the details and want to spend as much face time with them as possible. E1 also spoke of wanting to meet the customer as early as possible and discuss with them how the delivery of their purchase will take place.

Again, the practice of defining tools and processes appears to be concentrated in Companies A and B, probably driven by the customer orientation of the business. A1 said that they decide what process to use, such as Kanban or Lean or another methodology and what the cadence of Sprints and releases and team practices will be. In Company B, they tend to be biased towards some kind of Kanban and will define at this stage what tools they will use to track the project. Trello was mentioned as a popular choice in both Companies A and B.

5.1.4 Project execution

The project execution phase comes after planning and initiation and usually involves a full team of engineering staff to implement the change that is required. The tasks that happen in this phase depend on the type of project and the nature of the environment that the team is working in.

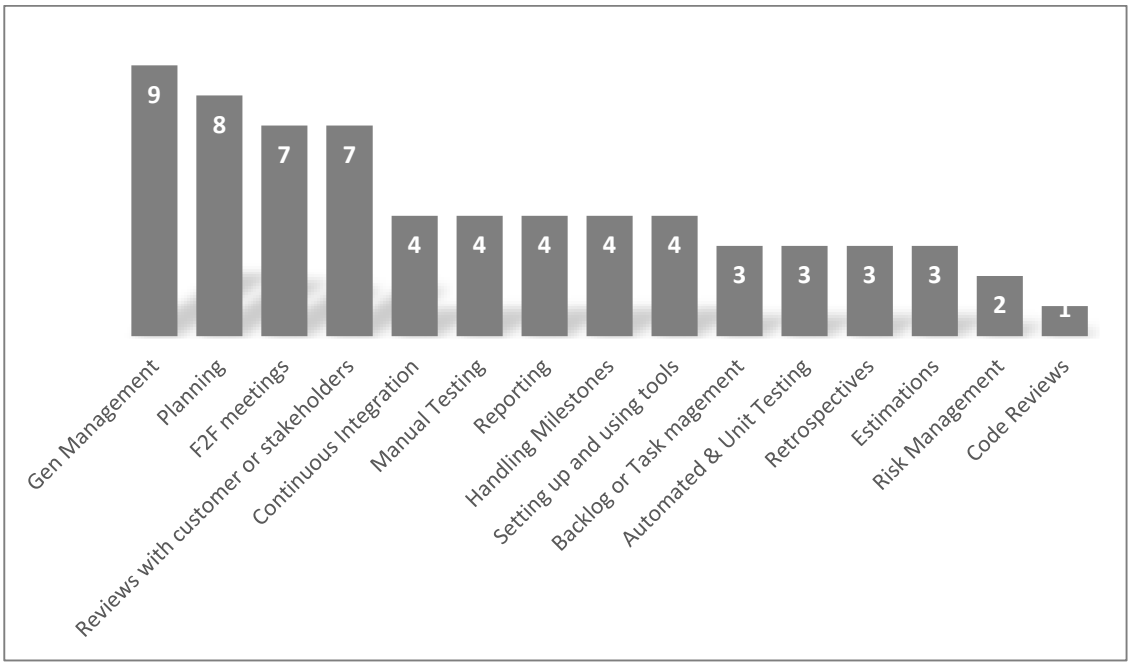


FIGURE 19. Tasks during project execution.

During execution, the task of general management ranges from solving problems as they arise to ensuring that processes get followed. A1, B1 and C2 saw that dealing with issues as they arise and removing impedances that are stopping the team from progressing is important. C2 also thinks that finding alternative solutions is important. There is also the act of ensuring that the right information is available with four of the participants stating that they spend time gathering information from the customer, getting expert opinions from other teams and that the information makes sense. Change management was brought up in Company E and that is again a result of the delivery style of projects that were sold as a specific solution. With respect to processes, B1 stated that they like to find the best-known practices and scale them up as necessary and E1 discusses applying rules and processes in the projects while, at the same time, trying to find ways to improve them.

Planning activities revolved around activities such as Sprint planning, which is a popular practice in Company C, cited by all the participants. The length being an unusually long,

one-month iteration cycle. In these meetings, the rolling plan is done prioritizing what will be done in the coming weeks. Then there are team-planning activities such as technology studies cited by A1, B1 and C2. Company E does a plan of what is going to happen in the coming months as well and clarification of what each the company and the customer is expected to do to ensure the delivery.

Face-to-face meetings happen in many formats such as the standard daily Scrum meeting. In Company B the Scrum meeting is held three times a week as opposed to the daily Scrum recommended in textbooks. The team sit close together so a more frequent meeting is not seen as valuable. This trend is similar in Company A where A1 stated that the stand-up may only happen once a week due to the close quarters of the team. Company C still favour the daily Scrum as a practice. In Company E no form of Scrum exists but there are weekly team meetings for status review purposes.

Reviews appear to take many forms in the projects. In those companies that have direct contact with the customer, a weekly review with the customer is standard practice. In Companies A and B they like to show the progress of the product to the customer and get feedback. In Company C, iteration reviews are done at the end of Sprints and demonstrations are carried out. The Product Owner accepts the work that is done. Also, in Company C, compliance reviews are carried out to ensure that the product adheres to certain standards such as privacy and security guidelines.

Continuous integration done in the industry standard way is evident in Companies A, B and C. When a developer commits code to the main code line, the build system will automatically be triggered and create a build. Then any automated tests such as unit tests will be executed and results sent to the team. The team will be expected to stop and fix the build if there are failed tests or the build fails.

Manual testing varied a little among the companies that do it. In Companies A and B the manual testing tended to be ad-hoc. Both companies have emphasis on automating their tests but A1 mentioned that they have testing cafes where people – not necessarily in the team – try to break the software, but as rule all tests are automated within the team. Both A1 and B1 said that they limit automated testing to the integration level because UI tests tend to be too cumbersome. Unit testing is the bare minimum level of testing that is carried out. In Company C, C1 told there are teams to do certain manual tests on release

candidates starting with smoke tests (sanity check) before heavier testing that can take up to a week.

Reporting was not so heavily recognized in the professional services companies. The two participants from Company B brought it up as sharing information between projects and a basic report that is sent to the customer. It is probably not so heavy in this company due to the high contact that they have with the customer. The only other company to bring up formal reporting was E, where the main issues are documented along with the progress. Some hint of project finalisation show up here with both participants stating explicitly that a project-closing document is always written. It was the only evidence seen of any formal closing practices among the participants.

Milestones were only evident in the larger Companies C and E. In Company C all three spoke of a concept milestone – perhaps an approval to continue – at the start of the project but no evidence exists of other milestone during the life of the project. In Company E, several milestones were listed throughout the project lifecycle which marked the commencement of certain delivery events.

Some effort is put to maintaining tools in the companies practising software development. A1 told that they like to have a tool chain that can deploy everything automatically, even into production environments. B1 explained that the tool chain depends on what programming languages are used and also the environment that the customer is using for production activities. Company E had all their information stored in Visual Studio, such as diagrams and To-Do lists, and used this method to track the project to some extent.

In the execution phase, only C1 spoke anything, beyond a passing mention, with respect to the Backlog management. The Backlog is groomed and prioritized and those items that might need pre-study are also identified. As mentioned before, Company E keeps their To-Do list in Visual Studio and they look at the relationships between items and prioritize development and cross off items as the get done.

As we know from the section discussing Scrum artefacts, retrospectives are used to reflect on team performance and practices, and look for ways to improve upon them. They are also known as continuous improvement. A1 told that Company A has a very big emphasis on retrospectives and enforces them as part of the company culture. They look at the

product and its quality and also if the team and customer are happy. One method given as an example is the use of pulse surveys to measure the level of happiness.

As discovered in the initiation phase, very little emphasis is placed on effort estimation and this trend continues in execution. A1 told that they do some kind of estimation to get to grips with what they have to do. C1 stated that '*from time to time there is a need to understand and estimate some bigger, long lead items*'. E2 emphasized that the cost structures need to be continually updated.

Risk management was only mentioned as a way to escalate when a project will be delayed, which is quite correct. Those who mentioned it, were one person in Company C and likewise E. No real information was disclosed about the techniques used to identify risks.

Only Company A1 gave any indication of code reviews taking place. The continuous integration environment is used and code can only be committed to the main branch once somebody has reviewed it, i.e. the continuous integration environment tool chain enforces a minimum code review practice. Also code coverage is measured. What is measured is not clear but, based on the authors experience, the automated unit and integration tests will be the execution test set.

5.1.5 Lean management and waste

As we recall from section 3.5, the definition of waste is anything that does not add value to the product or service. During the discussion the interviewees where asked to describe how they reduce waste in their projects. It should be noted here that this question was not asked in the pilot interview and therefore A1 is not included in the response pool. It should also be noted the participants were not introduced to the concepts of waste from the TPS handbook, as can clearly be seen from the types of answers.

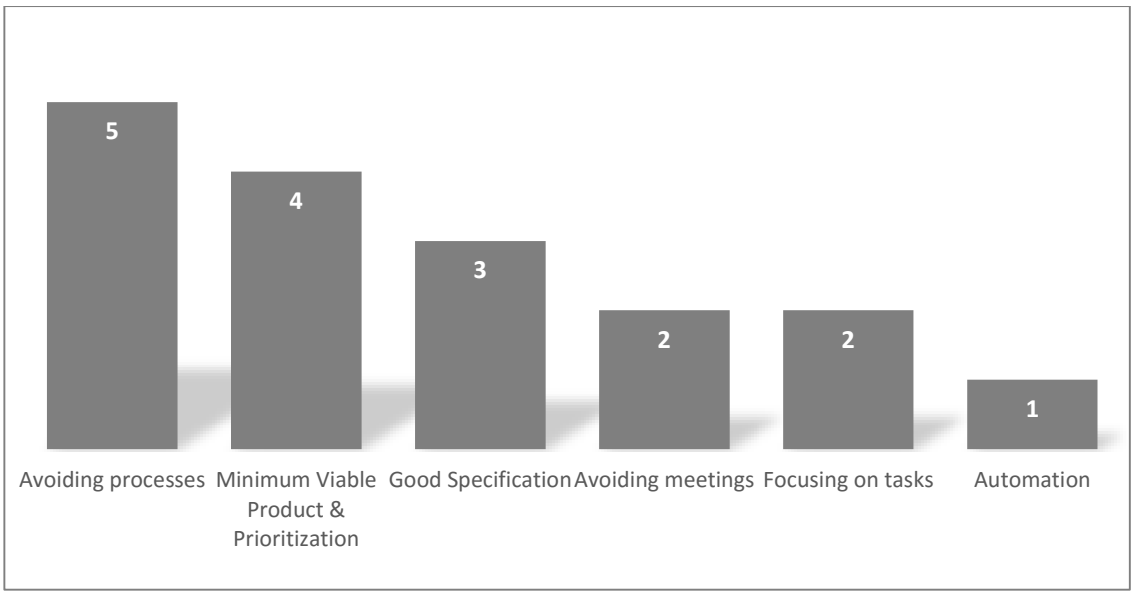


FIGURE 20. Methods of waste avoidance.

Avoiding process was the favourite method of reducing waste brought up by five of the eight participants. In some cases the strategies were about keeping it simple, for example, B1 discussed that they try to use lightweight tools and that do not require any steep learning curve. B2 discussed that the use of processes for the sake of it should be avoided, such as do not use Scrum if it is not needed. C3 talked about how processes easily grow over time as they get added to all the time. Sometimes it is needed to take a look at the processes and see if they add value. C3 also mentioned in the past that many processes took place because somebody in management thought they were a good idea without thinking of the overhead costs.

The creation of the MVP was approached in different ways. B1 told that being in close cooperation with the customer is their way of avoiding unnecessary development work. In Company C, technologies are evaluated before they are developed further and the use of quick prototyping is also a practice used. In Company D, the opinion of what is waste differed in this respect. They do throw away features that they decide are bad but it is not seen as waste because all involved learned more about the technology.

Good specification is the method used in Company E, but this can be attributed to the nature of the business. Bad specification could be quite costly in the end products that are manufactured. C3 also mentioned that good specification would help reduce waste be did not feel that the current level of specification quality is optimal.

Meeting avoidance also trends in Company E where both participants spoke about avoiding meeting as far as possible or keeping them short. E1 feels that if really needed, phone calls can go a long way to save people's time.

Any many ways, focusing on tasks is similar to avoiding processes or meetings. B2 feels that adapting a working style that focuses on the work, not meetings is important. C1 said that at some point you need to ensure that you are spending enough time with the real deal.

Only one person mentioned automation, which cannot be classed as a theme but is included here simply because it is an important strategy because it follows the principle of Jidoka, which is the concept that humans and machines can detect faults.

5.2 Why have such project management methodologies been chosen?

This section tries to discover if the project teams have freely chosen their working methods or are they chosen by some external entity. It also tries to understand if the methods in use have been matched to the project type, or are they simply chosen as some default. This is looked at from the point of view of process constraints, which directly asks how the process was chosen and how the organization in which the project exists supports the projects, which also might have some influence.

5.2.1 Process constraints

When trying to evaluate the process constraints it was only looked at from the point of view of who chose the process.

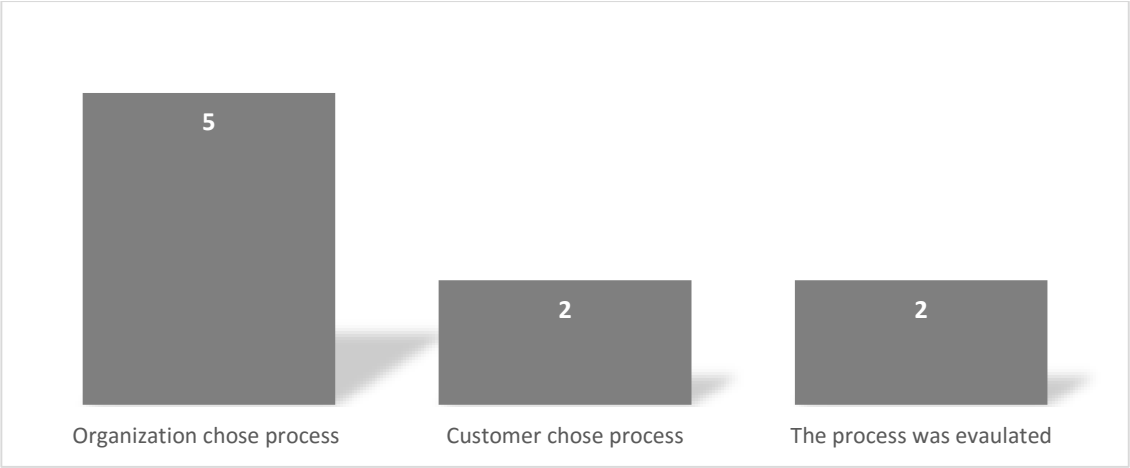


FIGURE 21. Process constraints.

In Companies A, C and E the process is chosen by the organization. A1 stated that '*it is just the way we do things in the company*' and C1 stating that '*I didn't choose it, I got the impression it was chosen*'. It is not clear if C1 could actually choose if they wanted to. C3 discussed that they use Scrum because it is the most well-known of the frameworks and also it is adapted from the company way of doing things. In Company B, there are apparently different ways of executing projects but it is not clear what the method to choose is. In Company E, the process is chosen by the organization but E1 did in fact state that, as an organization, they did earlier evaluate different process and chose the one they use because it is the best fit for their project types.

The participants form Companies A and B also told that the customer may provide some constraints about what process is used. This may well be the definitive answer to the question how Company B chooses when they have different methods as stated in the last paragraph.

5.2.2 Organizational support

Organizations support projects in many ways from providing resources, guidance as well as learning and career opportunities to its people. In this section, it is discussed how the project managers see the organization and what kind of support is provided.

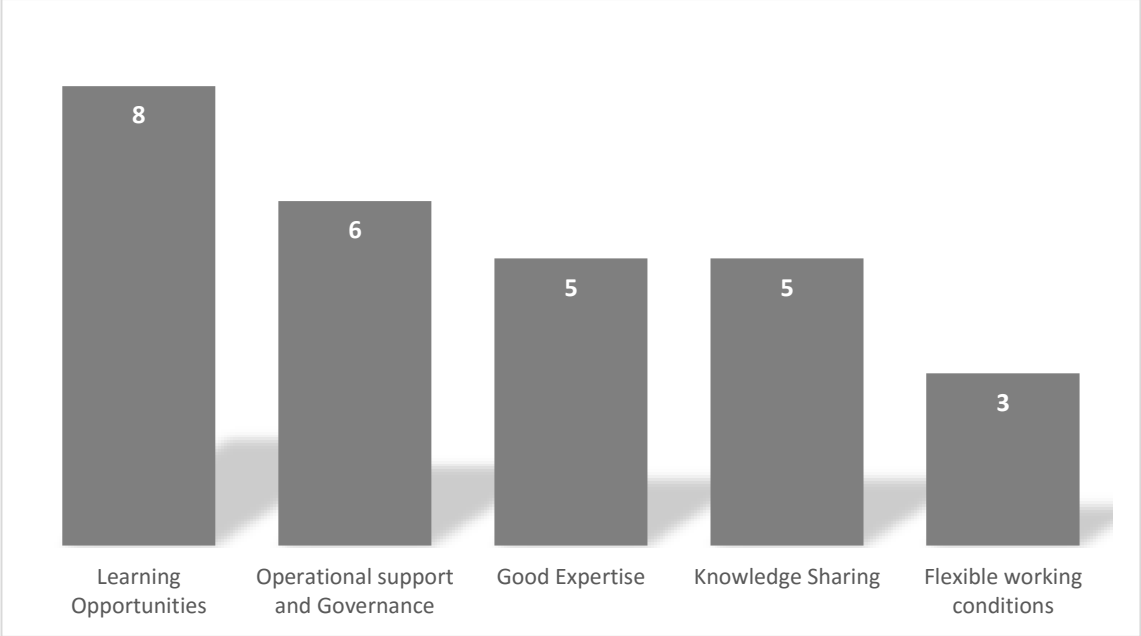


FIGURE 22. How organizations support projects?

The ‘Operational support and governance’ trends in the bigger Companies C and E, according to the data. These companies obviously have the resource to oversee projects in a more profound manner but support manifests in other ways too. For example, A1 said that they get support by having the freedom to do the right thing themselves and they know the management will support them in that. In Company C, they get multiple types of support from providing processes as well as training to gain the needed expert knowledge. The management also gives project steering via a panel of experts in various important fields. In Company E, support is given via means of having experts in the organization that tasks can be delegated to, such as purchasing and logistics.

In five of the companies there was high value placed upon having people around in the organization whose skills could be drawn upon in time of need, i.e. good expertise. Comments such as ‘*a lot of knowledgeable people here*’ and ‘*experts with different points of view*’ set the theme.

On the subject of having experts comes the method by which the knowledge they have is shared. In Company B, for example, they have Friday talk sessions where an expert will do a common room presentation about their area of expertise or teams can tell about their best practices. Company E gave another example of when the project practices were being defined, they had workshops where all the people could give their input about their experiences.

In Companies A and B, flexible work appears to be important. A1 commented '*we get a lot of freedom to do whatever we need to make the customer happy*'. B2 told that they do not really have mandated working hours or ways of working, just as long as they get the job done.

On the subject of the conspicuously missed item of learning opportunities, the kind of learning opportunities were looked into a little deeper.

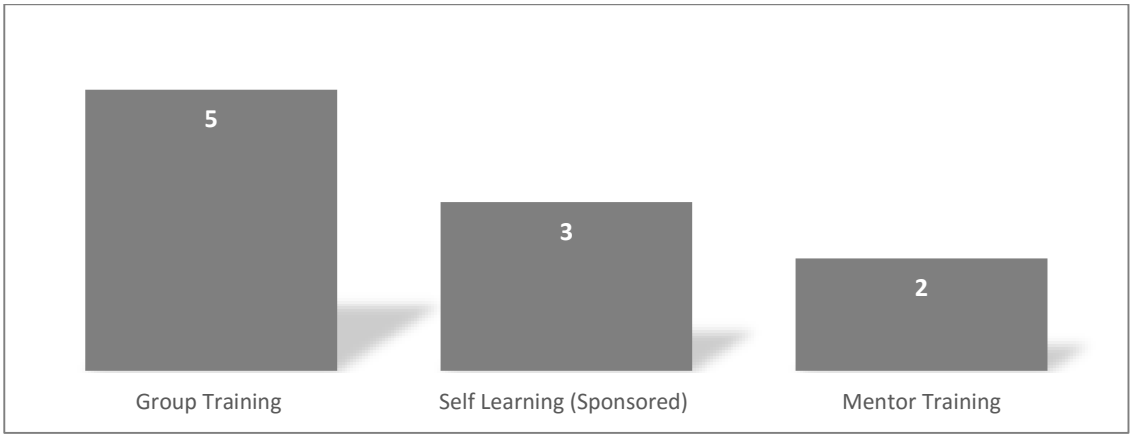


FIGURE 23. Types of training

Group training was evident in Companies B, C, D and E. Mostly in the form of classroom events and presentations which provide information on subjects of concern to the company. The most interesting was the game company. The personnel from there also provide community recompense by teaching a game engines course as TAMK. Self-directed learning is popular in Companies A and B, where people are given time to study things that interest them or that are relevant. The curveball is thrown by Company A, who actually pay a development bonus to those who develop open source software in their free time. Mentor training also takes place in Companies A and B as described earlier in expert knowledge sharing. People who are experts in a certain field are expected to give trainings to others, especially newcomers.

5.3 What constraints are projects subjected to by their organizations or customers?

When looking at the constraints of project a few dimensions were investigated in the interviews. Projects are constrained from within like discussed in section 3.2. – that is scope time, cost and quality. The second point of view is how the teams keep their

customers satisfied. It is not clear from the data if this dimension actually constrains the teams in any way but it certainly has impact on working behaviours. Last, a short insight into to virtual teams is given. That is do the teams have to work in a virtual environment and whether it impacts the team performance.

5.3.1 Project constraints

Time is constrained mostly by deadlines but it only showed up in Companies B and C as a major factor. In all these cases, the form of the constraint was a deadline in which the project has to be completed and the team would probably adjust scope to meet the deadline.

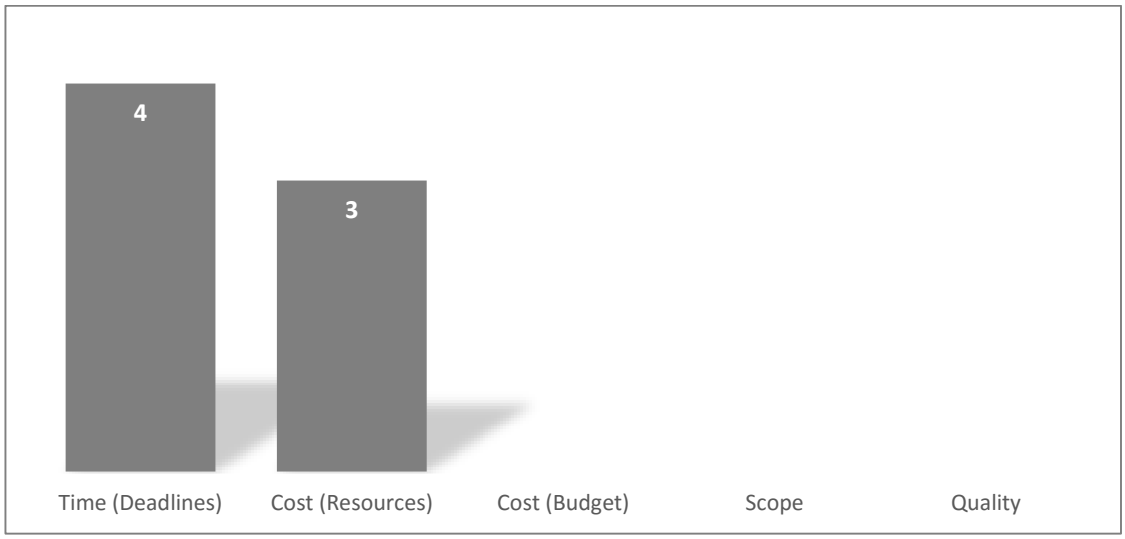


FIGURE 24. Project Constraints.

Only the people in Company C spoke of a second restraint, which is cost in the guise of resources or, to be more specific, headcount. This, however appeared to be a bigger issue from the participants because they all spoke of it more thoroughly. All the three participants in Company C stated quite clearly that there are rarely enough resources to carry out a project and one said that in some cases resources are removed as the project goes on due to higher priorities of other projects. So it appears that the organization governs who is in a project, not the project itself.

None of the participants discussed the budget and three of the participants actually stated specifically that they do not track any kind of budget in their projects. The scope and

quality of the projects appears to have the same fate, with nobody stating what level of quality is required from the project.

5.3.2 Customer satisfaction

Soliciting feedback was a trend that showed up in Company C, which is interesting because there is no direct contact between the team and its customers. Feedback is collected via forums, forums and the service department. A similar tactic is used with the gaming Company D who make pre-releases to a trusted community who provide feedback about the game.

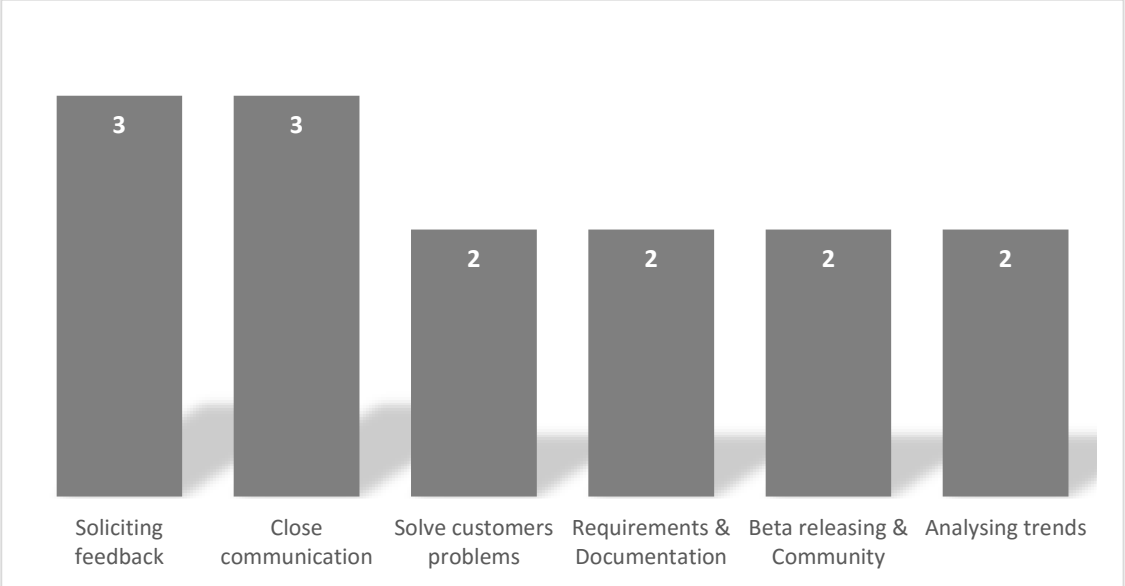


FIGURE 25. Customer satisfaction techniques.

Close communication is favoured by Company B, who like to get closely involved with the customers. B2 said that it is good to visit and find the pain point of the customer and find way to alleviate them. C3 also highlighted something in this area, which is trying to meet customer expectations. This is probably achieved through the above-mentioned forums.

Solving the customer's problems is very closely related to close communication in the sense that communicating and finding out what is needed helps to solve the problems. B2 like to ensure that everybody is on the same page and E2 solves feels that it is his key responsibility to ensure that the customer gets what was agreed.

Good documentation trended in Company E probably as a result of the need to ensure that specifications are accurate. Documentation is used as a tool for communicating what the customer will get and also what the customer is responsible for doing themselves. According to E2, it is also a way of ensuring that the expectations are met.

As already mentioned, both Companies C and D make pre-releases of their software to communities. They look for trends, for example, C1 said that there are many active users who like to test the latest applications and if several hundred people are saying that we need a feature, we listen to them. Apparently through the gaming network used by Company D, gaming enthusiasts may even buy pre-releases of a game and also give valuable feedback.

As told earlier, there are product owners in Company C that track the trends in the industry in which they operate. They also perform competitor analyses to find out what features the customers like in competitor products.

5.3.3 Virtual teams

In Companies A, C and D the teams were mostly co-located in the same office with only E2 stating that the team was distributed also in the customer locations. This is not so clean-cut, though, because it appears that in many of the teams there are contributors from other sites. In Company C, it appeared to be stakeholders that they were dependent upon, such as other teams and their deliveries, not actual team members. Company D had one person from another city that had a specialist skill. Although not clear from the data collected, Company B also appeared to have everybody located in the same place unless they were placed in a customer’s premises.

FIGURE 26. How the teams are distributed?

It was also possible in some cases to determine from the data if having distributed teams actually causes difficulties.

In the cases that it is difficult, the main issue is time zone difference. E2 who has people on the customer site told that the time difference is so big that there are a lot of out of working hours meeting that affect free time.

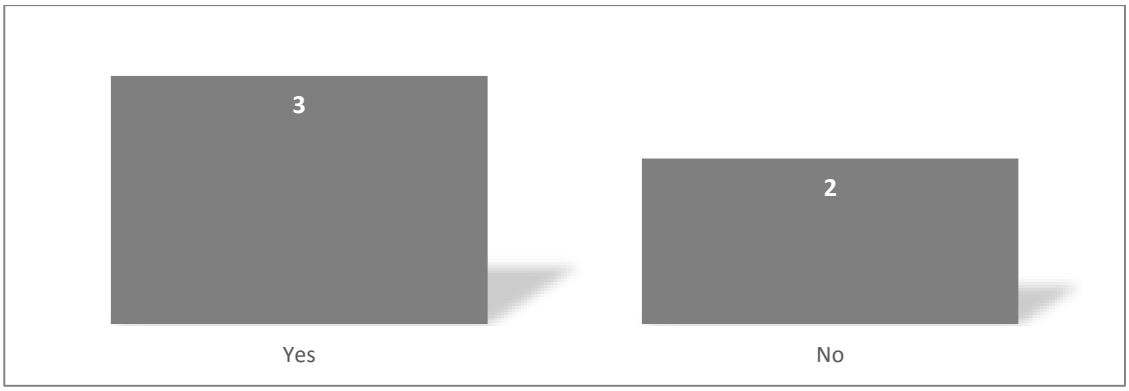


FIGURE 27. Does a distributed team cause problems?

In the case where it was not deemed a problem – for example, C1 thinks the time difference is an advantage because things get done overnight. C2 simply felt that because there is no managerial relationship it is just a communication role and not really a problem.

5.4 What do project managers see as the ideal environment to manage projects in?

To round off this research and taking it beyond current state analyses, it is an investigation as to ‘what could be’. This is looked at from three points of view. First, it is to discuss what motivates project managers. The reason for this is because in modern leadership it is well understood that motivated people perform usually beyond expectations. Then, it is to delve into past experiences of the interviewees in order to find out if they were ever on projects that went exceptionally well and what they think was the key reason for the success. The last part is the Dream phase and interviewees were asked to think about the ‘perfect world’, which seems to only exist in project management textbooks.

5.4.1 What motivates project managers?

The question of what the interviewees enjoy about project management was asked at the beginning of every interview. The question was asked at the start to relax the interviewee before the heavier questions were asked, but the answer contributed to the Dream phase.

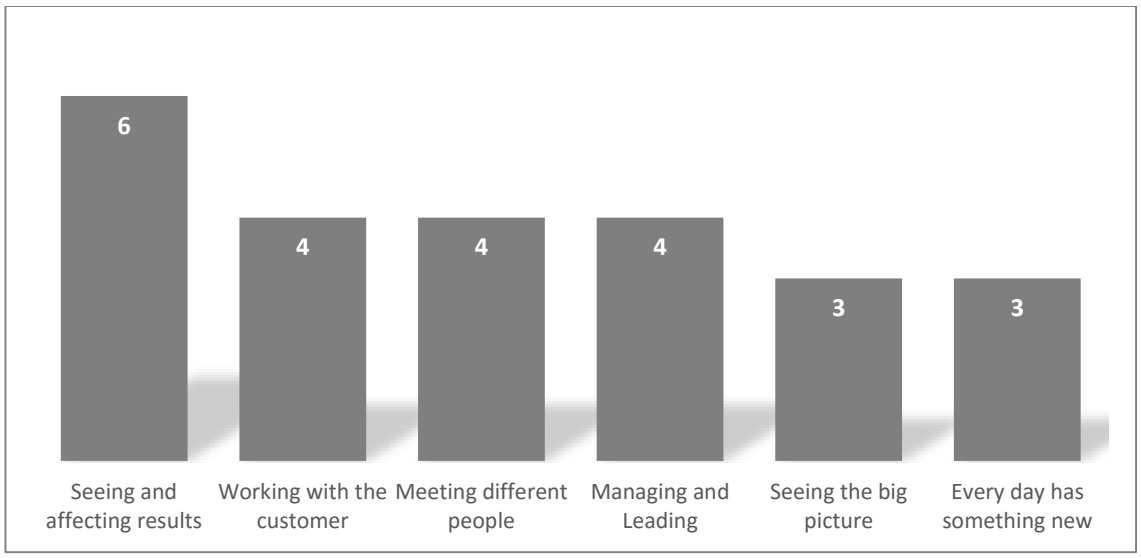


FIGURE 28. Motivational factors for project managers.

The most common theme among the respondents was seeing and affecting the results of their work. This was not the most obvious theme to be recognized since it was expressed in very different manners during the interview. It was also not the most significantly talked about motivator among the people who mentioned it; in many cases it was mentioned at the end of the answer or in passing when discussing other factors.

A1 expressed this as having wishes taken into account during the project and B1 expressed this through being able to guarantee the success of the project as part of the discussion about working with the customer. C1 likes to do new things all the time. This is perhaps rather tenuous categorization but it can be assumed that having different activities is a results-based mind-set and was therefore it is included. C3 was squarely in this theme by making clear statements about being able to affect the product and seeing the results and being able to say '*that this is my work*'. D1 also clearly stated that seeing the results of the work and putting all together was a key thing for him. E2 also likes to get some big things done and also mentioned the tenuous category of having something new every day.

The second and third strongest themes were working with the customer and meeting different people. During analyses these two items were separated but it could be argued that the line between these two is very fine because meeting the customer is in many cases can be classed at meeting new people. If these two categories are merged, it in fact becomes the strongest theme in the answers. This also coincided with the fact that this was clearly the motivator which the respondents expressed enthusiastically when discussing their motivations. It was often the first thing mentioned or the most significant part of the motivation discussion by those who discussed it.

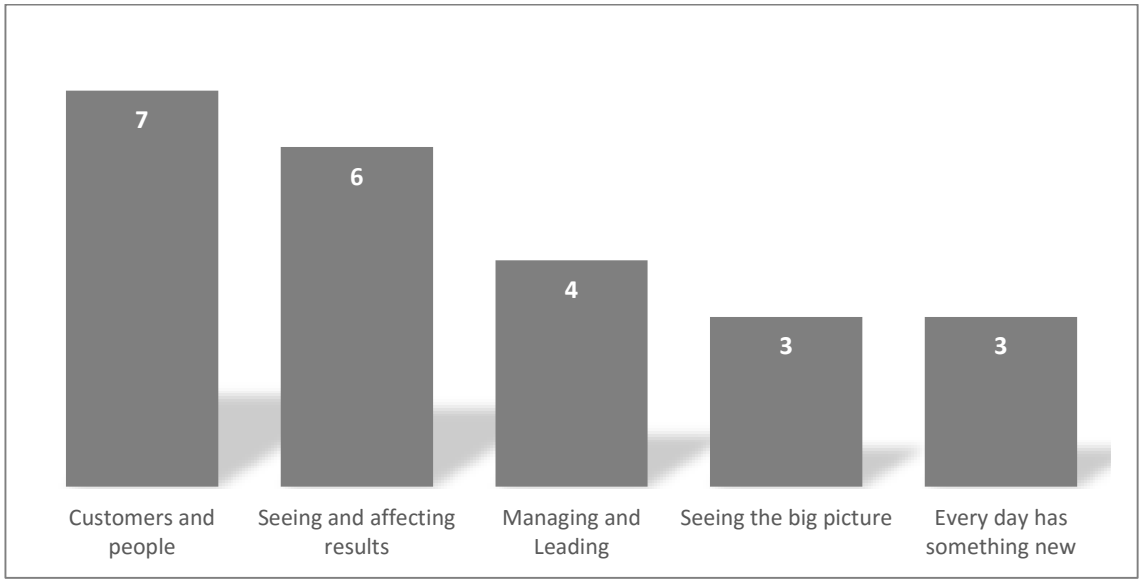


FIGURE 29. Motivational factors with working with customer and meeting people combined.

For A1, discussing with the customer and leading the customer relationship was high on the agenda. B1 likes having a close relationship with the customer and talking with a lot of different people a statement closely made by B2 as well. C1 had meeting of all kinds of people as their primary motivator. C2 likes to communicate and collaborate with different people and is quite effective at that. In the games company, D1 likes to get people and technologies together. This is with respect to the wider business of the person in testing different technologies for entertainment purposes. E1 likes working with customers and get pleasure from solving their problems.

Managing and leading was a less prominent theme brought up by less than half of the interviewees. A1 likes leading the content of the project and trying to meet the career development needs of his team. C2 being a technical lead likes to put himself in the shoes of his development team when managing. D1 likes organizing, specifically the system. It is not clear if this was technical organization or management of the whole solution, including people. E2 clearly stated that he likes being in charge and managing people.

The last of the themes are around the big picture manifested as statements about sharing knowledge and seeing the big picture. Three of the respondents valued having different tasks every day and being able to contribute in different ways; coupled with the freedom that often comes along with the role of Project Manager to fulfil their day, rounds up the key motivators.

5.4.2 Why do successful projects succeed?

This question was to designed to prompt the interviewee into thinking about the positive things in project management and act as a precursor to the later question of the ideal project environment.

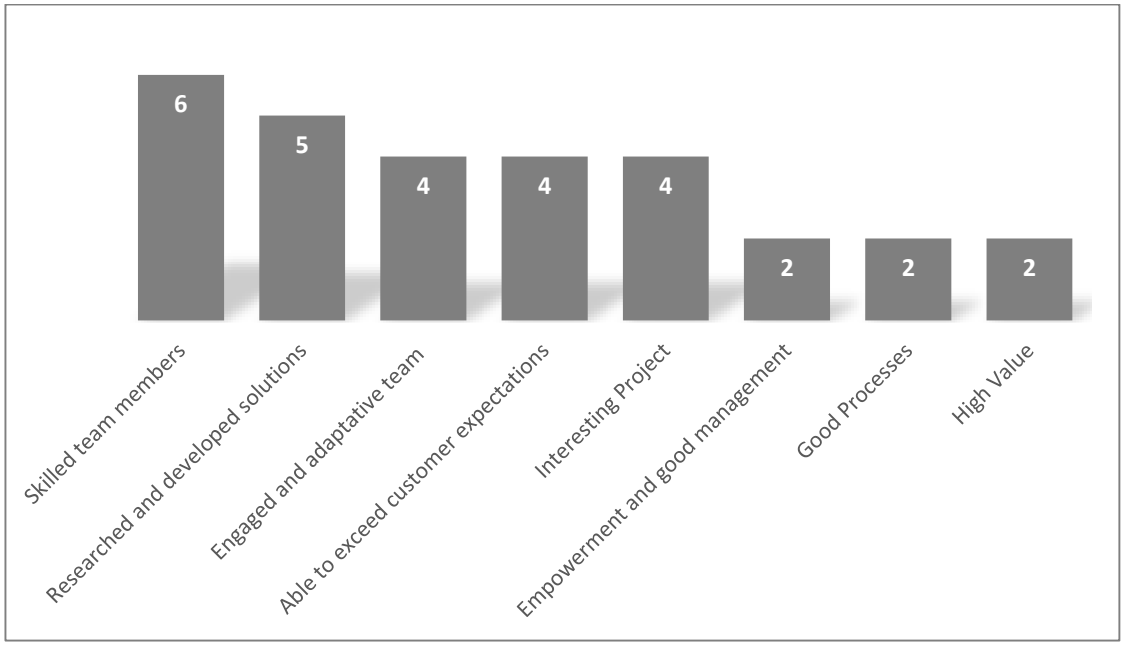


FIGURE 30. Why projects went well?

Six of the participants felt that having skilled and expert team members was an important factor in making the project that they spoke of successful. B2 told that the team in question had skills that complemented each other. C1 discussed having people that cared and who understood the technology to get the job done helped. This sentiment was also carried by C3 who said that the team was dynamic and able to adapt to different tasks. C2 felt that the team had a deep understanding of the technology. D1 said that he needed the long experience of the developers in order to make the project in question successful. Having highly trained people was how E1 made a successful project, which is a slightly different sentiment to experience, indicating that the job in question was perhaps more mechanical.

Allowing and trusting the team to do research independently, and solve the problem directly was also a good success factor, i.e. taking care of the whole solution. A1 told of a project where the team were asked to research the solution to a problem that the customer had no idea how to solve. The team in question came up with an end-to-end solution which solved the customer dilemma very neatly. This theme continued in the discussions with B1, B2 and C2 where the teams were left to their own devices and came up with a solution. D1 explained a situation where he was told that he could do whatever was needed to get the solution working, a kind of empowerment.

Having a team that is engaged and is made up of motivated people was also brought up in many of the discussions on the subject. B1 told quite directly that '*what made it succeed was commitment to the project*' and that they wanted to show what they could do. B2 told of the team working very closely together led to success. C1 and C3 also spoke of highly engaged people who could adapt as necessary to get tasks done.

There was a clear theme that exceeding the expectations of the customer was seen as being a successful project. This was raised all who spoke of the subject. The only extra comment was not using the whole budget, which was cited by B1.

Working with a topic the interested the team members was also raised. In a continuation of A1's story, he felt that find the solution was interesting for the team and raised the motivation level to succeed. B1 mentioned directly '*that it was a really interesting topic*' and C2 was telling that he had high-level knowledge to start with that he was able to deepen.

Empowerment was raised by two of the participants during the interview. C3 explained that it can be managed chaos and being empowered to manage that as seen fit helped a lot. As it was already mentioned earlier, D1 was clearly told by his management to do whatever was needed and it was felt that this freedom brought about success.

Good processes was brought up by both participants in Company E. E1 felt that if everybody works as expected the project goes well. E2 told that having mature processes lets everybody know how they should work and bring about success.

Working on a high-value project was a factor for B1 and C2, not necessarily high monetary value but something that will be useful to customers or the company. In other words, the feeling of being involved with something important.

5.4.3 What would be an ideal project environment?

This is the question that the participants were approached with in the Dream phase of the appreciative inquiry. The idea of the question was to draw out from the project managers, their opinions about what could be improved in their environment.

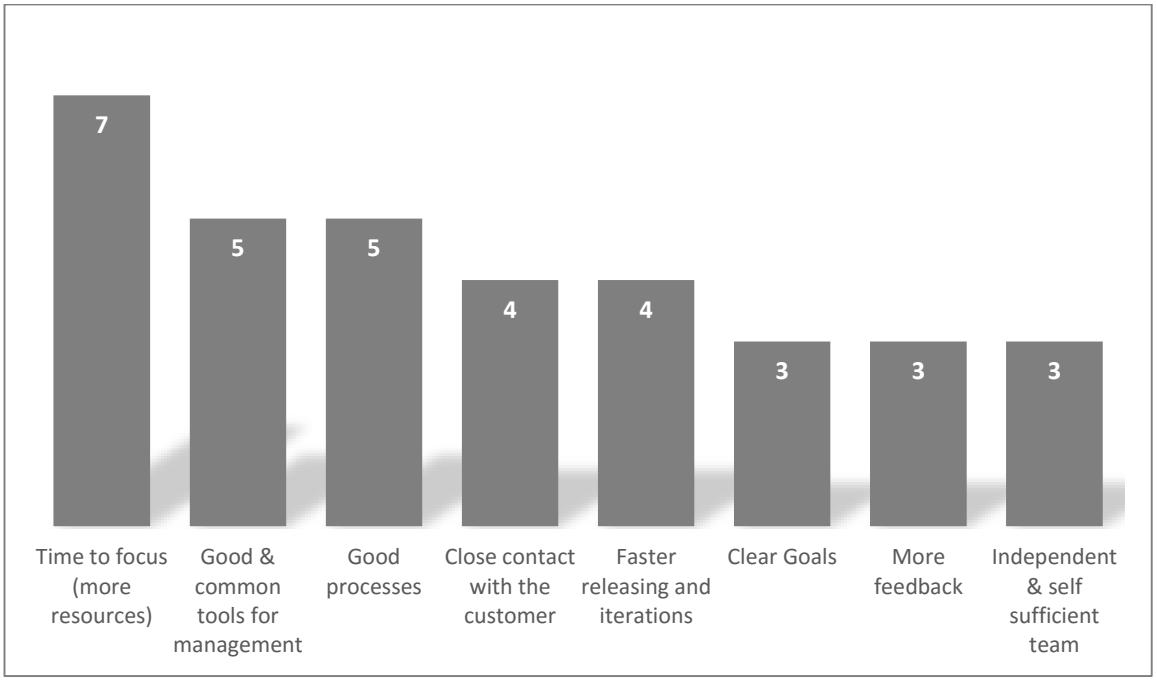


FIGURE 31. The ideal project environment.

A large number of the participants expressed the lack of time and resources to get project completed well. The participants from Companies C, D and E told how they would like to have enough people who would be dedicated to the project. There were also comments made about how the team are constantly distracted with other things. B2 made a suggestion that during the mornings people should be allowed to focus on their work and the afternoons would be better for meetings and other disturbances. C1 would appreciate time to read reports and other inputs, such as feedback, so that it could be interpreted properly, enabling the right actions to be taken. C2 and C3 would appreciate time to look into the technology properly and the ability to concentrate on one feature at a time.

Good tools for tracking and managing the project were also high on the list of demands. A1, C1 and E1 would like to see a good tool in place that enables accurate tracking of the project. In Company E, both participants would like to see the harmonization of tools

across organization to allow people to get the needed information easily. Backlog management cropped up in as well in a couple of cases.

In contrast to the findings in the section about Lean and waste, process are seen as important. B1 would like that the organization should provide good ways of working but would like to have different parameters depending on the type and size of the project. C1 would like some kind of PMO office to help with the running of projects and in Company E where processes are well established there is still a need to ensure that they are Lean.

Closer customer cooperation is something that several aspired too. B1 wants the customer to be involved and interested and able to respond faster to queries. E2 described the need to have a lot of trust between himself and the customer to have ideal working conditions.

In the software project the participants would really like to shorten the release cycle. Example to achieve this was an automated releasing mechanism which would allow smaller increments to be released. C3 expressed that the minimum viable product should be properly understood and released as soon as possible and then build new features on top. E2 expressed that the ways of working can be heavy and it would be nice to reduce the time by finding the optimal level of processes, for example.

Clear goals are simply having a good understanding of what is needed from the customer. In Company B both participants would prefer that the customers are able to elicit their needs more clearly and as earlier mentioned even to know their needs in the first place. In Company E the need for better specification and scoping projects better was desirable.

B1 wanted an environment that embraces feedback, which is a surprising statement considering the highly Agile nature of the company. B2 would like better transparency within the organization, so that all could see what is happening. C1 wanted more channels to get feedback – this probably originated from the lack of direct contact with the customer.

Independence was discussed in many ways. A1 wanted that people could more freely choose which project they are working on. C3 was of the mind-set that all contributors should work inside the team and that the team would take full responsibly for the product and figure out what is needed.

6 CONCLUSIONS

In this chapter the questions are returned to and interpretation of the key points in the discussion chapter are raised. Not all points raised in the discussion section are raised here because the data is either self-evident or there is not much to interpret.

6.1 How projects are managed in a selection of Finnish companies?

Originally, the goal of this research question was to identify what processes teams are using to execute project. This proved hard to achieve because the discussions were not structured enough to reveal this information. Using hindsight, it may have been wiser to ask the participants to prepare by writing down their execution processes. The interviews did reveal a lot about the activities that are taking place and what kind people are taking part in the projects. It was, however, clear that none of the participants described textbook processes, but more like mixed and matched ways of working as discussed in the introduction. This can be seen as a good thing because it shows that people are thinking for themselves and adapting according to the environment. In many cases the impression that projects are being managed with minimum of effort was evident. This could lead to projects not running as smoothly since important steps such as good risk management and planning are skipped, in order to get started on the development work. Next, the main discussion points of section 5.1 are concluded.

Roles

If it is assumed that most of the software team in this study are practicing some kind of Agile such as Scrum, XP and Kanban; there are a lot more roles identified than expected. Referring back to section 3.6 a typical Scrum team consists of a Scrum Master, Product Owner and the ‘equal’ development team.

As already mentioned, the role of Scrum Master, where it exists, appears to be carried out by a senior member of the organization like a Project Manager. It would be good to find a well-written guide on how the roles of Project Managers and Scrum Masters can be separated. Agile is very naïve in the sense that it expects Scrum Masters act only as a servant in the team. Projects need control at some level for many reasons, such as resource management and financial control, i.e. both roles need to be filled.

It also appears that most teams have some kind of Lead Developer, which it is assumed to be a more senior person as well. This evidence from the data starts to look like a hierarchy in the teams. While there is no evidence to show that having more senior members is counterproductive, it is hard to determine how a team can be 'self-organizing' and having a 'team competence' when some individuals are having a stronger weight in decision-making. The Agile manifesto clearly states that the best architectures and designs emerge from self-organizing teams. It could also be questioned whether or not the presence of external Software Architects exacerbates the situation.

The Product Owners, where they existed, did not appear to fit the description from Agile handbooks. They were classified as the business owner, and the person who communicates with the outside world and the person who gives the overall concept. There is no mention of writing user stories, which begs the question: 'Who is writing the user stories in the Backlog and how they are being accepted?'

The other roles mentioned are expected to be found in most teams, such as testing personnel and user interface designers. It was not clear whether or not these people were sitting within teams are they part of other organizations, which concentrate on those tasks. If they sit outside the team, there is a high risk that those principles of Lean thinking are forgotten. Examples of this could be creating specifications that are not used, and software progressing too far before it is reviewed and needs discarding.

Responsibilities

As mentioned the number of themes was low and it is a well-known fact that the tasks that project managers have to carry out in a project are very broad and sometimes random. As one participant mentioned, they basically do everything that the other members of the team do not do.

Resource management is a critical thing in any organization and to have the right people in the project at the right time is of utmost importance. Interesting aspects that appeared here were, for example, the active movement of people between teams to promote learning. Also, project managers need to find people that have specialist skills, is a particular challenge. It would be interesting to see how this can be achieved. One idea

that the author has come across in the past is people's skills and knowledge being included in company phonebooks.

The only other highlight here is the revelation that the project lead might be the person writing the user stories. There were hints of this in some conversations at least. Good user stories form the foundation of having a reliable source of specification and a way to verify that the specification is met. There is a danger that user stories written by project leads risk the quality because of two reasons. The first is that the project leader is conflicted in accepting stories at the end of the Sprints because he is biased towards moving the project forward. The second is that somebody who does not dedicate significant time and effort is likely to write poor stories, leading to the classic problem of not having clear criteria (goals) and therefore lack of true understanding if the criteria was met. If this is a fact, and something that is common in teams, it would be interesting to research how the quality of the product is ensured when the acceptance criteria is defined by an involved party.

Initiation and planning

Project initiation in the software industry has changed dramatically from old school thinking. The contrast can be seen here between the R&D projects of Companies A to C, and the delivery projects of Company E. Company E is following old school thinking having a clear sales phase ,and then implementation of what is sold. This is a suitable process for that company due to the nature of their projects.

As expected, the initiation and planning of R&D projects is more likened to a clamber or more aptly named Scrum to understand what is actually needed by both the customer and the team contracted to make the change. It is very positive that although some un-clarity exists – it is a widely recognized fact that the scope of the project needs to be understood before proceeding. It appears to be done early in the project lifecycle by the majority of project managers. How tightly the scope is controlled after this point is not clear in this research and it would be good future research to understand how scope creep impacts the project.

The trend among the most participants is that they draw upon experts from outside the team during this phase to assist in clarifying what is needed. For example, many of the

companies recognize the need for user interface professionals to give input and design. These inputs are to bring cognitive thinking and human appeal to the software. The phase heavily involves meetings and discussions to bring many ideas to the table as to how the problems can be solved.

The use of prototyping only appeared to be in in clear use in two of the companies, although it should not be forgotten that other companies do use a ‘deliver fast and often strategy’, which is similar to prototyping. In section 3.5, the Lean start-up was introduced and it was stated ‘plans rarely survive the first contact with a customer’ and ‘the faster you deliver something to the customer and get feedback, the lower the chances of project failure’. Prototyping is the implementation of this thinking. In the author’s experience, when customers and even those who specify the system see the actual software in action, they realize that it is not really what they wanted and ask for it to be changed.

Project execution

Among the participants, the task of general management seemed to be more like ‘firefighting’ that systematic management. In other words, dealing with issues as they arise – also known as reactive management. This is probably a result of the evidence that there is either very poor or no risk management in the projects. If proper risk analyses were carried out at the start of the project, most of the issues that arise could have been changed to planned tasks that were expected. It also assists in understanding the real size of the project and would make for informed decisions at the project go/no-go reviews.

Milestones appear only to be the method of large companies only. This is probably for historical reasons but it would be interesting to understand how the companies that do not have milestones really identify whether or not the project is complete. Agile projects have a possibility to go on if nobody identifies the minimum viable product because the Backlog in a truly Agile team will probably always contain non-mandatory items.

It is interesting to see that effort estimation appears to have lost its value – at least in the software projects – so organizations have drifted towards the #noestimates way of thinking. The fast delivery cycle in those smaller companies has probably eliminated the need for estimations as Zuill described in his blog.

There appeared to be very little evidence of the management of Backlogs in the teams, even the ones practicing Agile. Whether it was not done or it was simply forgotten to be mentioned will remain unclear. In either case, it seemed to have very little emphasis. If the Backlog is not constantly maintained and items remain similar to the start of the project it could be argued that the project is actually being executed as a Waterfall model.

Based on the evidence, Continuous Improvement appears to have gone out of fashion in a big way. Company A understands the value of retrospectives by enforcing them. However, it would be interesting to know why the others do not see the value of retrospectives. Perhaps it just feels like bureaucracy or a waste of time. Does this mean that teams simply continue working in the same way?

Even though most of the team are not practising pure Scrum or other Agile methodologies, most have adapted the daily stand-up or Scrum meeting as practice. This is obviously a very important forum for the team members to understand the situation in the project and help each other. Maybe another point of view could be that the ritual of the Scrum meeting makes the team feel like they are being Agile, even though this might not be the case.

Lean management and waste

When discussing waste, none of the participants appeared to bring up the content of the TPS handbook or discuss Lean management techniques from it. Process avoidance and in the same category meeting avoidance is quite the opposite of what the TPS promotes. In TPS having everything on hand only when needed can only be achieved with thoroughly thought through processes. Perhaps the point being made by the participants is that very often processes appear to just be effort overhead and does not add value to the end product in any way. A good thought process for anybody creating a process should be asking the question of what value it brings and also quantifying it.

Prototyping and evaluation is a good example of Reis's Build-Measure-Learn feedback loop and in line with Lean start-up thinking. Building the technology quickly for evaluation and pivoting (throwing away bad design) is a good example of failing fast.

Being given space to focus on delivering the product is a good example of how the modern working environment can easily become unproductive, as people have to multitask. Context switching is understood to be rife in larger companies and is disruptive. Could this also be a good example of where agility is weak? Could it be that having to stop for daily Scrums or fixing build breaks actually disrupts thought processes and slows down development? It would be a phenomenon that could warrant further research.

6.2 Why have such project management methodologies been chosen?

This question did not get answered in the detail that was wished for. Although a good majority stated that the organization had chosen the processes. There was no real insight into why the organization had chosen them and this extended to the customer decisions too. It was not found out whether or not was evaluated as being the best process for the project type and circumstances. In most cases, the interviewee did not know and just accepted the fact.

It is quite normal that an organization chooses what process a project uses for various reasons. The most common is that it is easier to oversee the projects if they all follow the same steps. It should be noted here that the process or methodology does not dictate everything and that teams are quite free to work as they see fit within the frameworks, and use the best practices. As we know, in Agile the retrospective is seen as critical and therefore there must be room for own best practices within the team.

Organizational support can mostly be summed up as project and teams are not being left to fend for themselves and work out everything alone. The overall theme is that organisations support projects and people by providing knowledge and guidance on how to execute projects and provide expert input on a demand basis. The subject of experts and knowledge was seen as a very positive thing by the participants. Just as no person can have all the answers, the same axiom can be extended to teams or even whole companies, thus the prevalence of consultants in the industry.

Another positive insight was that all the companies appeared to value investment in their staff. This mostly manifests through on the job training and differs in each company. The bigger firms tend to offer more classroom or structured learning opportunities. The smaller firms were decisively more ingenious in providing learning opportunities through

using in-house skills to bring the level of knowledge in the organization up. Company A has also apparently seen past the risk of losing intellectual property by asking its employees to contribute to open source projects. Another possible angle of research, is if the knowledge gained with the open source development yields better results for Company A than the financial gain of licensing the lost in intellectual property.

6.3 Which constraints are projects subjected to by their organizations or customers?

Based on the findings, it can be seen that at least in the case of the software projects, they are hardly constrained at all. As already discussed in section 3.2, lack of constraints can lead to Yourdon's (1997) Death March projects. It is not clear from the data whether or not projects are in such as state but it would be interesting to see how many projects are successful under these circumstances.

Since deadlines are the most solid constraint in the projects, it appears that the other three dimensions of scope, cost and quality are left floating. In Company C the costs are being limited by means of the number of people on a project indicating that the scope is the dimension that is bound to change. This matches quite well with the principle of DSDM thinking, which was discussed in section 3.6.6 – that time and costs should be fixed and the functionality can be adjusted based on the 80/20 rule. It would be interesting to study further if deadlines really are deadlines in these cases, or are they actually allowed to slip as well, leaving all three dimensions floating.

The quality of software projects is quite hard to define. In the author's experience, it is usually dictated by the number of release blocking and serious errors which exist. Meaning that the software will not be released with errors which will disturb the user in normal operation.

Keeping the customers satisfied is probably one of the most important ways of staying competitive and the different strategies used here show that not only one solution works. A good way to find out what the customer wants is to ask. This is the shortfall that Reis highlighted in Lean start-ups, that 'business plans rarely survive the first contact with a customer'. Both asking the customer what they actually need and doing beta releases to the potential customer alleviates this problem. Assuming that the teams have the courage

to pivot where necessary, as well as building on positive feedback follows Reis's guidance.

When being in direct contact with the customer, close communication builds trust, which in turn creates an environment where the customer feels valued and that they will get what they need. In that relationship of trust it will be easier to discover what the customer's pain points are and being able to sell them what they actually need.

Using requirement documents in the software industry has been found to be non-effective. Very often the customer of software projects is not always clear what it is that they really need. This is one of the reasons Agile came about, to adapt to that changing need of customers. Documentation suits some environments, such as the one that Company E works in, where the products are fairly well predefined and need only to be delivered. If heavy requirement documents and contracts are needed, then the appropriate execution methodology needs to be selected.

Analysing trends differs from the other methods of understanding the customer because no feedback is asked. The requirements are based on looking at competitors' products and following the ideas. This can be little flawed in the sense that it assumes that the competitors are getting it right but often forums will give clues to that question. This is somewhat a red ocean strategy and will not necessarily lead to competitiveness as discussed in Kim and Mauborgne's article 'Red Ocean Traps' (2015).

6.4 What project managers see as the ideal environment to manage projects?

The ideal or perfect environment for projects is obviously some kind of Holy Grail. Through continuous improvement the working lives of people can be enhanced bringing about higher motivation and engagement. None of the findings here are particularly new and, in fact, most are quite reasonable – and therefore reachable. This section can almost be read as a general 'lessons learned' – also known as a retrospective – that can be used as input at the project initiation phase. Perhaps a project charter would be a suitable place to add some goals that the team would be interested in. Next, we will look quite directly at each of the three questions again.

What motivates a project manager?

They main conclusion that can be drawn from this question is that project managers clearly like meeting people and working with their customers. It was the clear item of discussion among the respondents and can be taken as a highly valued part of the work. Those who do not have direct customer contact found a way of virtual customer contact through online feedback and other forums. What is behind this is probably the subject of psychology, and not in scope of this thesis, but it is clear that ensuring that the project managers have some kind of customer contact, will improve the perception that they are in a good job. Obviously, filling the role with people that are biased towards extroversion and good social intelligence is probably important considering the number and different types of people that they must deal with.

Being able to see and affect results was also clearly high on their wish list. This is probably the ability to be able to have some say in the outcome of a project and what the end product will be. On the contrary, perhaps managing projects that are specified too accurately and written in stone could probably quickly lead to the person seeking other challenges.

Surprisingly, managing and leading was not the key motivator for the people in the sample pool. All but one of the people were Finnish, so maybe this is a cultural issue. About half have the need to manage and organize and thus would probably show up as strong preference in a personality test such as a Reiss Motivation profile. That would be a possible field of research to analyse the profiles of people in charge of projects.

It is difficult to comment of the conclusions of the other themes, since so few had the preference for them. In this case, it is probably best left to the readers' own interpretation.

Why do successful projects succeed?

The discussion about successful projects brought up many interesting dimensions to what makes a project tick. Clearly, working with skilled people that know what they are doing helps and makes life easier. Getting the team engaged is another factor all together. Having an interesting or high value subject obviously lights the fire that brings about engagement.

It can also be read that empowerment and working in an R&D environment are a good combination. In those projects where the team were empowered to find the solution themselves they appeared to excel. Having full ownership of the solution obviously works to get the project working well.

According to the PMP Exam Prep book (Mulcahy 2013, 18) the 'PMI does not approve of gold plating (adding extra functionality)' to projects. It is interesting to see here that project managers generally feel differently. They like to exceed the expectations of the customer. Whether this is gold plating or not remains in question, but statements made such as '*doing more than they asked for*' and '*things they did not expect*' indicates so. Perhaps the PMI thinking is correct when a well specified project is undertaken. These statements were of course made by people working in an Agile manner and exceeding the expectation is probably quite easy when the customer is not clear on what they want in the first place.

Ideal projects environments

Clearly time – or lack thereof – is an effect of the competitive world of business. Among the research pool there appears to be a lack of time – or ability – to concentrate. What is causing the distractions is hard to tell from the data collected. Are the people disturbed by noise in the environment, interruptions from co-workers, meetings or the need to check their favourite social media account on a regular basis? C1's comment, that '*having time to read and interpret*' is another dimension that clearly comes from the fact, that there is too much to do in the day so tasks are streamlined to help fit everything in. This lack of time could cause cracks to appear in the overall quality of the products. It would be interesting to research how both distractions and lack of time affect the quality.

All projects need some kind of tool chain to at least establish a task list, and track whether or not those tasks are being completed. Where tools came up in discussion, the impression was given that the current tools do not satisfy the needs of the project. It was also apparent that in the larger companies, different departments are using different tools and therefore interdepartmental communication was hindered. A 'one size fits all' tool is usually impossible to find, but it should be obvious that all departments running projects that are related should find common tools.

Closer contact with the customer and more feedback go hand in hand to some extent. How the closer contact is achieved and how the feedback is solicited is important. Faster deliveries facilitate feedback because the customer will, hopefully, have something to say. Getting closer with the customer and building trust is also a good catalyst for getting feedback.

Empowerment is another well-known leadership technique, to motivate individuals and it seems that this extends to teams as well. As discovered in the projects that went well, many felt that being fully responsible was a key for success. Although it is not clear from the data, so conclusions could be drawn that many of the teams have contributors from outside the immediate team. This also indicated that some team members might also work for other teams, too. If the project manager does not have full control on the activities of the team, new kinds of waste can be introduced, for example, waiting. How detrimental this is, is another area for further research.

7 SUMMARY

The original goals of this thesis were optimistic in finding or understanding the methodologies that software companies use to manage projects in Finland. As the thesis progressed this goal became a hard to reach target for various reasons.

Initially, the idea was to interview two people from six companies to control the data integrity by having more than one point of view per company. As volunteers were sought, it became difficult to achieve this, because in some cases only one person was available.

When the companies were approached, all were very willing to have their staff interviewed, but actually securing interviews was somewhat harder. In some cases, although the company's directors were willing, the Project Managers were not as available as first thought. The original plan was that the interviews would be done over a two-week period, but it actually took far longer to get time with some of the participants.

In some cases, securing the interviews also required permission from the company's legal department. For the thesis to be relevant it needed to remain in the public domain, Non-Disclosure Agreements (NDA) could not be signed. This combined with the requirement of some participants to remain anonymous led to some negotiating and ground rules to be laid out before the interviews could commence. In the end, one of the companies was dropped from the research because the legal process was not moving forward in a timely manner.

On the whole, the use of the Appreciative Inquiry during the interview process proved very fruitful. Using such an interviewing strategy set the scene in the interviews and did keep the interviews mostly positive, as was the goal. The interviews were very positive and an enjoyable process.

The data analyses proved tougher than expected. The sheer amount of data produced became a heavy workload to have transcribed and checked for accuracy. Then the breaking of the data into the meta-themes, themes and sub-themes was a very labour intensive exercise, which took far longer than anticipated. It was also mentally demanding and required long period of concentration to complete, not something that could be done

in spare moments. Once the analyses were complete in the Excel sheets, and the themes were identified, writing those out to the discussion chapter was fairly straightforward.

The interviews were not structured, which brings into question the reliability of the research data. The conversational method of the interviews meant that only ideas that came to the participants mind were raised, as a result of this many things were left unsaid – and being unsaid they were hardly taken into account at all. These kinds of phenomena are clearly shown when discussing roles. For example, many mentioned the lead developers but not the rest of the developers in the team. Looking at this another way, the research data highlighted that which is important to the participants and is therefore relevant.

The data analysis was done manually, without the assistance of language analyses tools. A license for this type of tool was not available and the acquisition of such a tool, along with the learning curve required to use it, led to the use of manual methods. This means that the identification of the themes, in other words, how the statements were interpreted, was based on human interpretation and possibly bias in the interpretation. Another person would possibly interpret the statements in a completely different way. In others words, this thesis is heavily based on the author's own thought processes and understanding of the statements.

The number of companies interviewed here is very limited compared to the number of companies that are operating in Finland. This low number cannot be seen as representative of how companies manage software projects in Finland. Nor can the participants in some of the larger companies be seen to be representative of the working practices of the whole company.

After the whole process was finished, probably the most interesting finding in this research was the questions of what motivates project managers. Their experiences of what makes go well and what they see an ideal project environment, is probably the result to take forward.

REFERENCES

A1. 2015. Project leader. Interview 26.5.2015. Interviewer Cork, P. Tampere.

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. 2002. Agile Software development methods. Retrieved 2.6.2015. http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf

Agilemanifesto.org. 2015. Principles behind the Agile Manifesto. Read 8.4.2015. http://Agilemanifesto.org/iso/en/principles.html

Art of Lean Inc. 2006. Basic TPS Handbook. Retrieved 20.4.2015. http://www.artoflean.com/files/Basic_TPS_Handbook_v1.pdf

Ashmore Ph.D, S., & Runyan, K. 2014. Introduction to Agile methods. Addison-Wesley Professional. Print ISBN-13: 978-0-321-92956-3; Web ISBN-13: 978-0-13-343524-5

B1 2015. Project manager. Interview 11.6.2015. Interviewer Cork, P. Tampere

B2 2015. Graphics specialist / project leader. Interview 12.6.2015. Interviewer Cork, P. Tampere.

Berrisford, S. 2005. Using Appreciative Inquiry to drive change at the BBC. Strategic Communication Management 4/2005 p22 - 25.

Blank, S. 2013. Why the Lean start-up changes everything? Harvard Business Review 5/2013.

Brewer, J. L., Dittman, K. C. 2013. Methods of IT Project Management 2nd ed. Purdue University Press. Retrieved 6.4.2015.

Bruch, H., & Meges, J. I. (2010, April). The Acceleration Trap. Harvard Business Review 4/2011. https://hbr.org/2010/04/the-acceleration-trap/ar/1

businessmodelgeneration.com. 2015. Business Model Canvas. Strategyzer AG. Retrieved 21.4.2015.

C1 2015. Project manager. Interview 29.6.2015. Interviewer Cork, P. Espoo.

C2 2015. Project manager / technical leader. Interview 29.6.2015. Interviewer Cork, P. Espoo.

C3 2015. Project manager. Interview 29.6.2015. Interviewer Cork, P. Espoo.

Carter, D. 2005. Appreciative Enquiry. Training Journal 9/2005. p. 25 - 28.

Cooke, J. L. 2012. Everything You Want to Know About Agile : How to Get Agile Results in a Less-than-Agile Organization. Cambridgeshire: Ely.

D1 2015. CEO / game producer. Interview 30.6.2015. Interviewer Cork, P. Tampere.

Davis, A. M., Bersoff, E. H., & Comer, E. R. 1988. A strategy for comparing alternative software development life cycle models. IEEE transactions on software engineering, 14.10.1988. p1453 - 1461.

dictionary.com. 2015. Kata. Read 21.4.2015. http://dictionary.reference.com/browse/kata

DiFalco, R. 2014. When Is Xp Not Appropriate. Read 8.6.2015. http://c2.com/cgi/wiki?WhenIsXpNotAppropriate

Downed, L., & Nunes, P. F. 2013. Big Bang Disruption. Harvard Business review 3/2013.

DSDM Consortium. 2014. The DSDM Agile Project Framework. dsdm.org. http://dsdm.org/dig-deeper/book/dsdm-Agile-project-management-framework

E1 2015. Project manager. Interview 30.6.2015. Interviewer Cork, P. Tampere.

E2 2015. Project director. Interview 30.6.2015. Interviewer Cork, P. Nokia.

Hammarberg, M., & Sundén, J. 2014. Kanban in Action. Manning Publications. Print ISBN-13: 978-1-61729-105-0

Hawks, D. 2012. Kanban vs. Scrum – How to Choose? Read 8.6.2015. http://www.Agilevelocity.com/kanban-vs-Scrum-how-to-choose/

Heusser, M. 2013. 'No-estimates' in Action: 5 Ways to Rethink Software Projects. Read 9.6.2015. http://www.cio.com/article/2381167/Agile-development/-no-estimates-in-action-5-ways-to-rethink-software-projects.html

http://www.techrepublic.com. 2006. Understanding the pros and cons of the Waterfall Model of software development. Read 8.6.2015. http://www.techrepublic.com/article/understanding-the-pros-and-cons-of-the-Waterfall-model-of-software-development/

Karlos, A., Martinsuo, M., & Kujala, J. 2011. Project Business. Helsinki. http://pbgroup.aalto.fi/en/the_book_and_the_glossary/project_business_2011.pdf

Killick, N. 2013. Neil Killick talks about Alternatives to Agile Estimation (#NoEstimates). realestate.com.au 17.8.2013. https://www.youtube.com/watch?v=3YkRkor5a-s

Kim, W., & Mauborgne, R. 2015. Red Ocean Traps. Harvard Business Review 4/2015. https://hbr.org/2015/03/red-ocean-traps

Kinnunen, J. 2014. Lean Practices Applied in an Agile Enterprise Web application Development Project. Tampere University of Applied Sciences. Master’s thesis.

Ladas, C. 2015. Scrum-ban. Read 21.4.2015. http://leansoftwareengineering.com/ksse/Scrum-ban/

Lewis, S., Passmore, J., & Cantore, S. 2008. Appreciative Inquiry for change management. Kogan-page. ISBN: 978 0 7494 5071 7

Little, J. D. 2011. Little's Law as Viewed on its 50th Anniversary. Operations Research, Vol 59 issue 3, p536 - 549.

Loitto, S. 2012. Agile in Waterfall. Industrial Management. Helsinki Metropolia University of Applied Sciences. Master's thesis.

Moreira, M. E. 2013. Beeing Agile: Your roadmap to the successful adoption of Agile. Apress. ISBN: 978-1-4302-5839-1

Mountain Goat Software. 2015. Scrum task Board. Retrieved 16.4.2015. http://www.mountaingoatsoftware.com/Agile/Scrum/task-boards

Mulcahy, R. 2013. PMP Exam Prep (8th ed.). RMC publications.

Mulesoft.org - Rinaudo, R. 2010. Implementing Kanban for Sustaining Engineering. blogs.mulesoft.org. 11.10.2010 http://blogs.mulesoft.org/implementing-kanban-for-sustaining-engineering/

Pahuja, S. 2012. What is Scrumban. Read 9.6.2015. http://www.solutionsiq.com/what-is-Scrumban/

PMI. 2013. A guide to the project management body of knowledge (PMBOK Guide) 5th edition. Project Management Institute.

Radigan, D. (2015, 02). Sprint review at Atlassian: how we do it. Read 16.4.2015. http://blogs.atlassian.com/2015/02/Sprint-review-atlassian/

Reis, E. 2011. The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Business.

Royce, W. 1970. Managing the development of large software systems. Proceedings, IEEE WESCON 8/1970, p328 - 338.

Schwaber, K. 1995. SCRUM Development Process - Advanced development methods. OOPSLA 95. Read 14.4.2015. http://jeffsutherland.org/oopsla/schwapub.pdf

Schwaber, K., Sutherland, J. 2013. The Scrum Guide. Read 14.4.2015. http://www.Scrumguides.org/docs/Scrumguide/v1/Scrum-Guide-US.pdf

Scrum-institute. (2015a). Sprint Burndown. Retrieved 15.4.2015. http://Scrum-institute.org/images_Scrum/Sprint_Burndown.jpg

Scrum Institute. (2015b). Scrum effort estimations - Planning Poker ®. Read 15.4.2015. http://Scrum-institute.org/Effort_Estimations_Planning_Poker.php

Scrum Institute. (2015c). Daily Scrum meeting / Daily stand-up meeting. Read16.4.2015. http://Scrum-institute.org/Daily_Scrum_Meeting.php

Scrum Institute. (2015d). Sprint review meeting. Read 6.4.2015. http://Scrum-institute.org/Sprint_Review_Meeting.php

Takeuchi, H., & Nonaka, I. 1986. The new product development game. Harvard Business Review, Vol 64 Jan/Feb 1986, p137 - 146.

The Economist. 2009. Taiichi Ohno. The Economist 3.7.2009. http://www.economist.com/node/13941150

Thomas, S. 2015. It's a delivery thing - Little's Law – the basis of Lean and Kanban. Read 28.5.2015. http://itsadeliverything.com/littles-law-the-basis-of-Lean-and-kanban

Wells, D. 1999. When should Extreme programming be used? Read 8.6.2015. http://www.extremeprogramming.org/when.html

Wells, D. 2009. Agile Software Development: A gentle introduction. Read 8.4.2015. http://www.Agile-process.org/

Wells, D. (2013, October 8th). Extreme Programming: A gentle introduction. Read 8.4.2015. http://www.extremeprogramming.org/

wikipedia. 2015. Gantt chart. Retrieved 7.4.2015. http://en.wikipedia.org/wiki/Gantt_chart

Yourdon, E. 1997. Death march: The complete software developer's guide to surviving 'mission impossible' projects. Prentice Hall.

Zuill, W. 2012. No Estimate Programming Series – Intro Post. Read 22.4.2015. http://zuill.us/WoodyZuill/2012/12/10/no-estimate-programming-series-intro-post/ 10.12.2012